Class announcements

- For Next Class
  - Read: MySQL Database Case

- For next Thursday
  - Database Project Due

- Database tutorials

Student Presentations

BUSINESS PAPER

Business Analysis Paper

- The paper should answer questions like:
  - What industry does the company compete?
  - What is the competitive environment like?
    - "Porter model" (competitors, substitute products, buyers, suppliers, new entrants.)
  - How did your company use IT to gain a competitive advantage?
  - How did the company use IT to support or enable its business processes and competitive strategies?
    - What technologies in particular did it use?
    - How much of the company’s success do you attribute to its use of IT and/or the company’s early adoption of IT into their business processes?

(Detailed guidelines are posted on the website)
Citing Sources

- Plagiarism is illegal and cheating and will not be tolerated!!!
- More than thirty words verbatim must be cited.
- Any facts or figures that are not your own must be cited.
  - Ebay's revenues in US Revenues in 2002 were $1.39 billion [1].

Citing Sources

You must cite your sources in the body of the text!!!!

"Semiconductors have found a place in virtually every electronic device in existence. This helps explain why the industry was able to reach $200 billion in sales before a slump brought the figure back down in 2001" [1].

End Note:


Suggested sources of Information

- Company website
- 10K report

  - This is the annual report public companies file with Security and Exchange Commission.
- Business article database

  - A database of articles from magazines like "Business Week" and economic journals.
  - Find it at: [www.bloomberg.com]
  - Click on "articles database" on left margin
  - Click on "by subject"
  - Click on "economics"
  - Try this tonight! And let us know if you have problems
- Industry specific publications
  - Example: Aviation Week
  - Books
  - Good Magazines (The Economist)
- Consulting groups: Forrester, Gartner, ...

More on layering

Slides modified from those by David G. Messerschmitt

Data and information

Application
Deals with information
Assumes structure and interpretation
Infrastructure
Deals with data types
Assumes standard data types
Architecture

Components

Component: A subsystem purchased "as is" from an outside vendor
(Alternative – building your own subsystem)

A component implementation is encapsulated (although often configurable)

Other Examples of components

Computer  
Disk drive  
Network  
Network router  
Operating system  
Integrated circuit  
Database management system

Why is a component implementation encapsulated?

Two ways to design a system

Available components

System requirements

Requirements

Decomposition from system requirements

Assembly from available components

HHC Architecture

The Palm OS we are buying "off the shelf" and integrating into our architecture. The Palm OS is a component.

Interoperability

- Components are interoperable when they interact properly to achieve some desired functionality
- Increasingly component interoperability cannot be dependent on end-user integration
  - PC and peripherals
  - Enterprise, inter-enterprise, consumer applications
  - Role for standardization
Outsourcing

- Outsourcing: A subsystem design is contracted to an outside vendor.
- Responsibility is delegated.

HHC Architecture

- HHC Application
- Coordination With HHC Server
- User Interface
- Data Management

- Suppose we choose to pay another firm to develop the user interface.
- This is called Outsourcing.
- Why would we do this?

System Integration

- Suppose we:
  - Bring together all these subsystems
  - and test them...
- This is called System Integration.

System integration

- Bring together subsystems;
- make them work together;
- to achieve a goal.

- Requires:
  - Testing
  - Making modifications to architecture and/or subsystem implementation

Can System Integration be Outsourced?

- Of course!

Supplier Types

- Three types of suppliers:
  - Component Suppliers
  - Custom Subsystem Developers
  - System Integrators
- (Some suppliers are 2 or even 3 of above.)
Two ways to sell Software

- **Product**
  - Customer installed and operated
  - Often (but not necessarily) sold or licensed at a fixed price

- **Service**
  - Functionality provided over a wide-area network
  - Often (but not necessarily) sold by subscription

Recall: Infrastructure and Applications

**Infrastructure**
- Equipment and/or software used by many applications

**Applications**
- Provide specific capabilities and features serving individual users.

Industry Structure (cont’d)

Four possibilities

<table>
<thead>
<tr>
<th>Product</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Office</td>
<td>Hotmail</td>
</tr>
<tr>
<td>Application</td>
<td>Infrastructure</td>
</tr>
<tr>
<td>Personal computer</td>
<td>Internet DNS</td>
</tr>
</tbody>
</table>

Application Service Provider

- Two types
  - Bundled
    - An infrastructure provider bundles applications with their infrastructure
    - Example: AOL, telephony service providers
  - Unbundled
    - A provider of an application service without providing an infrastructure service
    - Examples?

Examples of unbundled ASP model

- Yahoo: Web-based calendar
- Hotmail: Web-based email
- Schwab: Web-based stock trading
**Unbundled ASP model**

**Advantageous to user**
- Proven way to reduce installation, integration, and maintenance costs
- Contractual obligation for availability and quality
- Location independence

**Unbundled ASP model (con’t)**

**Advantages to supplier**
- Ongoing revenue stream supporting upgrade and maintenance
- Usage-based revenue better aligned with user’s value proposition
- Opportunity for price discrimination, advertising revenue, etc.

**Some pricing alternatives**

- Price discrimination?
- Usage dependent?
- Terms and conditions
  - fixed, leasing, per-use, subscription
  - warranty, service level agreements
- Bundles
  - maintenance, support, releases, provisioning and operations
- Who pays?
  - sometimes not the end user

**Infrastructure acquisition**

- Build and operate
- Build but do not operate
- Do not build but operate
- Neither

**Trend**

- Outsourced operations
- System integrator
- Service provider

**Application acquisition**

- Application
  - Develop internally
  - Buy as product
  - Contract development
  - Product w/ customization

- Software supplier
- Outsource developer
- Supplier, consultants

**Stovepipe vs. Integrated Infrastructure**

- **Stovepipe architecture**
  - Turnkey Solution
    - Single supplier provides all encompassing solution
    - (complete with infrastructure)

- **Integrated Infrastructure**
  - Separate infrastructure that can support many applications
From stovepipe to layering

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.

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

Role of Venture Capital in Computing.

- Open interfaces allow small firms to contribute components without having to develop entire solution.
- Fast decision making and no supplier lock-in.
- Other Advantages?

Standardization

- 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

Purpose of a standard?

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
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

de jure
- GSM, ISON Telephone interface
de facto
- Microsoft Windows API (Application Programming Interface)
- Intel Pentium instruction set
- Voluntary industry standards body
- IEEE (Institute of Electrical and Electronic Engineers)
- IETF (Internet Engineering Task Force)
- Industry consortium
- W3C (World Wide Web Consortium)
- SET (Secure Electronic Transactions)
- Best practice
- Windowed GUI

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/extendible approaches for flexibility
  - e.g. XML, Java

Old giving way to the new

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

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

Approaches

Consensus
- ISO

Collaborative design
- MPEG

Competitive "bake off"
- ETSI

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

- 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

Application I
- Aggregation accessing multiple databases
- Sharing two or more applications accessing the same databases

Application II

Databases

Relational table

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td></td>
</tr>
<tr>
<td>Dept</td>
<td></td>
</tr>
<tr>
<td>Record</td>
<td></td>
</tr>
</tbody>
</table>
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

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.

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 die
  - number
Move token to indicated square
  - yes
  - no
  - "jail" square
  - "board"
Move to "jail" square
  - yes
  - no
Land on "jail" square
  - yes
  - no
Finish
  - turn

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

Player 1

One-turn algorithm

Player 2

This is a protocol interaction diagram

Time

Application and infrastructure

The application defines its own application-level protocols

Internally, the network uses protocols to implement the services it provides

Example:

Layered Protocols Example

Three simple protocols

One-way message: send-receive

Two-way interaction: request-response

Push: publish-subscribe

Send - Receive
**Request - Response**

![Request - Response Diagram](image)

**Send - Acknowledge**

![Send - Acknowledge Diagram](image)

**Example: HTTP (Hyper Text Transfer Protocol)**

1. User activates URL
2. HTML document (or invokes “helper application”)
3. Browser displays document (of HTML) or invokes “helper application”
4. HTTP request
5. HTTP response

**Locating things**

by
David G. Messerschmitt

**Three ways of locating things**

**Name**
- "Joe Blog"
- 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 to is (R,D,D,R,R,R,R)
Is (R,D,D,R,R,R,R) an address?
No! -- not an address, because it depends on starting point

Example

Address of 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
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
<html>
  <head>
    <title>Super Widget</title>
  </head>
  <body>
    <h1>Super Widget</h1>
    <h2>Widgets Incorporated</h2>
    <p>123456789</p>
    <p>$300</p>
  </body>
</html>
```

Example: XML

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

XML in Ecommerce example

```
<stuff4u>
  <product>
    <name>Super Widget</name>
    <price>$300</price>
  </product>
  <product>
    <name>Amazing Gadget</name>
    <price>$500</price>
  </product>
</stuff4u>
```

XML in ecommerce example 2

```
<xyzman>
  <productinfo>
    <product>
      <name>Super Widget</name>
      <price>$300</price>
    </product>
    <product>
      <name>Amazing Gadget</name>
      <price>$500</price>
    </product>
  </productinfo>
</xyzman>
```
Family lineage

- SGML: Standardized in mid 90s by ISO
- HTML: Emphasizes formatting and presentation of documents
- XML: Prepared in mid 90s
  - Emphasizes structure of documents
  - Purpose-and industry-specific extensions

Distribution for Midterm

<table>
<thead>
<tr>
<th>Score Range</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0, 10)</td>
<td>1</td>
</tr>
<tr>
<td>(10, 20)</td>
<td>1</td>
</tr>
<tr>
<td>(20, 30)</td>
<td>4</td>
</tr>
<tr>
<td>(30, 40)</td>
<td>12</td>
</tr>
<tr>
<td>(40, 50)</td>
<td>15</td>
</tr>
<tr>
<td>(50, 60)</td>
<td>10</td>
</tr>
<tr>
<td>(60, 70)</td>
<td>10</td>
</tr>
<tr>
<td>(70, 80)</td>
<td>30</td>
</tr>
<tr>
<td>(80, 90)</td>
<td>70</td>
</tr>
<tr>
<td>(90, 100)</td>
<td>10</td>
</tr>
<tr>
<td>(100, 110)</td>
<td>5</td>
</tr>
</tbody>
</table>