Class announcements

- For Next Class
  - Read: MySQL Database Case
- Database tutorials

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>Internet DNS</td>
</tr>
<tr>
<td>Infrastructure</td>
<td>Personal computer</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 (cont’)**

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
- warrantee, service level agreements
Bundles
- maintenance, support, releases, provisioning and operations
Who pays?
- sometimes not the end user

**Infrastructure acquisition**

Infrastructure

- 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

Trend

- Software supplier
- Outsourcing developer
- Supplier, consultants

**Stovepipe vs. Integrated Infrastructure**

**Stovepipe architecture**

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

**Integrated Infrastructure**

- Separate infrastructure that can support many applications

**From stovepipe to layering**

Many applications

Integrated Infrastructure
( Maybe broken into Additional layers.)

Application-dependent infrastructure

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.

**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
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

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, ISDN 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/extensible approaches for flexibility
  - e.g. XML, Java

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, xB6, 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 lock-in.

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"**
  - 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](mailto:david.messerschmitt@cmu.edu)
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></td>
<td>Name</td>
</tr>
<tr>
<td>Record</td>
<td></td>
</tr>
<tr>
<td>Field</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

- "PROJECT" each operation results in a new table

<table>
<thead>
<tr>
<th>Departments</th>
<th>Employees</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Database Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passengers</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Alice</td>
</tr>
<tr>
<td>Bob</td>
</tr>
<tr>
<td>Chris</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Departments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Dept ID</td>
</tr>
<tr>
<td>Alice</td>
</tr>
<tr>
<td>Engineering</td>
</tr>
<tr>
<td>Bob</td>
</tr>
<tr>
<td>Engineering</td>
</tr>
<tr>
<td>Chris</td>
</tr>
<tr>
<td>Sales</td>
</tr>
</tbody>
</table>

JOIN
Fields, columns, attributes

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

Records, rows

---

**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

Decision

Test

Finish

Sequence

Selection

Loop

Algorithm building blocks

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

Player 1

Player 2

One-turn algorithm

Time

This is a protocol interaction diagram.

Layered Protocols Example

HHC Server

Application Level Protocol

Send Pass. Data As Message

HHC Application

Send Packet

Palm OS

Networking Infrastructure

Windows OS

Break Messages into Packets

Networking Infrastructure

Send Packet

Acknowledge Packet

Link Level Protocol

Three simple protocols

One-way message: send-receive

Two-way interaction: request-response

Push: publish-subscribe

Send - Receive

Client

send

receive

Server

Time

Request - Response

Client

request

response

Server
Send - Acknowledge

Example: HTTP (Hyper Text Transfer Protocol)

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 to is (R,D,D,D,R,R,R,R)

Is (R,D,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>
<h1>Super Widget</h1>
<h2>Widgets Incorporated</h2>
<em>123456789</em>
<p>$300</p>
</html>
```

---

**Example: XML**

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

---

**XML in Ecommerce example**

```
Supplier
```

```
Retailer
```

```
Consumer
```

```
Stuff4U
```

```
Amazing Gadget $500
```

```
Supplier
```

```
Retailer
```

```
Consumer
```

```
Product info
From each Supplier sent in XML
```

```
Product info
From each Supplier sent in XML
```

---

**XML in ecommerce example 2**

```
Supplier
```

```
Retailer
```

```
Consumer
```

```
Stuff4U
```

```
Amazing Gadget $500
```

```
Supplier
```

```
Retailer
```

```
Consumer
```

```
XYZ Manufacturing
```

```
Super widget recognized and managed by SCM software.
```

---

**Family lineage**

- SGML
  - Standardized in mid 80s by ISO
  - Introduced in Early 90s
  - Emphasizes formatting and presentation of documents

- HTML
  - Emphasizes structure of documents

- XML
  - Proposed in mid 90s
  - Purpose- and industry-specific extensions