Outline For Today

- Class Announcements
- Student Presentation
- Markup Languages
- Middleware
Class announcements

- Database Assignment due date extended!
  - Wednesday May 25th

- Shiv out of town May 17 - 23

- Assignment 5 Due May 18 (Today!).

- Read Ch 18 for Monday

- Student Presentation Monday
  - Michael Ponce
  - Open Slot - Any Volunteers?
Student Presentations

- Sean Martin
- Cathy Zhu
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>
<br>
<p> $300 </p>
</html>
```
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>
    <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 Stuff4U

Retailer

Stuff4U Stuff4U

Super Widget $300

Amazing Gadget $500

Consumer
XML in ecommerce example 2

Product info
From each Supplier sent in XML

XYZ Manufacturing

Super widget recognized and managed by SCM software.

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

**SGML**
- Emphasizes formatting and presentation of documents
- 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
Data sharing among applications

Options include:

- Messages with defined formats
- Documents (eg. XML)
- Shared databases
- Remote method invocation middleware

The first two are the most practical for inter-enterprise applications
Middleware
Middleware
Middleware

- **Layer of Software between OS and Applications**
  - Hide Heterogeneity of Operating Systems
    - Make it possible for an application to run on a machines with different operating system.
  - Provide Common Services to Applications
    - ex: allow objects on different machines to invoke each other’s methods
    - This is called *Remote Method Invocation* (RMI)
Middleware

- Easier to introduce new middleware than new OSes
- Thus, most innovations in infrastructure software happen in middleware
Example: IBM’s software strategy revolves around middleware

- DB2
  - DBMS middleware
- WebSphere
  - Supports web application servers
- Tivoli
  - System management and Configuration management tools
- Rational
  - Software development framework
- Lotus
  - Document Management and Collaboration Tools
IBM

Does IBM make most of its revenue from hardware or software?

Neither!!

<table>
<thead>
<tr>
<th>2004</th>
<th>Revenue</th>
<th>Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services</td>
<td>$46 billion</td>
<td>$34 billion</td>
</tr>
<tr>
<td>Hardware</td>
<td>$31 billion</td>
<td>$21 billion</td>
</tr>
<tr>
<td>Software</td>
<td>$15 billion</td>
<td>$1.9 billion</td>
</tr>
</tbody>
</table>
Middleware

**Some Categories**

- **Transaction processing**
  - Simplify the coordination of complementary resource managers

- **Message-oriented middleware**
  - Support message and queuing capabilities where resource mangers are *not* available simultaneously (like workflow)
Some Middleware Categories

Distributed object management
- Support applications that are distributed across heterogeneous platforms and organizations

Mobile code
- Allow application code to be moved and executed on heterogeneous platforms
- Without prior software installation
Message Oriented Middleware

- Allows applications to send messages to each other asynchronously.
  - Remote application may not be running
  - Remote computer may not be on

- MOM queues the messages until the receiver is ready to receive.
Message Ordering Middleware

Business management application decides to make a widget.

Application

MOM

“Make a Widget!”

Message Queue Managed by MOM

Application

MOM

Factory
Message Oriented Middleware - Products

- Sun
  - JMS - Java Messaging Service
    - Specifies the API (interface) to MOM
  - Java System Message Queue
- IBM
  - WebSphereMQ
    - Supports JMS and IBM's API MQI
- Microsoft MSMQ
- VeriQ VCOM
Mobile code and Java

by
David G. Messerschmitt
Reminder: two key requirements

With networks, new emphasis on:

- **Portability**: applications run across multiple platforms (avoid lock-in)
- **Interoperability**: pieces of application must work together (benefit from network effects)
Dynamic portability: mobile code

☆ Send code (as a message) to a host

Mobile code:
Code moved to
A remote computer and
executed there

�行 the program
represented by that code
Dynamic portability: mobile code

Send object *with all instance data*

**Mobile Object:**
Object with instance data pushed to remote Computer.
Mobile Agents

- A Mobile Object with ability to adaptively adjust its itinerary is a Mobile Agent.
Some mobile code advantages

- Executing program closer to user can enhance interactivity
- Shifting location of computation can enhance scalability
- Mobile code originating from a common source can enhance interoperability and bypass network effects
Idea of mobile agents

☆ Agent launched

ⓘ Agent executes in each host, modifying its state

MA

ⓘ Agent returns
Java virtual machine

Java program

Compilation

JIT compiler

VM interpreter

Bytecode: low level but machine independent

Native machine instructions

Mobile code
VM as spanning layer

Applications

JavaBeans component framework

Java virtual machine

Operating system

Java VM spanning layer
SUN/Java strategy

License Java freely, even to rival Microsoft

- Why?

License terms give Sun a modicum of control over the “standard”

- Why?

How does Sun expect to make money?
Distributed Object Management

Client Object \rightarrow Server Object

Remote Method Invocation

Client Object \leftarrow Server Object

Return Data

Execute
Distributed object management

Emphasis is on interoperability

- Allows objects on one host to invoke methods of objects on another host
- Platform, language independent

Portability is not the emphasis
Interoperability requires:

- Common structure of data
- Common interpretation of data
- Agreement on protocols
Before and after
What is the acronym?

* Common Object Request Broker Architecture
  
or

* Concerned Off-Road Bicyclist Association?
Protocol layer

Location-independent application
Object Management Group CORBA standard

Application
Object request broker
Internet Inter-ORB Protocol (IIOP)

User datagram protocol (UDP)
Transmission control protocol (TCP)

Internet protocol (IP)
Subnetworks
Portability not promised

Location-dependent application

Application 1  Interoperability  Application 2

ORB 1  ORB 2

Internet Inter-ORB Protocol (IIOP)

CORBA standard does not insure ORB-to-ORB portability
OMG process

Identify need
Request for proposals
Process to

- choose best
- or ask proposal advocates to work together