The Use of Java™ and Web-based Architecture in Mission Management Applications

GSAW Feb 21-23, 2001

Bruce Bohannan
Linda Martz
Agenda

- Evolving Needs of Mission Management
- Web Impact of Development
- Enabling Technologies
- Side-benefits of Java™
### Evolving Needs of Mission Management

#### Growth of Complexity

<table>
<thead>
<tr>
<th>Number of Operators</th>
<th>Simple</th>
<th>Complex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>Multiple</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Collaborating Operators</th>
<th>Single user as coordinating point for multiple “virtual users”</th>
<th>Multiple</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• procedural, human intensive</td>
<td>• procedural inadequate because remote users</td>
</tr>
<tr>
<td></td>
<td>• SW managed</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Interest, eyeballs</th>
<th>Only operators see products</th>
<th>User Community wide area visibility required</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Resources</th>
<th>Single resource, single type</th>
<th>N Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Multiple types</td>
</tr>
<tr>
<td></td>
<td></td>
<td>System-of-systems</td>
</tr>
<tr>
<td></td>
<td></td>
<td>P2P collaboration</td>
</tr>
</tbody>
</table>
Evolving Needs of Mission Management

**Web Impact on User Expectations**

- Portability
- Access from anywhere
- Access to “any” necessary data
  - Search engines
  - Links to related “stuff”
  - Data fusion
- Easy version releases (download new versions)
- Light weight client boxes
- User defines their view of system (www.mymissionplanning.gov)
- Interfaces to legacy
Enabling Technologies

Architecture Principles

• Component-Based Architecture
  – Clearly defined interfaces exist between each layer
    • Provides flexibility in deployment and implementation
    • Components in one tier can be modified and replaced independent of the other tiers
  – Accommodates rapid evolution

• N-Tier Architecture
  – Enables portability through easy deployment on a different environment (platforms, databases, user interface)
  – Scalability through ability to upgrade servers and load-balancing across multiple servers without impact to rest of architecture
Enabling Technologies

Commercial Standards/Products

- J2EE™
- Corba/RMI
- Web Browsers
- Application Servers
- DBMS evolving into application/web servers
  - Products to provide web data content
  - Gui builders
  - Interfaces to Corba, etc.
- HTTP
- XML
- EJ Bs, SJBs
- Java™
- New development methodologies
  - Extreme programming
  - RUP
Enabling Technologies

J2EE™

Client-Side Presentation

Server-Side Presentation

Business

Data

Hybrid

Java™ Applications

CORBA

Legacy Backend C++

ODBC

Database Server

J2EE™

Java™/C++ Applications

RMI/CORBA

Entity EJBs

JDBC

Database Server

Web Browser

Java™ Applets

Servlets

RMI/CORBA

Request

Legacy Backend C++

(Related) C++ Code

CORBA

SQL

Legacy Systems

HTML/JavaScript

Java™/C++ Applications

Servlets

HTTP

HTML/XML
Enabling Technologies - J2EE™
MM N-Tier Implementation (example)

Client-Side Presentation
- Event Builder
- Task Builder
- Scheduling I/F
- Timeline
- Map
- Performance Assessment
- Schedule Window Manager
- Message Handler

Server-Side Presentation
- Event Ingestion
- Event Search
- Event Editor
- Task Ingestion
- Task Selector
- Task Search
- Session Manager
- Opportunity Gen.
- Batch Schedule
- Manual Schedule
- Schedule Export
- Schedule Copy
- Timeline
- Map Interface
- Report Selection

Business Tier
- Session EJBs
  - Task Manager
  - Event Manager
  - Target Manager
  - Resource Manager
  - Visibility Manager
  - Activity Manager
  - Allocation Manager
  - Schedule Session
  - Sched. Window Mgr.
- Entity EJBs
  - Task
  - Event
  - Target
  - Resource
  - Activity
  - Allocation
  - Window Lock

Stand-alone Applications
- Perf. Assessment Scripts
- Visibility Generation
- Ephemeris Propogation

Data
- Tasks
- Events
- Targets
- Resources
- Activities
- Allocations
- Window Locks
Side-benefits of Java™
Software Cost Equation

Cost of S/W Functionality = \left( \frac{\text{Total Functionality}}{\text{Staff-Months}} \right) \left( \frac{\text{New LOC}}{\text{New Functionality}} \right) + \left( \frac{\text{Effort in Staff-Months}}{\text{New LOC}} \right) + \left( \frac{\text{Reuse LOC}}{\text{Reuse Functionality}} \right) + \left( \frac{\text{Integration effort in Staff-Months}}{\text{COTS Functionality}} \right) + \text{Cost of COTS}

- We have identified potential for up to 12% project savings in development when using Java™
Side-benefits of Java™ - Inherent Savings

Line of Code Counts

- Graphics
- Text Disp
- Algorithm
- DB Access
- Framework

Java
C++
**Side-benefits of Java™ - Inherent Savings**

**Increased Productivity**

- Even if Java™ was equal to C++ in LOC counts, it is quicker to build
  - *More built-in functions* – This includes both built-in libraries as well as automatic memory management and support for multi-threading.
  - *Simpler language syntax* – Features which make Java™ both easier to learn and use, such as lack of pointers, array checking, single inheritance model, and no operator overloading.
  - *Faster code iteration* – No waiting for compiles and links.
  - *Better documentation* – Due to automated document generation using Javadoc as well as quantity of online tutorials and examples.
Side-benefits of Java™ - Inherent Savings

Reduced Cost for COTS Usage

- Java™ development may required as much as $300K less COTS purchases
  - Some products used for Java™ are less than C++
    - ex: C++ must be purchased, Java™ is free
    - Profilers, visual development tools
  - Some products not needed for Java™ because of built-in capabilities
- C++ purchases not required for Java™
  - Object Request Broker
  - Distribution Middleware
  - Utility libraries (Objectspace, Dbtools, XRT/Table)
- Java™ needs not used with C++
  - Browser
  - Application Server

GSAW 2001
Side-benefits of Java™ - Process-Dependent Savings

Life Cycle Quality Improvements

- Requirements Analysis
  - Interface partitioning
  - Adaptation to evolving environment

- Design
  - Memory Management
  - Pointer Usage
  - Reference Semantics
  - Interface partitioning
  - Inheritance Model
  - Unit partitioning
  - Class Hierarchy
  - Real time type info
  - Everything is an object

- Maintenance
  - All items that support simplification of Design, Construct and Testing improves maintenance

- Construct
  - Memory Management
  - Pointer Usage
  - Reference Semantics
  - Concurrency/Multi-threading
  - Compile time checking
  - Built in libraries
  - Documentation

- Testing
  - Memory Management
  - Pointer Usage
  - Compile time checking
  - Run Time checking
  - Exception handling
  - Documentation

GSAW 2001
Side-benefits of Java™ Staffing

• Java™ may help improve our ability to staff future projects:
  – Hiring - More new hires are going to know Java™ than C++ since it is becoming the language of choice for computer science education.
  – Training – For the people that have to learn on the job, Java™ is simpler to learn.
  – Retention - Developers find Java™ more “fun” and on the leading edge of technology, which will help staff retention.
Java™ is Not a Silver Bullet

- Performance – Will Java™ provide the operational performance required by our customers?
  - Preliminary benchmarks show Java™ improvements have made this a minor issue
  - Want to double check for specific problem areas

- Large-scale Development Configuration Management-
  Developments with greater than 10 people may not be able to fully use the Java™ Interactive Development Environments (IDEs) because of their immature interaction with CM tools.

- Size - How big can a system get before it will substantially impact the Virtual Machine?
Conclusions

• The Market is Driving Architecture Changes
  – Users more sophisticated because of the Internet
  – Focus on standard interfaces (HTTP, XML for physical implementation)

• Evolution is Do-able
  – Separation of client and server applications
  – Legacy components become plug-ins