Java™ 2 Enterprise Edition Seminar

Presented By:
Capstone Consulting
Seminar Agenda

• 0800 – 0830 Continental Breakfast
• 0830 – 0845 Capstone Introduction – Dave Holloway
• 0845 – 0930 J2EE Overview – Randy Ott
• 0930 – 0940 Brief Intermission
• 0940 – 1040 J2EE Technologies – Brad Little
• 1040 – 1050 Brief Intermission
• 1050 – 1130 J2EE Benefits & Business Cases – Jim Richards
• 1130 – 1145 BEA Presentation – Craig Fortune, BEA Systems
• 1145 – 1300 Q & A Luncheon
Capstone Introduction
Is your success dependent on the rapidly changing IT industry?

If it is, a stable foundation is vital for survival.

Capstone provides the critical link to bring a project to successful completion.

At Capstone we never lose sight that all of the pieces must work together to provide the finished product.
Capstone Consulting

- Capstone Consulting is an e-solutions and Professional Services technology company

- Capstone focuses on providing products and services for existing and emerging OO and e-merging technologies such as J2EE, EJB, Java, XML, Web Services, C++, and CORBA

- Capstone Consulting holds a strong IT foundation which is important to the success of a company
Capstone Services

- **Professional Services**: Consulting for System Architecture Analysis & Design, Application Development
- **Mentoring**: working side by side with your staff, Capstone experts enable your team to assimilate this technology to quickly increase productivity
- **Training**: As part of our full-service offerings, Capstone provides Training Courses in Object Oriented Technologies including, J2EE, Java, XML, Web Services, OOAD/UML, and C++
- **Outsourcing**: Capstone welcomes the opportunity to bid on projects in which you may have a need to leverage the latest in software technology
Capstone Offerings

• Capstone provides consulting and courses in a variety of emerging technologies

• We offer services in middleware tools, including
  – J2EE with a focus on WebLogic Server
  – WebSphere
  – iPlanet
  – SilverStream
Capstone Offerings

• In addition to our J2EE services, we offer mentoring and courses on
  – Object Oriented Analysis and Design (using the UML) with tools such as Rational and TogetherSoft
  – Our newest courses focus on the rapidly evolving middleware technologies such as B2B, workflow and XML/Web Services
Capstone Overview

- Average 10+ years of experience
- Extensive backgrounds in distributed computing
- Development focus on design and reuse
- BEA Partner
- BEA Certified Instructors
- IBM Certified Instructors
We Can...

• Provide the foundation for e-Commerce and B2B applications
• Provide the foundation for integration with legacy systems
• Design and develop robust, reusable, maintainable systems
• Train and mentor your employees in the latest technologies
J2EE Overview
Application Evolution

• Mainframe
• Client Server
• Distributed Computing
Mainframe Applications

- Presentation, Business Logic, and Data Access all in one application
- User Interface hard to customize or make user-friendly
- Business logic in hard-to-extend 4GL like COBOL
- Data stored across disparate files and structures
Client Server Applications

• Prior to mid 1990s, most applications executed on a single machine

• The first attempt at distributing an application:
  – Application broken into a series of client-server processes
  – Local copies of databases which were replicated and synchronized often

• Problems with client-server approach:
  – Single point of failure with processes
  – Hard to scale ("Ripple Effect")
  – Proprietary protocols
What is Middleware?
What is Middleware?

• Middleware is the software (connective glue) between the application programs you use and the operating system and base networking capabilities of your computer.
• Some refer to it as the *slash* in client/server.
What is Middleware?

- Middleware provides a set of higher-level distributed computing capabilities with standards-based interfaces.
- These interfaces allow applications to be distributed more easily and to take advantage of other services provided over the network.
What is Middleware?

- Middleware can be viewed as a set of services that are accessible to application developers through an API.
- An example of a middleware service is a remote procedure call (RPC) that allows functions on a remote server to be accessed transparently by the developer.
Why use Middleware?

- Application developers and system integrators can use middleware services as building blocks to construct enterprise-wide information systems that use distributed computing resources effectively.
Middleware Products

- Remote Procedure Calls (RPC)
- Transaction Processing Monitors (TPM)
- Message Oriented Middleware (MOM)
- Object Request Brokers (ORB)
- Application Servers
Distributed Application Goals

• High Availability
  – Systems need to be up 24/7/365 for client use

• Scalability
  – Additional resources can be easily added to meet growing demand

• Maintainability
  – Keep business logic in reusable units
  – Isolate database access from the user interface
Applications can be divided into 3 layers:
- Presentation Logic
  - Users view of application (windows, browsers, command line)
- Business Logic
  - The components that implement your business practices and policies
- Data Access Logic
  - Retrieval and storage of data to RDBMS, files, legacy systems.

By keeping these layers separate, reusability and maintainability are maximized.
What is J2EE?

• Bundle of Java™ technology standards

• Middleware infrastructure that serves as the building block upon which highly available, scalable, and maintainable enterprise systems can be built
Java 2 Enterprise Edition™ (J2EE)

- Early 90’s, shift from client-server to multi-tier application models
- Sun Microsystems development efforts led us to the J2EE standard
- Convergence of three concepts
  - Server-side behaviors on any web server
  - Connectors to existing enterprise systems
  - Modular, self-contained, easy to deploy components
- J2EE addresses the problems of developing a distributed application
- Java 2 Platform, Enterprise Edition:
  - http://java.sun.com/j2ee/
High-Level J2EE Architecture

J2EE Platform

Clients
- Web Browsers
- Desktop Applications
- Mobile Devices (WAP)

The Enterprise

Presentation Logic

Business Logic

Backend Enterprise Information Systems
What is J2EE? (cont.)

- A reference platform
- Implementations of this platform are referred to as Application Servers
- Market Leading Application Servers
  - BEA WebLogic Server
  - IBM WebSphere
  - Sun/Netscape iPlanet
  - Allaire Jrun
  - And others…
Motivation for J2EE

- Faster time to market
- Shortage of developers
- Multiple platforms
- Distributed applications
- Reduced costs
- Component Reuse
- Ease of Integration
Motivation for J2EE (cont.)

- Easy-to-access services to their customers, partners, employees, and suppliers
- Standard for developing enterprise applications
- Making middleware easier, less complex coding
J2EE Applications

• Becoming the de-facto standard for distributed component development
  – **Highly Available**, to meet the needs of today’s global business environment.
  – **Scalable**, to insure that business transactions are promptly processed
  – **Maintainable**, by breaking business logic into reusable components
  – **Secure**, to protect the privacy of users and integrity of the enterprise.
  – **Reliable**, to ensure that business transactions are accurately processed
J2EE Platform Benefits

• Simplified architecture and development
  – Maps easily to application functionalities
  – Enables assembly-time and deploy-time behaviors
  – Supports separation of expertise

• Scalability and automatic load-balancing to meet demand variations (I.e. Connection Pooling for faster access to data)
J2EE Platform Benefits (cont.)

• Integration with existing information systems
  – Accessing relational data from Java™
  – Managing and coordinating transactions across heterogeneous enterprise information systems
  – Accessing information in enterprise naming and directory services
  – Sending and receiving messages via enterprise messaging systems
  – Sending and receiving e-mail
  – Calling Common Object Request Broker Architecture (CORBA) services
Competing Platforms?

Microsoft.NET  J2EE

How do they compare?
.NET vs. J2EE Summary

• Features:
  – .NET and J2EE offer pretty much the same laundry list of services but in different ways

• Portability:
  – .NET works on Windows platform only
  – J2EE works on any platform with a compliant Java Virtual Machine and set of required platform services
.NET vs. J2EE Summary

• Languages:
  – .NET theoretically supports development in many languages (once IL compilers are created for them)
  – J2EE is a single-language platform

• Interoperability
  – .NET and J2EE both provide for a SOAP based messaging scheme to allow components from each platform to communicate with each other
Gartner Group on J2EE…

• J2EE is rapidly becoming the middleware of choice, according to analysts. "J2EE is the standard for application-server middleware today," says Daryl Plummer, group VP for Internet and E-business technologies at Gartner. "If you don't have J2EE compliance, you'd better be Microsoft; if you're not Microsoft, you better get J2EE compliance real quick."
Gartner Group Forecast

- 60% of all new applications will use Java by 2004
- “By 2003, Java will be the dominant platform used to process server side XML…”
Intermission

Duration: 10 minutes
J2EE Technologies
J2EE Architecture
J2EE Standard Services

- HTTP
- HTTPS
- Java™ Transaction API (JTA)
- RMI-IIOP
- Java Naming and Directory Interface™ (JNDI)
- JDBC™
- Java™ Message Service (JMS)
- JavaIDL
- JavaMail™
- JavaBeans™ Activation Framework (JAF)
Java Transaction API

- JTA consists of two parts:
  - An application-level demarcation interface that is used by the container and application components to demarcate transaction boundaries.
  - An interface between the transaction manager and the resource manager used at the J2EE SPI level (in a future release)
HTTP and HTTPS

• HTTP is:
  – the HyperText Transport Protocol
  – HTTP client-side API is defined by the `java.net` package.
  – HTTP server-side API is defined by servlet and JSP interfaces

• HTTPS is:
  – A secure version of HTTP using Secure Socket Layer (SSL)
  – HTTP over SSL is supported by the same APIs as HTTP
Remote Method Invocation (RMI)

• RMI is a mechanism for invoking methods on remote objects
• RMI uses serialization to pass data by value between two objects
RMI-IIOP

- The RMI-IIOP subsystem is composed of APIs that allow for the use of RMI-style programming that is independent of the underlying protocol.

- Provides an implementation of these APIs that support both the J2SE native RMI (JRMP) and the CORBA IIOP protocol.

- J2EE Applications are required to use the RMI-IIOP protocols when accessing EJB components.
Java Naming and Directory Interface (JNDI)

- The JNDI API:
  - is the standard API for naming and directory access
  - Has two parts:
    - Application-level API for application components access to naming and directory services
    - Server provider interface to attach to a provider
Java DataBase Connectivity (JDBC)

- JDBC is the API for connectivity with database systems
Java Message Service (JMS)

- JMS is:
  - A standard API for messaging that supports the point-to-point messaging model as well as the publish-subscribe model
  - J2EE 1.2 specification requires JMS to be available, but does not require any messaging service implementation to be available
  - Future version of J2EE will require an implementation of both models
JavaIDL

- JavaIDL allows J2EE components to invoke external CORBA objects using IIOP protocol

- CORBA objects may be written in any language, and typically exist outside of a J2EE product
JavaMail & JavaBeans
Activation Framework (JAF)

• JavaMail is an API that allows application components to send Internet electronic mail
• J2EE specification requires that the API as well as a service provider be available
• JAF is a framework used by JavaMail to associate mime types to Java classes that conform to the JavaBeans specification
Interoperability

- Many of the APIs described provide interoperability with components that are not part of the J2EE platform
J2EE Application Components

- Web Container
  - Java™ Servlets
  - Java™ Server Pages (JSP)

- EJB Container
  - Enterprise Java Beans (EJB)
Java™ Servlets

• Servlets are Java classes that provide a protocol-based request/response service

• HTTP Servlets are the most common type and are typically used to dispatch requests
Java™ Server Pages (JSP)

- JSPs are text based documents that create web pages based on client requests. They intermingle Java with HTML.
- JSPs:
  - Provide for the separation of presentation from logic.
  - Allow for the separation of development roles.
Enterprise Java Beans (EJB)

- EJBs provide interfaces and architecture for building distributed component-based applications
- EJBs provide distributable and deployable business services to clients
- EJBs:
  - are self-contained
  - are reusable
  - are business objects
  - have varying levels of state
  - have well defined interfaces
Enterprise Java Beans (EJB)

- **Session EJBs**
  - Stateful
  - Stateless

- **Entity EJBs**
  - Container-Managed Persistence
  - Bean-Managed Persistence

- **Message-Driven EJBs (EJB2.0)**
  - Asynchronous stateless
J2EE Platform Roles

• **J2EE Product Provider**
  – Implements product providing component containers, platform APIs, other features

• **Application Component Provider**
  – Produces J2EE components: EJBs, Servlets, JSPs, HTML documents

• **Application Assembler**
  – Assembles a set of components into a complete J2EE application
J2EE Platform Roles

- **Deployer**
  - Responsible for deploying web applications into particular operational environment

- **System Administrator**
  - Responsible for configuration and administration of enterprise’s computing and network infrastructure

- **Tool Provider**
  - Provides tools used for development and packaging of application components
J2EE Architecture

Client-Side Presentation

- Browser
  - HTML
  - Java Applet
- Desktop
  - Java App
- Other Device
  - J2EE Client

Enterprise Information System
J2EE Architecture

Client-Side Presentation

Browser
- HTML
- Java Applet

Desktop
- Java App

Other Device
- J2EE Client

Enterprise Information System

Communication:
- HTTP
- RMI

Firewall
J2EE Architecture

Client-Side Presentation
- Browser
  - HTML
  - Java Applet

Desktop
- Java App

Other Device
- J2EE Client

Server-Side Presentation
- Web Server
  - JSP
  - JSP

Java Servlet

J2EE Platform

HTTP

RMI

Firewall

Enterprise Information System

Java™ 2 Enterprise Edition Seminar
J2EE Architecture

Client-Side Presentation
- Browser
  - HTML
  - Java Applet
- Desktop
  - Java App
- Other Device
  - J2EE Client

Server-Side Presentation
- Web Server
  - JSP
- Java Servlet
- J2EE Platform

Enterprise Information System

- Firewall
- HTTP
- RMI

Java™ 2 Enterprise Edition Seminar
J2EE Architecture

Client-Side Presentation
- Browser
  - HTML
  - Java Applet
- Desktop
  - Java App
- Other Device
  - J2EE Client

Server-Side Presentation
- Web Server
  - JSP
- Java Servlet
  - J2EE Platform

Server-Side Business Logic
- EJB Container
  - EJB
- EJB
- J2EE Platform

Enterprise Information System

Firewall

HTTP

RMI

Java™ 2 Enterprise Edition Seminar
Java™ 2 Enterprise Edition Seminar

J2EE Architecture

Client-Side Presentation
- Browser
  - HTML
  - Java Applet
- Desktop
  - Java App
- Other Device
  - J2EE Client

Server-Side Presentation
- Web Server
  - JSP
- Java Servlet
  - EJB Container

Server-Side Business Logic
- EJB
- J2EE Platform

Server-Side Data Access Logic
- EJB
- J2EE Platform

Enterprise Information System
- SQL
- RMI

Presentation
- Client-Side
- Server-Side

Data Access Logic
- Server-Side

Firewall
- HTTP
- RMI

Java™ 2 Enterprise Edition Seminar

Capstone
At the Center of Middleware
New Initiatives

• **Web Services**
  – SOAP, WSDL, UDDI, ebXML, JAX API’s
  – Interoperability

• **Java™ Connector Architecture (JCA)**
  – Standardization of legacy system adapters

• **Enterprise Java Beans 2.0 (EJB2.0)**
  – Message-Driven EJB introduced
  – Entity EJB Relationships
  – EJB Query Language
Intermission

Duration: 10 minutes
J2EE Benefits & Business Cases
Reasons to Use J2EE

• **Time to Market is much faster**
  – No need for infrastructure development
  – Developers focus on business domain logic
Reasons to Use J2EE

• **Ease of Integration**
  – APIs provide for high interoperability with entities outside of J2EE
    • RMI-IIOP
    • JRMP
    • HTTP / SSL
  – Web Services (SOAP, ebXML, WSDL, UDDI) also complement J2EE providing another means of interoperability
Reasons to Use J2EE

• **J2EE is Standards-Based**
  – “Train-Once, Develop Anywhere”
  – Increases pool of potential employees
  – Reduces the need for “home-grown” systems
Reasons to Use J2EE

- **Component Reusability**
  - EJBs are self-contained and only comprise core business domain logic making them deployable across J2EE compliant application servers
Software Process & J2EE

A Case Study
Project Example – Software Process Focus

• Requirements
  – Workflow automation
  – Configuration of devices
  – Information stored in a directory
  – Highly available/failover
  – Software must be of high quality and have a quick time to market

• Artifacts available:
  – Business vision
  – Architecture ideas

• Team
  – Domain savvy
  – Limited large software project experience

• Software engineering team – estimated 5 to 20 staff
Software Process Startup

• Management agreed to adopt best practice
• Requirements in different forms and different levels of specification
• Team not familiar with J2EE
• Initial team split:
  – Demonstration needed to address two risks
    • Would the technology work?
    • Exposure to potential clients
  – Requirements gathering
    • Uses cases were developed
    • Software and domain experts working together
Early Architecture Decisions

• Needs:
  – Component based
  – API eventually needed
  – Event engine
  – Highly available
  – Short development cycle

• Answer:

  J2EE

  Rational Unified Process
Project X Timelines

• Elaboration iteration – requirements and proof of concept
  – 3 calendar months
  – J2EE Architecture selected
• Construction Iteration 1 3.5 calendar months
• Construction Iteration 2 – Beta – 3 calendar months
• Construction Iteration 3 – First ship – 4 calendar months
Tools Selected

- OOAD Tool: Rational Rose
- Software Process: Rational’s RUP
- J2EE: Orion/SilverStream/WLS
- IDE: Forte/Visual Café
- Unit testing: Junit
- Configuration Management: Perforce
- Defect Tracking: TeamTrack
- Project planning: MS Project
How J2EE Affects Software Process

- Vision document
  - Architecture chosen and constraints listed
- Software Development Plan
  - Training
  - Previously architecture selected much later
  - Can be started early because architecture is selected
- Risk driven
  - Still key
  - Depending on staff, J2EE may be a risk
- Prototypes
  - In J2EE environment
- Use cases not affected
How J2EE Affects Software Process

• Proceed with J2EE up front, not at the end
• Refine the architecture and select components
  – Will you purchase J2EE components?
  – Which Java libraries will you use?
• Prototypes should include J2EE unless:
  – The risks are not in the architecture (rare) AND
  – Team seasoned on J2EE tool that was selected
• Executable architecture should have J2EE
How J2EE Affects Software Process

• Tools selected should integrate with selected J2EE
  – For now, define the boundaries of the tools
    • OOAD modeling tool will do …
    • IDE will do …
    • J2EE tools will do …
  – In the future the tools will be one and its name will be …

• Implementation plan
  – How many tiers?
  – Where will J2EE components be placed in packages/subsystems?
How J2EE Affects Construction

• J2EE development environment must be provided for all developers

• Deployment plan is important
  – Deployment is a specific step
  – How will configuration management be done on:
    • deployment descriptors
    • WARs, EARs, Manifests, etc.

• Implement patterns and idioms early
  – Implement vertically at least once even if risks are more horizontal

• Remember the importance of database design/implementation
How J2EE Affects Transition

- Administrative type work with J2EE tools
- Easier to vary deployment
  - Are many applications deployed in one application server, utilizing common resources?
  - Is an application deployed across multiple application/web servers (separating presentation from business logic)?
- If architecture is sound the scalability issues are addressed by J2EE
- Potential of switching of J2EE vendor due to price and scalability
- Keep the “soul of the system” intact
- Tuning for performance more straight forward when adhering to Application Server guidelines
- Hot deployment of components makes production migration easier
Software Process Summary

• J2EE affects the software process
• More architectural risks addressed early on
• Prototypes easier to implement
• Easily integrated into current software processes
Financial Services Engine

A Case Study
Financial Services Engine

- The building block for financial transactions
Financial Services Engine

- Integrates with existing financial systems
Financial Services Engine

• Resource pooling provides scalability and redundancy
Financial Services Engine

- Reusable components and objects model the domain
Financial Services Engine

- The engine separates the business logic from the presentation logic allowing for greater reusability.
Financial Services Engine

- As presentation requirements evolve, additional components can be easily integrated.
Financial Services Engine

• Reuse occurs on the presentation side as well with common presentation components
Financial Services Engine

- Integration with other products is simplified by reusing existing components
Financial Services Engine

- Integration with legacy systems also becomes simplified
FSE Summary

• Separates business logic from presentation logic
• Represents your domain
• Promotes reusability
• Integration with other systems is simplified
Capstone Consulting

Overview, Who, Why, Next Steps and Quiz Show
A Foundation for Growth

Software life cycle costs are large, especially during the maintenance cycle of a project.

New technologies can help reduce maintenance costs and promote software reuse.

Implementing commercial software tools can increase scalability, reliability and maintainability.

Finding expertise within your company may be difficult or impossible.

That's where Capstone can help.
Capstone Can Help

• Understand emerging technologies
  – Help you implement them successfully for increased business value
• We work side by side with your staff to enable them to assimilate technologies
• Capstone secures the gap and works to train your own staff to fill the technology chasm. Help you understand the benefits and pitfalls of emerging technology
• Capstone helps in training and mentoring, architecture, requirements gathering, project design and development consulting
Capstone Who

- Software Architects/Designers
- J2EE Experts in the Midwest, with a national presence
- Focused on emerging technologies
- Not all things to all people
- Integrity
Capstone Why

• Local, but national experience
• Training and mentoring
• Work along side your team
• Laser focused
• Personnel are broad and deep
Capstone Next Steps

• On-site evaluations
  – Half-day evaluation
  – Review your environment and make suggestions

• Classes

• Mentoring

• Web Services Seminar, Oct. 18
Capstone Quiz – For Prizes

- What is the platform of choice when implementing a java-based, platform-independent distributed object system that is highly available, maintainable and scalable?

- Name the top three J2EE Vendors?

- Who is the primary driving force behind the J2EE Specification?

- What Omaha company specializes in OO Technologies, Java Middleware Software Solutions, and Internationally-known Java Middleware Training?
Thank-You