OpenESB: Connecting the Enterprise

Sang Shin
Java Technology Architect
Sun Microsystems, inc.
Agenda

- JBI
- OpenESB
- NetBeans Support of JBI & OpenESB
- JBI/OpenESB and GlassFish
- Java EE Service Engine
- Intelligent Event Processor (IEP)
- Enterprise Data Mashup (EDM)
JBI (Java Business Integration)
What Is JBI?

- JBI provides a standard application integration framework
- JBI to Application integration is what Java EE is to Enterprise application development and deployment
Why JBI?

• Point-to-point integration model is not scalable and hard to maintain
• The traditional EAI model has its problems
  > Proprietary integration server
  > Vendor lock-in
• There is a need for an open standard framework for application integration
What Is JBI?

- Standard “meta-container” for integrating “service containers”
  - Service containers host service units
  - Service can be located locally or remotely
- Plug-in architecture
  - 3rd-party service containers can be plugged in
- Message driven
JBI Key Concepts

• Service Engines (SE)
  > Service container hosting business logic or system service service unit

• Binding Components (BC)
  > Pluggable remote connectivity – they are service containers too whose services are remotely located

• Normalized message router (NMR)
  > High efficient in-memory message router
JBI Service Engines (SE) & Binding Components (BC)

- **Service Engines**
  - BPEL SE
  - Java EE SE
  - IEP SE
  - EDM SE
  - SQL SE
  - XSLT SE
  - WLM SE
  - ETL SE

- **Binding Components**
  - HTTP BC
  - SMTP BC
  - MQSeries BC
  - HL7 BC
  - SAP BC
  - JMS BC
  - File BC
  - CICS BC
  - DCOM BC
  - CORBA BC
  - ...

open-esb.dev.java.net/Components.html
Scenario 1: Remote through HTTP BC

- BPEL
- NMR
- HTTP BC
- *Marshall DOM to <xml>*
- SOAP/HTTP
- Network layer

Network layer
- SOAP/HTTP
- JAXWS
- *Unmarshal <xml>*
- Create DOM
- JAXB
- WS.helloWorld(name)
Scenario 2: Local through NMR

- BPEL ➔ NMR ➔ HTTP BC ➔ Marshall DOM to <xml> ➔ SOAP/HTTP ➔ Network layer ➔ SOAP/HTTP ➔ JAXWS ➔ Unmarshal <xml> ➔ Create DOM ➔ JAXB ➔ WS.helloWorld(name)

Advantages:
- Performance
- Transaction propagation
- Security context propagation
What is Project OpenESB?

• Project OpenESB implements an Enterprise Service Bus (ESB) runtime using Java Business Integration (JBI) as the foundation

• It also provides various tools for the development, deployment, and management of composite applications
OpenESB Architecture
NetBeans Support of JBI/OpenESB
Types of SOA “NetBeans” Projects

• When creating a composite application, you typically use the following types of SOA “NetBeans” projects:
  > BPEL Module project
  > IEP (Intelligent Event Processor) Module project
  > EDM (Enterprise Data Mashup) Module project
  > XSLT Module project
  > SQL Module project
  > Composite Application project
  > And more (WLM, ETL, etc.)
Types of SOA “NetBeans” Projects

Choose Project

Categories:
- Java
- Java Web
- Java EE
- Java ME
- Groovy
- NetBeans Modules
- SOA
- Samples

Projects:
- Composite Application
- BPEL Module
- XSLT Module
- Data Mashup Module
- Intelligent Event Processing Module

Description:
Creates an empty Composite Application project, which may include multiple BPEL Modules and other types of Java Business Integration (JBI) modules.

Note: For more information, press the Help button and see the on-line help section on About Composite Application Projects.
What is a Composite Application?

- Contains a service assembly
  - Each service is captured in the form of a JBI module (BPEL, XSLT, IEP, EDM, etc.)
- A service assembly is deployment unit
  - Individual JBI module cannot be deployed by itself
  - Deployed over JBI server
- Test cases can be created
- Service assembly can be configured through CASA
  - Service access protocol (SOAP 1.1, SOAP 1.2, JMS, SMTP, and others) can be configured for each service
CASA Editor

- Composite Application Service Assembly (CASA)
- Graphical tool for configuring service assembly
BPEL Module Project

- BPEL Module project is a group of source files
  - BPEL file
  - WSDL file
  - XML Schema (*.xsd) files

- Will be added to a Composite application as a JBI module
Demo:
Building “Hello World” Composite Application

www.javapassion.com/handsonlabs/wscompositeapps/
“Hello World” Composite Application

• A very simple composite application
  > Receives a simple “Hello World” message from its client and echos it back to the same client

• A simple XML schema
  > Same XML schema for describing the XML document syntax of both request and response messages

• A simple WSDL document (reflecting the BPEL process to its client)
  > A simple synchronous request/response

• A simple BPEL process
  > Receive->Assign->Reply activities
Steps for Building a Composite App.

1. Create a “BPEL Module” NetBeans project
2. Create XML Schema (as part of BPEL module)
3. Create WSDL document (as part of BPEL module)
4. Create BPEL process (as part of BPEL module)
5. Create a “Composite Application” NetBeans project
   ➢ Add BPEL module
6. Deploy the Composite application
7. Test the Composite application
8. Debug the Composite application
Java EE SE
JavaEE SE

• Ideal place to execute complex business logic
• Bridge between JavaEE container and JBI container
• Provides support for
  > Transactions
  > Resource Pooling
  > Security
• Ability to expose your EJB/Web applications to multiple transports (using BCs) – just add bindings to your WSDL
Use Case Scenario
Usage Scenario: Loan Processing

- Loan Requestor Service:
  > LoanRequestProcess
    - WS-I BP
    - BPEL Orchestration
  > LoanProcessor
    - JavaEE
  > TransformReport
    - XSLT
  > LoanReportStore
    - Business Partner thru FTP
  > LoanReportMailer
    - Legacy thru JMS
JBI-based Infrastructure

BPEL Loan Request Process

JavaEE Loan Processor EJB

XSLT Transform Report

NMR

WS-I BP LoanRS WS

JMS ReportMail

File ReportStore

JBI-based Infrastructure
NMR Loan Request Process

BPEL

JavaEE Loan Processor EJB

XSLT Transform Report

WS-I BP LoanRS WS

JMS ReportMail

File ReportStore

File

ReportStore

ReportMail

LoanRS WS

BPEL Loan Request Process

JavaEE Loan Processor EJB

XSLT Transform Report

WS-I BP

Sun Microsystems
Architecture Refactoring

BPEL Loan Request Service

XSLT Transform Report

JavaEE Loan Processor EJB

NMR

WS-I BP LoanRS WS

JMS ReportMail

File ReportStore

[Diagram depicting various components and services related to architecture refactoring and enterprise integration]
Demo: Building Loan Composite Application

www.javapassion.com/handsonlabs/wscompositeapps/#Exercise_3
www.javapassion.com/handsonlabs/wscompositeapps/#Exercise_4
Demo Scenario: “Loan Processing”

- EJB Web service provides business logic of “approving” or “denying” loan applicant
  - Age should be over 18 years old
  - Yearly income has to be over $20,000
- BPEL process invokes EJB Web service
Intelligent Event Processor (IEP)
What is IEP?

• Models event-driven business process
  > Pretty much all business applications are event-driven
• IEP applications continuously perform
  > Collate and process events from various sources
  > Detect pre-defined patterns & conditions
  > Trigger another event or invoke other internal or external services
Business Service Monitoring

- Monitor business services through pattern matching and condition check
  - Track events by user - “User Behavior”
  - Detect spikes in service usage - “System Behavior”
- Automatic response
  - Alerts, Dashboard updates, service triggers, emails, etc.
Real-life Use Cases of IEP

- How many times did Fred login as root for the last 24 hours?
- How many times is “purchase order” business process triggered for the last hour? And what is the average execution time?
- Is a credit card charged for gasoline twice within last one hour?
- Raise an alert when a stock price jump more than 10% relative to its 1 minute moving average price.
- Is the number of JMS messages in the broker increasing over time? What changed?
Demo: Insider Trading Detection

www.javapassion.com/handsonlabs/openesb_iep/#Exercise_2
Demo Scenario: “Insider Trading Detection”

• Monitor a stream of stock transactions, and detects potential insider trades

• Detect "suspicious" stock trades, that is, transactions whose price stands out relative to the stream
  > Compute 24-hour moving average of a stock price
  > If the stock price is 10% above or below the average, place it into the suspicious stock trade list

• Check parties to suspicious transactions against a table of previously identified persons-of-interest
Enterprise Data Mashup (EDM)
What is EDM?

• Provides a unified view of data from heterogeneous sources

• The sources of data can come from
  > Relational databases, flat files, spreadsheets, Web services, RSS/Atom, XQuery RowSet

• Can perform
  > Join'ing/aggregation, cleansing

• Can use external services
  > XLST
Demo: Enterprise Data Mashup

www.javapassion.com/handsonlabs/openesb_edm/#Exercise_1
Demo Scenario: “Join'ing tables from multiple databases”

• Join data from multiple databases from a single database server (or multiple database servers)
• Modify Join conditions
Summary & Resources
Summary

• SOA enables flexible and agile enterprise application architecture

• Services can be created and used using Java EE

• BPEL is a service orchestration language for creating composite applications

• Java Business Integration (JBI) and OpenESB are the enabling infrastructure technologies for SOA
Resources

http://www.javapassion.com/soaprogramming

https://open-esb.dev.java.net/
THANK YOU!

Sang Shin
sang.shin@sun.com