Oracle Developer Day

Track #1 - Session #4

BPEL Introduction

Presenter Name
Presenter Title
Agenda

- Orchestration
- What is BPEL?
- BPEL Programming Language
- Steps to Build a Business Process
- Oracle BPEL Process Manager

Orchestration Requirements

Connectivity
- Heterogenous Back Ends
- Sites of API and mechanisms
- Opaque/heterogeneous data definitions
- Synchronizing multiple data stores

Flow Control
- Asynchrony, Flow Coordination, Data Transformation, Compensation, Version Control, Auditing

Scalability
- Unpredictable loads
- Asymmetric performance capabilities

Management and Security
- Access control, Encryption, Logging, Metering
- Independent of the service

Interaction/Access
- Catalog, Customization, Access
Orchestrate Services

What is BPEL?

Markup language for composing a set of discrete services into an end-to-end process flow

- 10+ years of R&D from MSFT and IBM
- SOAP but also Java, JCA
- Rich Flow Semantics
- Optimized Bindings
- XPATH+XSLT+XQuery
- WS-Security
- A Process is a Service
BPEL History

Proprietary → Standard

- BPML (Letaltio et al) 2000/05
- XLang (Microsoft) 2001/03
- WSFL (IBM) 2001/05
- BPSS (eXtensible XML) 2001/06
- WSCL (HP) 2002/03
- BPEL4WS 1.0 (IBM, Microsoft) 2002/06
- BPEL4WS 1.1 (OASIS) 2003/04
- WSCI (Sun et al) 2005/2006
- WS-BPEL 2.0 (OASIS) 2005/2006

BPEL – Programming Language

- Variables
- Activities
- Control Flow
- Scope
- Fault Handling
- Event Handling
BPEL - Variables

- Used to store, reformat and transform messages
- Required to send and receive messages
- Each variable has a Type

Example:

```xml
<variables>
  <variable name="loanApplication"
    messageType="ns2:LoanServiceRequestMessage"/>
</variables>
```

BPEL - Activities

- **Primitive Activities**
  - `<invoke>`
  - `<receive>`
  - `<assign>`
  - `<reply>`
  - `<throw>`
  - `<terminate>`
  - `<wait>`

- **Structured Activities**
  - `<sequence>`
  - `<switch>`
  - `<pick>`
  - `<flow>`
  - `<link>`
  - `<while>`
  - `<scope>`
BPEL - Activities

- `<invoke>`
  - Invoke a service synchronously
  - Ex: Invoke Credit Service

- `<receive>`
  - Waits for the incoming message, either to start the process or for a callback
  - Ex: Wait for a message from United Loan

- `<reply>`
  - Return response for synchronous process, relate to initial `<receive>`

- `<assign>`
  - Copy data between variables, expressions and endpoint references
  - Used with XPath expressions and XSLT engine
  - Ex: Copy Loan Application from input payload to United Loan input

BPEL – Scope

- Scopes can be used to divide the business process into organized parts
- A `<scope>` is an execution context for the contained activities, and a process is, itself, a `<scope>`
- A `<scope>` defines local variables and can catch and handle either specific faults or all faults that occur with it
  - Ex: GetCreditRating Scope – Invoke Credit Service and catch exceptions
BPEL – Control Flow

- BPEL provides the usual branching and looping control flow constructs
- A `<sequence>` executes activities in serial order
- A `<switch>` executes at most one alternative based on expressions specified on child `<case>` elements with an optional `<otherwise>`
  Ex: Choose between United and Star loan offers based on lower APR
- A `<while>` loops through activities while a variable’s value is true
- BPEL provides a parallel control construct through the `<flow>` activity
  Ex: Invoke United and Star Loan services in parallel

Partner Links

- Links to all parties that process interacts
- Links can be to Web Services
  Ex: CreditService, UnitedLoanService, StarLoanService
- Links can be to other BPEL processes as well
- PartnerLinkTypes
  - Declares how parties interact and what each party offers
BPEL - Fault Handling

• Handle faults to enable completion of process using `<faultHandlers>`
• Use `<catch>` activity to handle specific faults
  Ex: Catch bad credit exception and terminate the process
• Use `<catchAll>` to handle all other faults

Steps to Build Business Process

1. Define Public Interface
2. Create Partner Dictionary
3. Create Message and Type Dictionary
4. Implement Transformation Logic
5. Implement Orchestration Logic
6. Create a Test Environment
7. Iterate
8. Live Pilot
9. Fine-tune Operations Tasks
Step 1: Define Public Interface

Deliverables:
- WSDL description of the interface of the implemented BPEL process

Step 2: Create Partner Dictionary

Deliverables:
- List of the WSDL of the services that will be invoked as part of the BPEL Process
- For each partner, document the order in which operations will be invoked (choreography)
- Make sure that each use case describes both positive and negative use cases
Step 3: Create Message and Type Dictionary

Deliverables:
- A set of XML Schema files that describe the type of the messages and XML documents used as part of the BPEL process.

Step 4: Transformation Logic

Deliverables:
- A set of XSLT and XQuery files that encapsulate mapping information across the various types used in the BPEL process.
Step 5: Orchestration Logic

Deliverables:
• Implement the workflow that ties the interactions across partners into an end-to-end business process.
• Make sure that all exceptions and timeouts are managed properly.

Step 6: Iterate

Deliverables:
• Add incrementally new partners
• Keep on improving exception management
• Create automated test and regression framework
Step 7: Create test environment

Deliverables:
- Implement dummy test services for each end point (could be BPEL or your favorite Web services publishing technology)
- Create test scenario for each positive and negative use cases
- Crash test, longevity test (integrity/reliability)
- Performance test, stress test

Step 8: Live Pilot

Deliverables:
- Wire BPEL process to real end points
- Run regression tests
Step 9: Fine-tune Operation Tasks

Deliverables:
• Exception Management
• Integration with Web Service Management Framework
• Security
• Archiving

Oracle BPEL Process Manager

Enterprise-strength infrastructure for designing, deploying and managing BPEL business processes

Oracle BPEL Process Manager

- BPM
- Oracle BPEL Process Manager

BPEL Designer
- XSLT Mappers
- Adapter Factory

BPEL Engine
- Workflow
- Rule

Application Server (Oracle, BEA, JBoss, IBM)

BPEL Console
- Dehydration Store (Oracle Database)
- WSIF Binding
- Queuing

Oracle BPEL Process Manager

- SAP
- EBS
- DB
- File...

- Comprehensive and native BPEL implementation
- Easy-to-use modeling tool
- Scalable and reliable engine
- Flexible binding framework
- Rich management and monitoring
- Get up and running in less than 15 minutes!
Cross Platform

Application Server
- Oracle Application Server
- WebLogic Server
- WebSphere
- JBoss

Database
- Oracle Database
- SQL Server
- Oracle Lite
- Sybase
- Pointbase

IDE
- JDeveloper
- Eclipse

Operating Systems
- Linux
- Window XP/2003
- Solaris
- HP UX
- zOS

BPEL Console

KEY FEATURES
- Visual Monitoring
- Auditing
- BPEL Debugging
- In-flight Instance Administration
- Performance Tuning
- Partitioning/Domains
DEMONSTRATION

Join Over 4,500,000 Developers!
Free Technical Advice
Free Software Downloads
www.oracle.com/technology/products/jdev
otn.oracle.com/tech/webservices
otn.oracle.com/tech/java
otn.oracle.com/bpel
Learn Oracle From Oracle

Oracle University

- Instructor led training
- Self-Study
- Online learning
- Oracle Certification
- Oracle iLearning
- Oracle Tutor

oracle.com/education

QUESTIONS & ANSWERS