Closer Look at Enterprise Service Bus

Deb L. Ayers
Sr. Principle Product Manager
Oracle Service Bus
SOA Fusion Middleware Division
The Role of the Foundation
Addressing the Challenges

- Efficiency
  - Utilization
  - Operations

- Competitiveness
  - Performance
  - Reliability
  - Flexibility

- Simplification
  - Consolidation
  - Standardization
  - Automation
Oracle Fusion Middleware Technologies

User Interaction
- Web 2.0 Portal, Rich Internet Apps, Mobile, Search, Desktop, Presence, VoIP

Enterprise Performance Management
- Planning, Budgeting, Financial Management & Reporting, Scorecards

Business Intelligence
- Data Integration, Query & Analysis, OLAP, Dashboards, Reports, Alerts, Real-Time

Content Management
- Web Content, Document, Records Mgmt, DAM, Capture and Imaging, Archiving, IRM

SOA & Process Management
- ESB, BPEL, Workflow, BAM, Rules, B2B, MDM, Data Integration, Governance, Event Processing

Application Grid
- Java EE Application Server, JVM, TP Monitor, In-Memory Data Grid, Application Cluster

Development Tools
- Unified SOA Development Tool & Framework

Enterprise Management
- SOA Management, Provisioning, Diagnostics, Configuration Management, Tuning

Identity Management
- Provisioning, Access Management, Audit, Directory, Role Management, Fraud Detection
Agenda

- Why Service Bus?
- Enterprise Patterns
- Closer Look
  - Sync to Async Bridge
  - Service Result Cache
  - EJB Proxy
- Q & A
Agenda

• Why Service Bus?
• Enterprise Patterns
• Closer Look
  – Sync to Async Bridge
  – Service Result Cache
  – EJB Proxy
• Q & A
Service Integration: Developer Perspective

- Application
- Remote Functionality
Brittle Over Time
With or Without Web Services
Why SOA Initiatives Fail: Technology or Governance?

Risk of SOA Project Failures

Technology Risk

Lack of Governance Risk

Introduction  Spreading  Exploitation  Plateau

More Risk

Less Risk

Time
Service Bus
PO Processing as an example

Oracle Service Bus
Coherence
Web-based console
virtualization
transport switching
pooling

caching

Oracle Restricted and Confidential
Agenda

• Why Service Bus?
• Enterprise Patterns
• Closer Look
  – Service Result Cache
  – EJB Proxy
• Q & A
Patterns for Service Bus

Customer Use Cases:
- Backend Application Integration
- Portal application consumes business services from back-end applications.
- Eliminate the brittleness of services using a SOA based architecture.

 Requires ESB Characteristics:
- Heterogeneous messaging backbone
- Content based routing
- Service enrichment
- Monitoring and reporting
- SOA based security
- Service workload and management
- Message Guarantees
- Service orchestration
- Distributed services across the enterprise
- Service discovery
Common Service Types

- **Traditional Web Services**
  - Pre-negotiated Interfaces Contract (WSDL)
  - Standards in place, supported by many vendors
  - SOAP over HTTP

- **Legacy Services**
  - Non-XML (XML) over File, FTP, MQ, JMS

- **POX (Plain Old XML)**
  - Structure of Payload to determines action
  - XML over HTTP

- **REST (Representational State Transfer)**
  - Based on Pattern of Service Invocation
  - Nouns vs. Verbs
  - URIs over HTTP
Proxy & Business Services
Adaptive Messaging In a Nutshell…

- Any to Any Protocol
- Any to Any Payload
  - XML
  - non-XML
  - Binary
- No WSDL Required

- Multiple communications paradigms
  - Request/response
  - Synchronous and asynchronous
  - One-to-many, many-to-one
  - Pub-sub
  - Mix-and-match (e.g. sync-to-async)
Let’s Build It…

- Create a simple service with a Web services interface that performs data translation and protocol conversion to invoke an existing service through an asynchronous message, wait for a response, and return a synchronous reply to the caller.
## Summary of ESB Usage Patterns

<table>
<thead>
<tr>
<th>Typical Use Cases</th>
<th>Customer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content, Metadata, Identity, -Based Routing</td>
<td>✓</td>
</tr>
<tr>
<td>Asynchronous Request/Reply</td>
<td>✓</td>
</tr>
<tr>
<td>Sync client to Async Business Service</td>
<td>✓</td>
</tr>
<tr>
<td>Security and Access Control</td>
<td>✓</td>
</tr>
<tr>
<td>Data services integration</td>
<td>✓</td>
</tr>
<tr>
<td>Customized Data Reporting</td>
<td>✓</td>
</tr>
<tr>
<td>Monitoring / SLA</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Heterogeneous Messaging</strong></td>
<td>✓</td>
</tr>
<tr>
<td>Error Handling in Pipeline</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Business Service Load-balancing &amp; Failover</strong></td>
<td>✓</td>
</tr>
<tr>
<td>Transport Header Override</td>
<td>✓</td>
</tr>
<tr>
<td>Service Lifecycle Management</td>
<td>✓</td>
</tr>
</tbody>
</table>
Agenda

• Why Service Bus?
• Enterprise Patterns
• Closer Look
  – Service Result Cache
  – EJB Proxy
• Q & A
**Message Flow Transaction**

- **Atomic Message Flow**
  - Simple as checking a box
  - All work committed or aborted

- **Two settings for Proxy**
  - Transaction Required ?
  - Same Transaction For Response ?

**Benefits**
Message flow can execute within TX regardless of the Inbound Protocol
Native EJB Inbound and Outbound Transport

- New native EJB Transport
  - Supports 2.1 and 3.0
  - Native Java Types can be passed along to JMS, Java Callout
  - Full Transaction semantics supported

Benefits
  - Mediate existing legacy EJB infrastructure
Service Result Cache
Scaling Application Infrastructure

If results not in cache, invoke service & cache results

Side Cache Pattern

Coherence Data Grid
Get / Put (Cachekey)

Primary Node

OSB 11gR1
Service Result Cache with Coherence

Checkbox Caching

Business Service

✓ Cache Results?
✓ Cache Token
✓ Time To Live

Result Cache is an implementation of Side Cache pattern
Service Bus vs. Process Orchestration

**Process Orchestration**
- Service contains business logic
- Service requires complex transaction management
  - Requires multiple transactions
  - Compensation logic required on rollback
  - Short or long-lived process
- Exception handling requires Human workflow
- Service needs to handle asynchronous callbacks reliably

**Service Bus**
- Service contains protocol, routing and transformation logic
- Service has short-lived, single transaction semantics
Final Thoughts…

Trends Emerging

• Federated ESB
  • Not enough to simply distribute across buses.
  • Management is KEY.
  • Homogenous until vendors decide on management standards to allow uniform service provisioning & routing updates.

• Event-driven SOA
  • Dynamic, unpredictable business events being correlated real-time feeding into SOA infrastructure

• Web 2.0 meets SOA & Traditional IT infrastructure
  • Watch out - IT Culture class emerging!
  • Service Bus will help adapt Traditional SOA to more ad-hoc services like REST, POX with security.
Discussion
If a URI is non-responsive, take the URI out of the pool
Bring the URI back in the pool when it is back-up

- Option for the system to automatically take non-responsive URIs out of the pool, and put them back in as they become responsive
- Alerts will be generated when the status of the endpoint changes from Up → Down and vice versa