Introducing SwitchYard

- New JBoss community project
- Next generation Enterprise Service Bus
- What happened to JBoss ESB?
- Taking the next evolutionary step
  - Focus on consistent, intuitive user experience
  - Refactor core to eliminate known pain points
  - Leverage standards and complimentary technologies
Service-Oriented Architecture
Service-Oriented Architecture

Exhibit #1: Not Helping
Service-Oriented Architecture
Service-Oriented Architecture
Service-Oriented Architecture

POJO (CDI)

Workflow (jBPM 5)

Business Rule (Drools)

Routing (Camel)

Orchestration (BPEL)

Implementation Logic

Application

Tuesday, June 26, 12
Bean Services

- POJO = Service ... ‘nuff said
- Easy to use
  - Annotation-driven
  - Config auto-generated
  - Service auto-registered
- Based on CDI
  - Standard programming model (Java EE / JSR 299)
  - Straightforward integration into the web tier
Bean Service
public interface OrderService {
    OrderAck submitOrder(Order order);
}
public interface OrderService {
    OrderAck submitOrder(Order order);
}

public class OrderServiceBean implements OrderService {
    public OrderAck submitOrder(Order order) {
        ...
    }
}
Bean Service

```java
public interface OrderService {
    OrderAck submitOrder(Order order);
}
```

```
@Service(OrderService.class)
public class OrderServiceBean implements OrderService {

    public OrderAck submitOrder(Order order) {
        ...
    }
}
```
@Service(OrderService.class)
public class OrderServiceBean implements OrderService {

    public OrderAck submitOrder(Order order) {
        // Check the inventory
        Item orderItem = inventory.lookupItem(order.getItemId());
        ...
    }
}
@Service(OrderService.class)
public class OrderServiceBean implements OrderService {

    @Inject @Reference
    private InventoryService inventory;

    public OrderAck submitOrder(Order order) {
        // Check the inventory
        Item orderItem = inventory.lookupItem(order.getItemId());
        ...
    }
}
<div id="content">
  <h1>New Order</h1>
  <h:form id="newOrder">
    <div>
      Order ID:
      <h:inputText id="orderID" value="#{order.orderId}" required="true"/>
      <br/>
      Item ID:
      <h:inputText id="itemID" value="#{order.itemId}" required="true"/>
      <br/>
      Quantity:
      <h:inputText id="quantity" value="#{order.quantity}" required="true"/>
      <br/>
      <h:commandButton id="createOrder" value="Create" action="#{order.create}"/>
    </div>
  </h:form>
</div>
@Named
@RequestScoped
public class Order implements Serializable {

@Inject
@Reference
private OrderService orderService;

public void create() {
    OrderAck serviceAck = orderService.submitOrder(this);
    FacesContext.getCurrentInstance().addMessage(null,
        new FacesMessage(serviceAck.toString()));
}
...
JSF + CDI + SwitchYard

New Order

Order ID: 
Item ID: 
Quantity: 1

Create

Service also available over SOAP. Try with soapUI using the Service WSDL.
Routing

- Message
  - Message Translator
  - Content Based Router
  - Content Filter
  - Aggregator
  - Messaging Gateway
  - Message Store
Routing Services

- Integrates Apache Camel as a service
- Camel provides
  - Routing engine and language(s)
  - Loads of EIP
- Camel as a service
  - Routes provide pipeline orchestration
  - Service interface
  - Binding abstraction
public class OrderServiceBuilder extends RouteBuilder {

    public void configure() {
        from("switchyard://OrderService")
            .log("Order Received : ${body}")
            .to("bean:prioritize")
            .filter().xpath("/order[@priority='high']")
            .to("switchyard://ShippingService");
    }
}
Beans In Camel

- Allows Java objects to be called inside a route
- Very useful for fine-grained integration tasks
  - EIP configuration - route, split, etc.
  - Metadata access
  - Bolt-on logic
- Bean registry is pluggable
import javax.enterprise.context.ApplicationScoped;
import javax.inject.Named;

@Named("MyBean")
@ApplicationScoped
public class SomeBean {

    public String doSomething(String content) {
        return "I did something with " + content;
    }
}

import org.apache.camel.builder.RouteBuilder;

public class CamelServiceRoute extends RouteBuilder {

    public void configure() {
        from("switchyard://InventoryService")
            .split(body(String.class).tokenize("\n"))
            .filter(body(String.class).startsWith("item:"))
            .bean("bean:MyBean");
    }
}
Workflow
Workflow Services

- Executable flowcharts with BPMN 2 and jBPM 5
- Integrated with BPMN 2 modeler
  - Eclipse and Web
- Expose workflow as a service
- Invoke services as part of a workflow
- Flexible mapping between process and message
Service Orchestration

- Orchestrate web services with BPEL and Riftsaw
- WSDL contracts, native integration
- Multiple bindings
- Latest Riftsaw uses SY as runtime
Application Architecture
Application Architecture
Application Architecture

Java

```java
public interface OrderService {
    OrderAck submitOrder(Order order);
}
```

WSDL

```xml
<portType name="OrderService">
    <operation name="submitOrder">
        <input message="tns:submitOrder"/>
        <output message="tns:submitOrderResponse"/>
    </operation>
</portType>
```

ESB

```xml
<interface.esb
    inputType="json:orders:Order"
    outputType="java:org.example.OrderAck"
    faultType="{urn:example}error"/>
```
Application Model

orders

OrderService
Application Model
<composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912" name="orders" targetNamespace="urn:switchyard-quickstart-demo:orders:0.1.0">
  <service name="OrderService" promote="OrderService">
    <interface.wsdl interface="wsdl/OrderService.wsdl#wsdl.porttype(OrderService)"/>
    <binding.soap xmlns="urn:switchyard-component-soap:config:1.0">
      <wsdl>wsdl/OrderService.wsdl</wsdl>
      <socketAddr>:18001</socketAddr>
      <contextPath>demo-orders</contextPath>
    </binding.soap>
  </service>
  <component name="InventoryService">
    <implementation.bean class="org.switchyard.quickstarts.demos.orders.InventoryServiceBean"/>
    <service name="InventoryService">
      <interface.java interface="org.switchyard.quickstarts.demos.orders.InventoryService"/>
    </service>
  </component>
  <component name="Order">
    <implementation.bean class="org.switchyard.quickstarts.demos.orders.Order"/>
    <reference name="OrderService">
      <interface.java interface="org.switchyard.quickstarts.demos.orders.OrderService"/>
    </reference>
  </component>
  <component name="OrderService">
    <implementation.bean class="org.switchyard.quickstarts.demos.orders.OrderServiceBean"/>
    <service name="OrderService">
      <interface.java interface="org.switchyard.quickstarts.demos.orders.OrderService"/>
    </service>
    <reference name="InventoryService">
      <interface.java interface="org.switchyard.quickstarts.demos.orders.InventoryService"/>
    </reference>
  </component>
</composite>
Application Editor
Application Architecture
Application Architecture
Application Architecture
Gateway Bindings

- Provide connectivity to/from external systems
- Decoupled from service implementation
- Our approach
  - Focus on key gateways for platform
  - Incorporate adapters from other communities
  - Straightforward pluggability for rolling your own
Gateway Bindings

• Out of the Box
  • Today
    • SOAP, JMS, File, FTP/SFTP/FTPS, SQL, Schedule, REST, JCA, UDP, TCP
  • In Progress
    • HTTP, JPA

• Extensibility
  • Camel components
  • Homegrown
    • Pluggable tooling, configuration, and deployment extensions
Application Architecture
Application Architecture

Remote → Data Validation → Data Transform → Policy → Declarative Behavior → Implementation Logic → Remote

- Data Validation
- Data Transform
- Policy
- Declarative Behavior
- Implementation Logic
Transformation

- Ubiquitous challenge in application integration and SOA
- Change in data representation
  - java.io.Reader -> java.lang.String
- Change in data format
  - CSV -> XML
- Change in data itself
  - Enrichment
Where To Transform

• In the provider?

• In the consumer?

• Add a routing service?
Where To Transform

• In the provider?
  NO!

• In the consumer?

• Add a routing service?
Where To Transform

• In the provider? **NO!**

• In the consumer? **NO!**

• Add a routing service?
Where To Transform

• In the provider?
  NO!

• In the consumer?
  NO!

• Add a routing service?
  MAYBE ...
Transformers

- Transformation is wired into SwitchYard core
  - Types declared via service contract
  - Transformer resolved dynamically at runtime
- Declarative, not procedural

- Java, JAXB, XSLT, JSON, and Smooks
@Transformer(from = "{urn:switchyard-example:orders:1.0}submitOrder")
public Order transform(Element from) {
    return new Order()
        .setOrderId(getElementValue(from, "orderId"))
        .setItemId(getElementValue(from, "itemId"))
        .setQuantity(Integer.valueOf(getElementValue(from, "quantity")));
}
Validators

- Declarative validation
- Supports XML Schema and Java validation
- Executes pre and post transformation

```xml
<validate.xml
  schemaType="XMLSCHEMA"
  name="{urn:example:purchasing}order"
  schemaFile="xsd/order.xsd"/>
```
Application Architecture

Test

Data Validation
Data Transform
Policy

Declarative Behavior

Remote

Implementation Logic

Remote
Testing

- Big Bang testing of SOA applications must stop!
- Develop and test your project iteratively
  - Service, transformation, binding, etc.
- SwitchYardRunner
  - Bootstraps runtime, components, and application
- MixIns
  - Enriches test case via composition vs. extension
  - CDI, HTTP, Smooks, BPM, HornetQ, Transaction, JCA
Service Test

```
@RunWith(SwitchYardRunner.class)
@SwitchYardTestCaseConfig(mixins = CDIMixIn.class)
public class InventoryServiceTest {

    @ServiceOperation("InventoryService.lookupItem")
    private Invoker lookupItem;

    @Test
    public void testItemLookupExists() throws Exception {
        final String ITEM_ID = "BUTTER";
        Item item = lookupItem
            .sendInOut(ITEM_ID)
            .getContent(Item.class);

        Assert.assertNotNull(item);
        Assert.assertEquals(ITEM_ID, item.getItemId());
    }
}
```
Service Test

Bootstraps SwitchYard runtime and handles test injection

```java
@RunWith(SwitchYardRunner.class)
@SwitchYardTestCaseConfig(mixins = CDIMixIn.class)
public class InventoryServiceTest {

@ServiceOperation("InventoryService.lookupItem")
private Invoker lookupItem;

@Test
public void testItemLookupExists() throws Exception {
    final String ITEM_ID = "BUTTER";
    Item item = lookupItem.sendInOut(ITEM_ID)
        .getContent(Item.class);
    Assert.assertNotNull(item);
    Assert.assertEquals(ITEM_ID, item.getItemId());
}
```
Bootstraps SwitchYard runtime and handles test injection

```java
@RunWith(SwitchYardRunner.class)
@SwitchYardTestCaseConfig(mixins = CDIMixIn.class)
public class InventoryServiceTest {

    @ServiceOperation("InventoryService.lookupItem")
    private Invoker lookupItem;

    @Test
    public void testItemLookupExists() throws Exception {
        final String ITEM_ID = "BUTTER";
        Item item = lookupItem.sendInOut(ITEM_ID)
                     .getContent(Item.class);

        Assert.assertNotNull(item);
        Assert.assertEquals(ITEM_ID, item.getItemId());
    }
}
```

Helper methods for CDI including Bean Scanning
Service Test

```java
@RunWith(SwitchYardRunner.class)
@SwitchYardTestCaseConfig(mixins = CDIMixIn.class)
public class InventoryServiceTest {

    @ServiceOperation("InventoryService.lookupItem")
    private Invoker lookupItem;

    @Test
    public void testItemLookupExists() throws Exception {
        final String ITEM_ID = "BUTTER";
        Item item = lookupItem.sendInOut(ITEM_ID).getContent(Item.class);
        Assert.assertNotNull(item);
        Assert.assertEquals(ITEM_ID, item.getItemId());
    }
}
```
@RunWith(SwitchYardRunner.class)
@SwitchYardTestCaseConfig(mixins = CDIMixIn.class)
public class InventoryServiceTest {

@ServiceOperation("InventoryService.lookupItem")
private Invoker lookupItem;

@Test
public void testItemLookupExists() throws Exception {
    final String ITEM_ID = "BUTTER";
    Item item = lookupItem.sendInOut(ITEM_ID)
        .getContent(Item.class);
    Assert.assertNotNull(item);
    Assert.assertEquals(ITEM_ID, item.getItemId());
}
}
@RunWith(SwitchYardRunner.class)
@SwitchYardTestCaseConfig(mixins = SmooksMixIn.class)
public class SmooksTransformationTest {

    private SmooksMixIn smooksMixIn;

    @Test
    public void testOrderTransform() throws Exception {
        // Verify the Order_XML.xml Smooks Java->XML binding
        smooksMixIn.testJavaXMLReadWrite(
            Order.class,
            "/smooks/Order_XML.xml",
            "/xml/order.xml");
    }
}
@RunWith(SwitchYardRunner.class)
@SwitchYardTestCaseConfig(
    config = SwitchYardTestCaseConfig.SWITCHYARD_XML,
    mixins = {CDIMixIn.class, HTTPMixIn.class})
public class WebServiceTest {

    private HTTPMixIn httpMixIn;

    @Test
    public void invokeOrderWebService() throws Exception {
        httpMixIn.postResourceAndTestXML(
            "http://localhost:18001/OrderService",
            "xml/soap-request.xml",
            "xml/soap-response.xml";
        }
    }
}
Demo Time!
Add SwitchYard to your application

[ExampleService] ExampleService $ project install-facet switchyard.bpm
[ExampleService] ExampleService $ project install-facet switchyard.soap

Create a BPMN 2 workflow service

[ExampleService] ExampleService $ bpm-service --serviceName ExampleService

Bind the service to SOAP / HTTP

[ExampleService] ExampleService $ switchyard promote-service --serviceName ExampleService
[ExampleService] ExampleService $ soap-binding bind-service --serviceName ExampleService --wsdl wsdl/Example.wsdl
Runtime Options

- Supported Containers
  - JBoss AS 7 (7.1.1.Final)
  - OpenShift (7.1.0.Final)
  - Java SE - unit test or standalone
  - Tomcat

- Application Deployment Options
  - JAR, WAR, EAR
Cloud

> rhc app create -a myapp -t jbossas-7
> cd myapp
> forge
> project install-facet switchyard
Wait! There’s more ...

- Administration
- Policy
- Governance
- Futures
- JBoss World: “SOA at Scale with SwitchYard”
  - Thursday @ 4:50pm
Join Us!

• Learn
  • http://www.jboss.org/switchyard

• Play
  • 0.5 Beta 1 is out!
  • http://github.com/jboss-switchyard/quickstarts

• Chat
  • chat.freenode.net #switchyard

• Fork
  • http://github.com/jboss-switchyard
Questions?
JUDCon
JBoss Users & Developers Conference
2012: Boston