JBoss Enterprise SOA Middleware

Internal Training

Tom Cunningham, JBoss ESB

Spring 2012

ESB / jBPM 5 integration

- BPM5Processor action – meant to process a single jBPM command
- Possible commands
  - startProcess
  - signalEvent
  - abortProcess

BPM5Processor Action

- <action name="jbpm5" class="org.jboss.soa.esb.services.jbpm5.actions.Bpm5Processor">
- <property name="process-definition-name" value="Evaluation.bpmn"/>
- <property name="process-id" value="com.sample.evaluation"/>
- <property name="process-action" value="startProcess"/>
- <property name="esbToBpmParams">
- <mapping esb="employee" bpm="employee" value="krisv"/>
- <mapping esb="reason" bpm="reason" value="Yearly performance review"/>
- </property>
-
- </action>
-

BPM5Processor Action

- Required attributes :
  - process-definition-name
  - process-id
  - process-action (StartProcess/SignalEvent/AbortProcess)
  - esbToBpmParams

EsbToBpmParams

- Different ways to supply mappings

```
<mapping esb="employee" bpm="employee" value="krisv"/>

<mapping esb="reason" bpm="reason" value="Yearly performance review"/>
```

- “esb” says to map to the “X” property in the message properties
- value provides a hard-coded value

Optional properties

- Handlers
- handlerClass, handlerHost, handlerPort

Camel Gateway

- Expands JBoss ESB routing and mediation
- Not specific to any one type of transport – allows you to make use of Camel components
- Many different types of Camel components – see http://camel.apache.org/components.html for a list

Camel Gateway

- Requires jbossesb-1.3.0.xsd version of the jboss-esb.xml schema or above
- Define your routes either in a JBoss ESB provider or list them in the <camel-gateway> section of your <listeners/>
- SOA-P only ships camel-core, you need to supply the component and any JARs it might require in server/<config>/deployers/esb.deployer/lib

Routes in provider

```
<jbossesb xmlns="http://anonsvn.labs.jboss.com/labs/jbossesb/trunk/product/etc/schemas/xml/jbossesb-1.3.0.xsd"
parameterReloadSecs="5">

    <providers>

        <camel-provider name="provider1">

            <camel-bus busid="bus1">

                <from uri="file://.../samples/quickstarts/camel_helloworld/build/input1?delete=true"/>

                <from uri="file://.../samples/quickstarts/camel_helloworld/build/input2?delete=true"/>

            </camel-bus>

        </camel-provider>

    </providers>

    <services>

        <service category="camel_helloworld" name="service1" description="Hello World" invmScope="GLOBAL">

            <listeners>

                <camel-gateway name="gateway1" busidref="bus1"/>

            </listeners>
```

Routes in listener

```
<jbossesb xmlns="http://anonsvn.labs.jboss.com/labs/jbossesb/trunk/product/etc/schemas/xml/jbossesb-1.3.0.xsd"
parameterReloadSecs="5">

    <services>

      <service category="camel_helloworld" name="service1" description="Hello World" invmScope="GLOBAL">

          <listeners>

              <camel-gateway name="gateway1">

                  <from uri="file://.../samples/quickstarts/camel_helloworld/build/input1?delete=true"/>

                  <from uri="file://.../samples/quickstarts/camel_helloworld/build/input2?delete=true"/>

              </camel-gateway>

          </listeners>

          <actions>

                ...
```

Global Pipeline Interceptors

- Set interceptors as properties in jbossesb-properties.xml :

```
<properties name="interceptors">
        <property name="org.jboss.soa.esb.pipeline.failure.interceptors"
        value=""org.jboss.soa.esb.listeners.message.GenericPipelineInterceptor"/>
        <property name="org.jboss.soa.esb.pipeline.instantiate.interceptors"
        value="org.jboss.soa.esb.listeners.message.GenericPipelineInterceptor"/>
        <property name="org.jboss.soa.esb.pipeline.start.interceptors"
        value="org.jboss.soa.esb.listeners.message.GenericPipelineInterceptor"/>
        <property name="org.jboss.soa.esb.pipeline.end.interceptors"
         value="org.jboss.soa.esb.listeners.message.GenericPipelineInterceptor"/>
</properties>
```

Global Pipeline Interceptors

Simple to create an interceptor class :

```
public class GenericPipelineInterceptor implements PipelineInterceptor {
	Logger logger = Logger.getLogger(GenericPipelineInterceptor.class);

	@Override
	public void processMessage(Message msg, ConfigTree config) {
		logger.info(msg.getBody().toString());
	}
}
```

Record Route

- Requires jbossesb-1.3.1.xsd schema
- Logging and debugging feature
- org.jboss.internal.soa.esb.ServiceRouteFilter in jbossesb-properties.xml
- If you want to record routes at the service level, set the service-record-route attribute to “true”
- On message level, set msg.getProperties().setProperty(

  ListenerTagNames.SERVICE_RECORD_ROUTE_TAG, "true");

Streaming Aggregator Action

- Big performance improvement over aggregation action in SOA-P 5.2 and earlier
- Don't have to know the total number of rows in the aggregation before you are finished – no need to pre-count the aggregation
- Shaved minutes of time off of a POC that aggregated large amounts of records (1 million +)

Streaming Aggregator details

- Set a unique ID for aggregation and the rank of the message within the aggregation
- message.getProperties().setProperty(

  AggregateDetails.AGGREGATE_DETAILS, new AggregateDetails(seriesUUID, Integer.valueOf(currentRecord++))
- When done, set the sequence count : ((AggregateDetails)message.getProperties().getProperty(AggregateDetails.AGGREGATE_DETAILS)).setSequenceCount(Integer.valueOf(currentRecord)) ;

Smooks Fragment Router

- Streaming Aggregator's ability to deal with message sets that are not pre-counted allows us to aggregate Smooks fragments
- Smooks Fragment Router now keeps an internal count of the number of fragments generated
- Automatically will send aggregation details