Taking Apache Camel For a Ride

Bruce Snyder
bsnyder@apache.org
October 2008
Keystone, Colorado
Taking Apache Camel For a Ride
Why it's bad when home owners change their minds about the bathroom's location late in a building project.
Protocol Integration is Common
Data Format Integration is Common
Integration is Messy!
INTEGRATION

Just because you can, doesn’t mean you should.
SOA = Spaghetti Oriented Architecture
Options For Integration

1. THE HOME DEPOT

2. Money

3. Open Source
Too Many Choices!
The Easiest Solution - Apache Camel

http://activemq.apache.org/camel/
Enterprise Integration Patterns

http://enterpriseintegrationpatterns.com/
Message Routing
# Message Routing

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Based Router</td>
<td>How do we handle a situation where the implementation of a single logical function (e.g., inventory check) is spread across multiple physical systems?</td>
</tr>
<tr>
<td>Message Filter</td>
<td>How can a component avoid receiving uninteresting messages?</td>
</tr>
<tr>
<td>Recipient List</td>
<td>How do we route a message to a list of dynamically specified recipients?</td>
</tr>
<tr>
<td>Splitter</td>
<td>How can we process a message if it contains multiple elements, each of which may have to be processed in a different way?</td>
</tr>
<tr>
<td>Aggregator</td>
<td>How do we combine the results of individual, but related messages so that they can be processed as a whole?</td>
</tr>
<tr>
<td>Resequencer</td>
<td>How can we get a stream of related but out-of-sequence messages back into the correct order?</td>
</tr>
<tr>
<td>Routing Slip</td>
<td>How do we route a message consecutively through a series of processing steps when the sequence of steps is not known at design-time and may vary for each message?</td>
</tr>
<tr>
<td>Throttler</td>
<td>How can I throttle messages to ensure that a specific endpoint does not get overloaded, or we don't exceed an agreed SLA with some external service?</td>
</tr>
<tr>
<td>Delayer</td>
<td>How can I delay the sending of a message?</td>
</tr>
<tr>
<td>Load Balancer</td>
<td>How can I balance load across a number of endpoints?</td>
</tr>
<tr>
<td>Multicast</td>
<td>How can I route a message to a number of endpoints at the same time?</td>
</tr>
</tbody>
</table>
History of Apache Camel
Camel Components

http://activemq.apache.org/camel/components.html
# Camel Components

<table>
<thead>
<tr>
<th>ActiveMQ</th>
<th>File</th>
<th>JBI</th>
<th>MINA</th>
<th>RMI</th>
<th>TCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveMQ Journal</td>
<td>FIX</td>
<td>JCR</td>
<td>Mock</td>
<td>RNC</td>
<td>Test</td>
</tr>
<tr>
<td>AMQP</td>
<td>Flatpack</td>
<td>JDBC</td>
<td>MSMQ</td>
<td>RNG</td>
<td>Timer</td>
</tr>
<tr>
<td>Atom</td>
<td>FTP</td>
<td>Jetty</td>
<td>MSV</td>
<td>SEDA</td>
<td>UDP</td>
</tr>
<tr>
<td>Bean</td>
<td>Hibernate</td>
<td>JMS</td>
<td>Multicast</td>
<td>SFTP</td>
<td>Validation</td>
</tr>
<tr>
<td>CXF</td>
<td>HTTP</td>
<td>JPA</td>
<td>POJO</td>
<td>SMTP</td>
<td>Velocity</td>
</tr>
<tr>
<td>DataSet</td>
<td>iBATIS</td>
<td>JT/400</td>
<td>POP</td>
<td>Spring Integration</td>
<td>VM</td>
</tr>
<tr>
<td>Direct</td>
<td>IMAP</td>
<td>List</td>
<td>Quartz</td>
<td>SQL</td>
<td>XMPP</td>
</tr>
<tr>
<td>Esper</td>
<td>IRC</td>
<td>Log</td>
<td>Queue</td>
<td>Stream</td>
<td>XQuery</td>
</tr>
<tr>
<td>Event</td>
<td>JavaScript</td>
<td>Mail</td>
<td>Ref</td>
<td>String Template</td>
<td>XSLT</td>
</tr>
</tbody>
</table>
Simple Routing

- Endpoint A
- File
- JMS
- Endpoint B

Message flow from endpoint A through a file to JMS, then to endpoint B.
More Simple Routing
Pipeline Routing
Multicast Routing

Diagram showing a message being sent from Endpoint A to Endpoint B through various protocols such as FILE, iBATiS, SFTP, and SMTP.
Multicast-to-Many Pipeline Routes
Language Support

- BeanShell
- Javascript
- Groovy
- Python
- PHP
- Ruby
- JSP EL
- OGNL
- SQL
- XPath
- XQuery
Getting Started -
The Camel Context

```java
CamelContext context = new DefaultCamelContext();
context.addRoutes(new MyRouteBuilder());
context.start();
```

```xml
<camelContext
    xmlns="http://activemq.apache.org/camel/schema/spring">
    <package>com.acme.quotes</package>
</camelContext>
```
Pattern Examples
# Message Routing

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Based Router</td>
<td>How do we handle a situation where the implementation of a single logical function (e.g., inventory check) is spread across multiple physical systems?</td>
</tr>
<tr>
<td>Message Filter</td>
<td>How can a component avoid receiving uninteresting messages?</td>
</tr>
<tr>
<td>Recipient List</td>
<td>How do we route a message to a list of dynamically specified recipients?</td>
</tr>
<tr>
<td>Splitter</td>
<td>How can we process a message if it contains multiple elements, each of which may have to be processed in a different way?</td>
</tr>
<tr>
<td>Aggregator</td>
<td>How do we combine the results of individual, but related messages so that they can be processed as a whole?</td>
</tr>
<tr>
<td>Resequencer</td>
<td>How can we get a stream of related but out-of-sequence messages back into the correct order?</td>
</tr>
<tr>
<td>Routing Slip</td>
<td>How do we route a message consecutively through a series of processing steps when the sequence of steps is not known at design-time and may vary for each message?</td>
</tr>
<tr>
<td>Throttler</td>
<td>How can I throttle messages to ensure that a specific endpoint does not get overloaded, or we don't exceed an agreed SLA with some external service?</td>
</tr>
<tr>
<td>Delayer</td>
<td>How can I delay the sending of a message?</td>
</tr>
<tr>
<td>Load Balancer</td>
<td>How can I balance load across a number of endpoints?</td>
</tr>
<tr>
<td>Multicast</td>
<td>How can I route a message to a number of endpoints at the same time?</td>
</tr>
</tbody>
</table>
Content Based Router

New Order → Router → Widget Inventory

New Order → Router → Gadget Inventory
Content Based Router

```xml
<camelContext
    xmlns="http://activemq.apache.org/camel/schema/spring">
    <route>
        <from uri="activemq:NewOrders"/>
        <choice>
            <when>
                <xpath>/order/product = 'widget'</xpath>
                <to uri="activemq:Orders.Widgets"/>
            </when>
            <when>
                <xpath>/order/product = 'gadget'</xpath>
                <to uri="activemq:Orders.Gadgets"/>
            </when>
            <otherwise>
                <to uri="activemq:Orders.Bad"/>
            </otherwise>
        </choice>
    </route>
</camelContext>
```
Message Filter
Message Filter
Splitter

New Order → Splitter → Order Item 1, Order Item 2, Order Item 3
Splitter Using XQuery
public class MyRouteBuilder extends RouteBuilder {
    public void configure() {
        from("activemq:Inventory.Items").
            aggregator(header("symbol").isEqualTo("IBM")).
            to("activemq:Inventory.Order");
    }
}
Message Translator

Incoming Message  Translator  Translated Message
Resequencer

Bruce Snyder — Taking Apache Camel for a Ride
Routing Slip

Attach Routing Slip to Message

Route Message According to Slip

Proc A

Proc B

Proc C
Throttler
Delayer
## Load Balancer

<table>
<thead>
<tr>
<th>Policy</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Round Robin</td>
<td>Balance the exchange load across the available endpoints</td>
</tr>
<tr>
<td>Random</td>
<td>Randomly choose an endpoint to send the exchange</td>
</tr>
<tr>
<td>Sticky</td>
<td>Sticky load balancing of exchanges using an expression</td>
</tr>
<tr>
<td>Topic</td>
<td>Send exchange to all endpoints (like a JMS topic)</td>
</tr>
</tbody>
</table>
Multicast
Demo
More Patterns
Wire Tap

Source -> Wire Tap -> Destination
Content Enricher

Basic Message → Enricher → Enriched Message

Resource
More Content Enricher

- Basic Message
- Enricher
- Enriched Message
- Resource
Content Filter

Message → Content Filter → Message
Combine Patterns
Configure Error Handling

```java
RouteBuilder builder = new RouteBuilder() {
    public void configure() {
        errorHandler(deadLetterChannel("file:errors"));
        from("bean:foo").to("seda:b");
    }
};
```
Configure Error Handling

```java
RouteBuilder builder = new RouteBuilder() {
    public void configure() {
        from("seda:a").errorHandler(loggingErrorHandler("FOO.BAR")).
        to("seda:b");

        from("seda:b").to("seda:c");
    }
};
```
Configure Exception Policies

```java
RouteBuilder builder = new RouteBuilder()
{
    public void configure() {
        exception(IOException.class)
            .initialRedeliveryDelay(5000L)
            .maximumRedeliveries(3)
            .maximumRedeliveryDelay(30000L)
            .backOffMultiplier(1.0)
            .useExponentialBackOff()
            .setHeader(MESSAGE_INFO, constant("Damned IOException!"))
            .to("activemq:errors");

        from("seda:a").to("seda:b");
    }
};
```
Beans
Make Context Discover Beans
Bean as a Message Translator
Bean as a Message Translator

*With Method Name*

```java
public class MyRouteBuilder extends RouteBuilder {
    public void configure() {
        from("activemq:Incoming").
            beanRef("myBean", "someMethod").
            to("activemq:Outgoing");
    }
}
```
Binding Beans to Camel Endpoints

```java
public class Foo {

    @MessageDriven(uri="activemq:cheese")
    public void onCheese(String name) {
        ...
    }
}
```
Binding Method Arguments
Injecting Endpoints Into Beans
Type Conversion
Type Convertors

Support for the following types:

- File
- String
- byte[] and ByteBuffer
- InputStream and OutputStream
- Reader and Writer
- Document and Source
Message Mapper
Message Translator

Incoming Message  Translator  Translated Message
Another Message Translator

```java
public class MyRouteBuilder extends RouteBuilder {
    public void configure() {
        from("activemq:FOO.TEST").
            transform(body().append(getDynamicText())).
        to("http://outgoing.com/foo");
    }
}
```
Business Activity Monitoring (BAM)
public class MyActivities extends ProcessBuilder {

    public void configure() throws Exception {

        // lets define some activities, correlating on an
        // XPath query of the message body
        ActivityBuilder purchaseOrder = activity("activemq:PurchaseOrders")
            .correlate(xpath("/purchaseOrder/@id").stringResult());

        ActivityBuilder invoice = activity("activemq:Invoices")
            .correlate(xpath("/invoice/@purchaseOrderId").stringResult());

        // now lets add some BAM rules
        invoice.starts().after(purchaseOrder.completes())
            .expectWithin(seconds(1))
            .errorIfOver(seconds(2)).to("activemq:FailedProcesses");
    }
}
Complex Routing is Easier

```java
from("http://localhost:8080/requests/").
  tryBlock().
    to("activemq:queue:requests").
    setOutBody(constant("<ack/>")).
    handle(Throwable.class).
    setFaultBody(constant("<nack/>"));

from(("activemq:queue:requests?transacted=true").
  process(requestTransformer).
  to("http://host:8080/Request").
  filter(xpath("//nack")).
  process(nackTransformer).
  to("jdbc:store");

from("http://localhost:8080/responses/").
  tryBlock().
    to("activemq:queue:responses").
    setOutBody(constant("<ack/>")).
    handle(Throwable.class).
    setFaultBody(constant("<nack/>"));

from("activemq:queue:responses?transacted=true").
  process(responseTransformer).
  to("jdbc:store");

from("http://localhost:8080/pull/").
  to("jdbc:load");
```
Finally, the Camel Truck!
Ride the Camel!

http://activemq.apache.org/camel/