Easy Enterprise Integration Patterns with Apache Camel, ActiveMQ and ServiceMix

James Strachan
http://macstrac.blogspot.com/

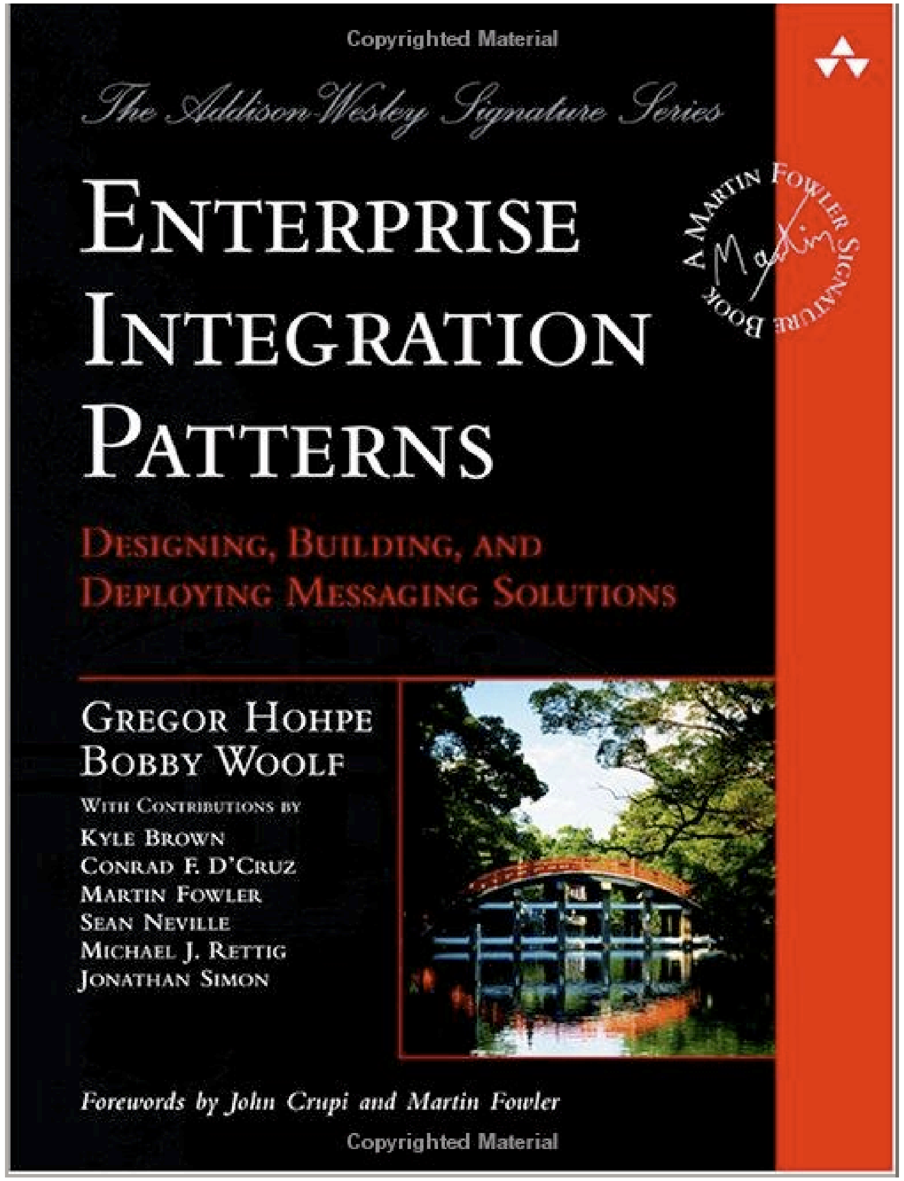
IONA | Open Source Community

http://open.iona.com/

ijtc 2007

What are Enterprise Integration Patterns?

Book by Gregor & Bobby!



Copyrighted Material

The Addison-Wesley Signature Series

ENTERPRISE INTEGRATION PATTERNS

DESIGNING, BUILDING, AND DEPLOYING MESSAGING SOLUTIONS

GREGOR HOHPE
BOBBY WOOLF

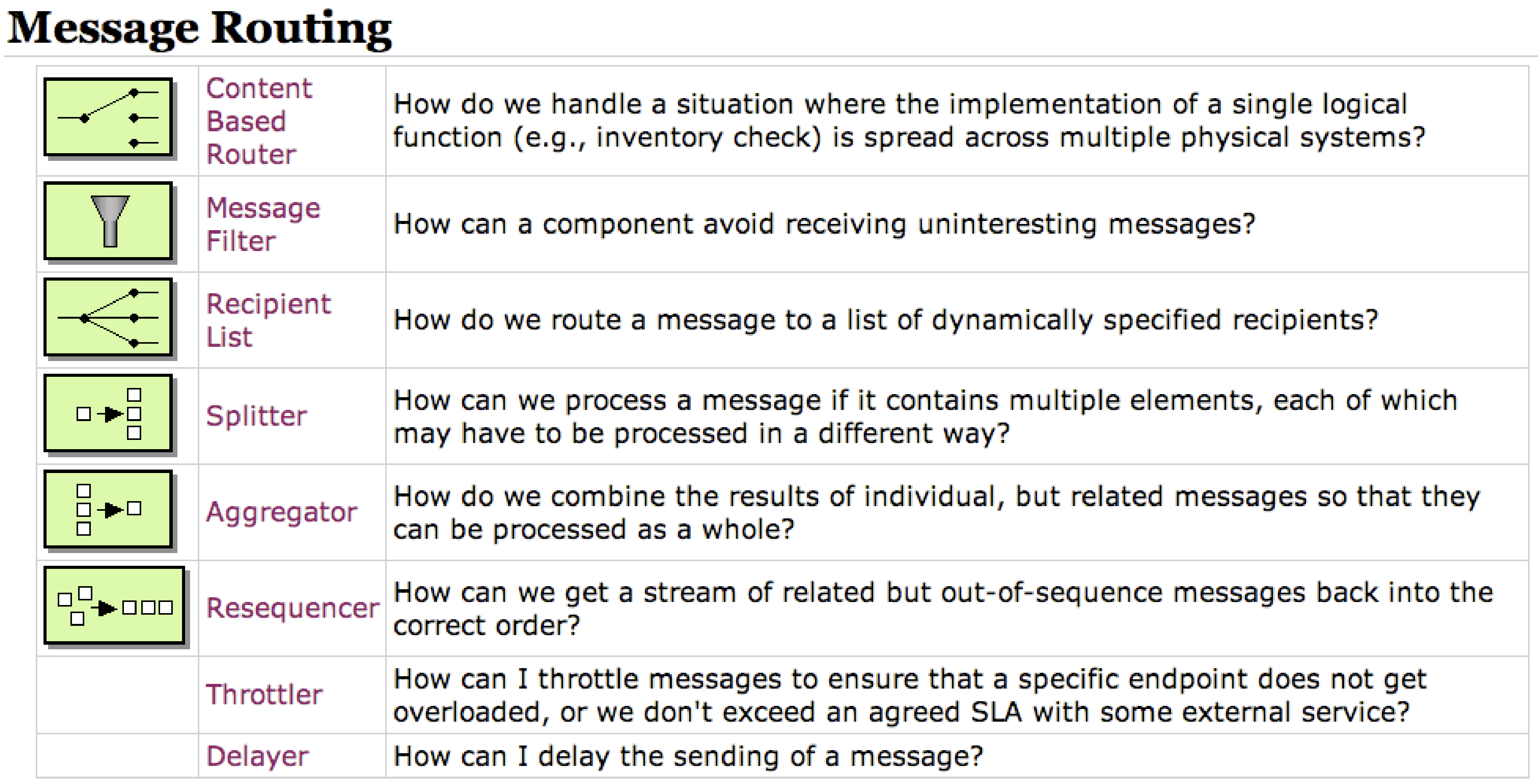
WITH CONTRIBUTIONS BY
KYLE BROWN
CONRAD F. D'CRUZ
MARTIN FOWLER
SEAN NEVILLE
MICHAEL J. RETTIG
JONATHAN SIMON

A MARTIN FOWLER SIGNATURE BOOK

Forewords by John Crupi and Martin Fowler

Copyrighted Material

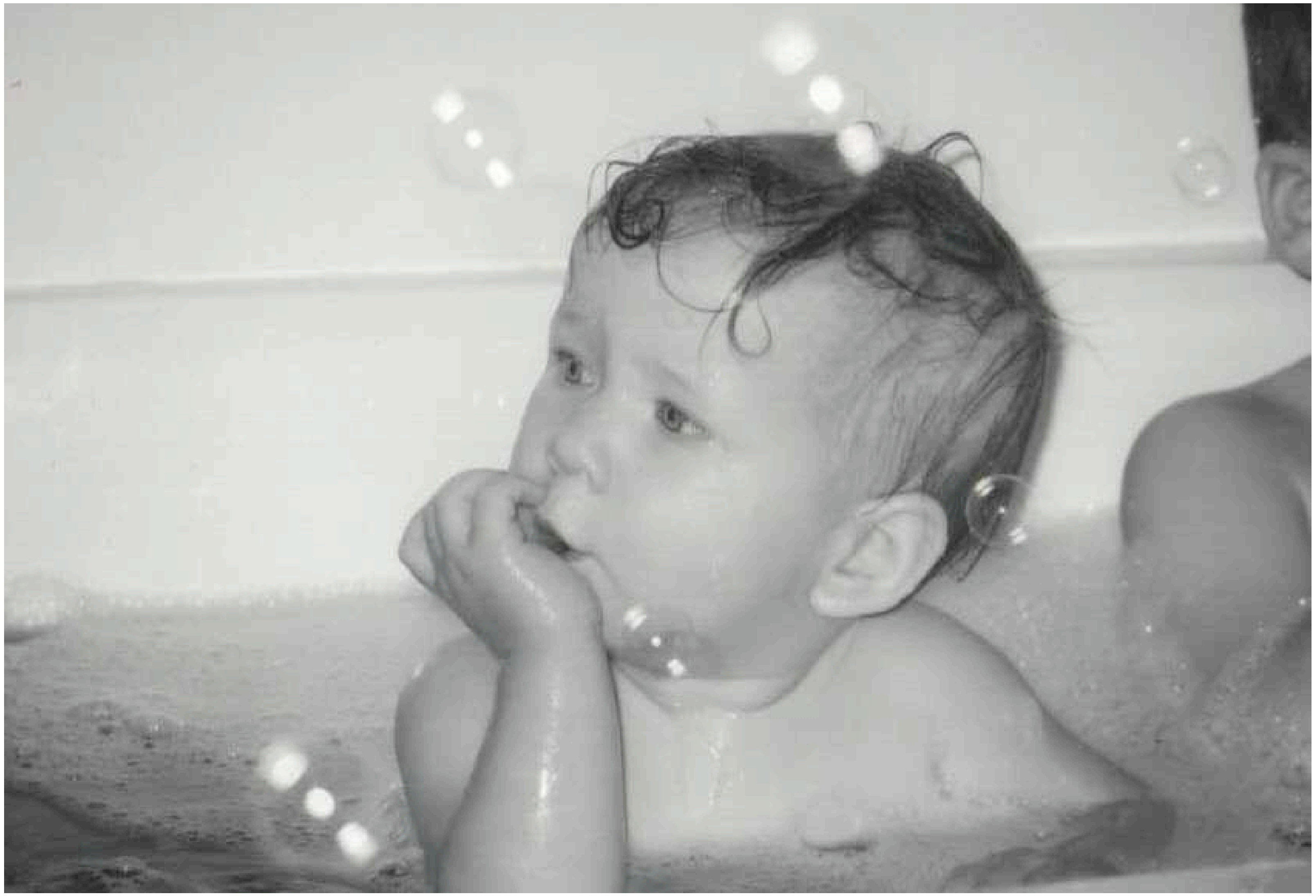
A selection of some of the patterns...



Message Routing

	Content Based Router	How do we handle a situation where the implementation of a single logical function (e.g., inventory check) is spread across multiple physical systems?
	Message Filter	How can a component avoid receiving uninteresting messages?
	Recipient List	How do we route a message to a list of dynamically specified recipients?
	Splitter	How can we process a message if it contains multiple elements, each of which may have to be processed in a different way?
	Aggregator	How do we combine the results of individual, but related messages so that they can be processed as a whole?
	Resequencer	How can we get a stream of related but out-of-sequence messages back into the correct order?
	Throttler	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?
	Delayer	How can I delay the sending of a message?

What is Camel?



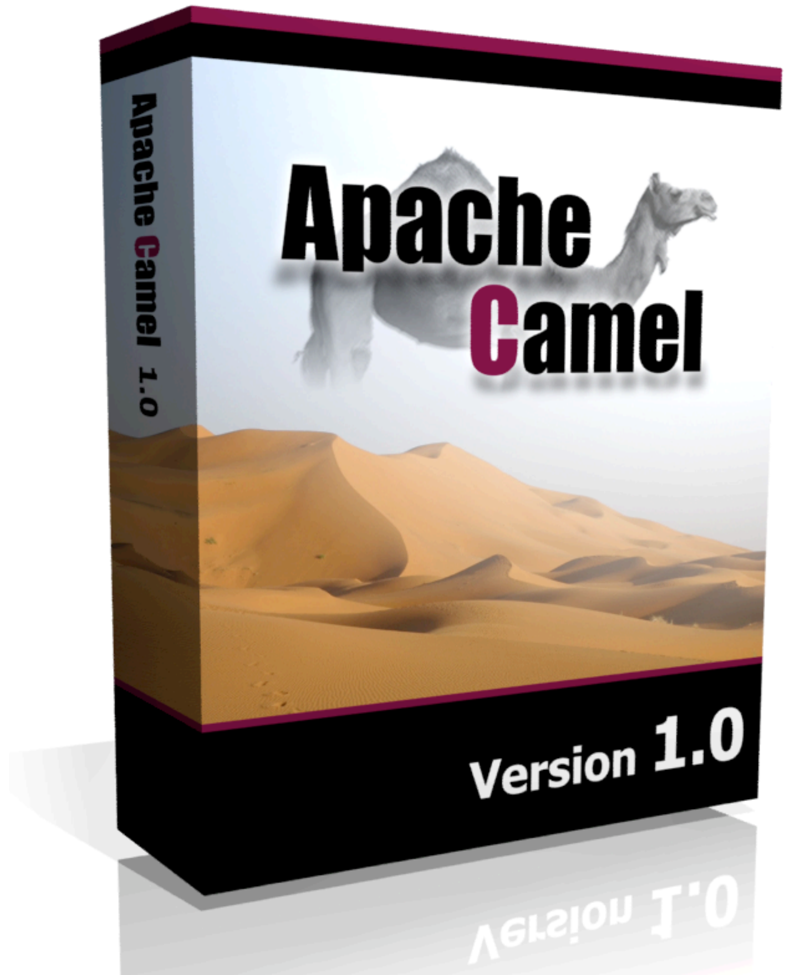
Why?

Aims of Camel

to make integration as simple as it can possibly be

What is Camel?

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

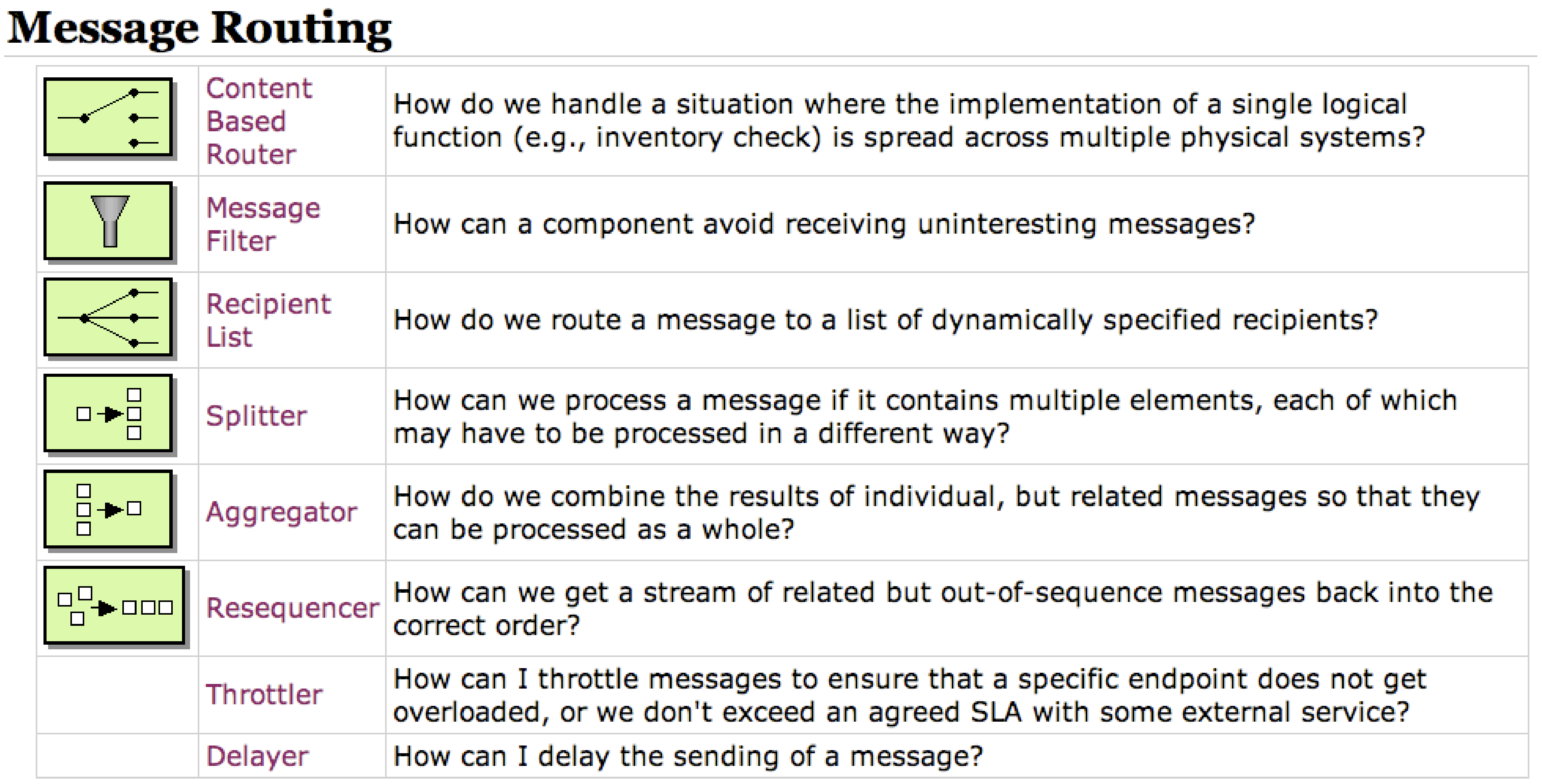


What is Camel?

Spring based Enterprise Integration Patterns

http://activemq.apache.org/camel/enterprise-integration-patterns.html

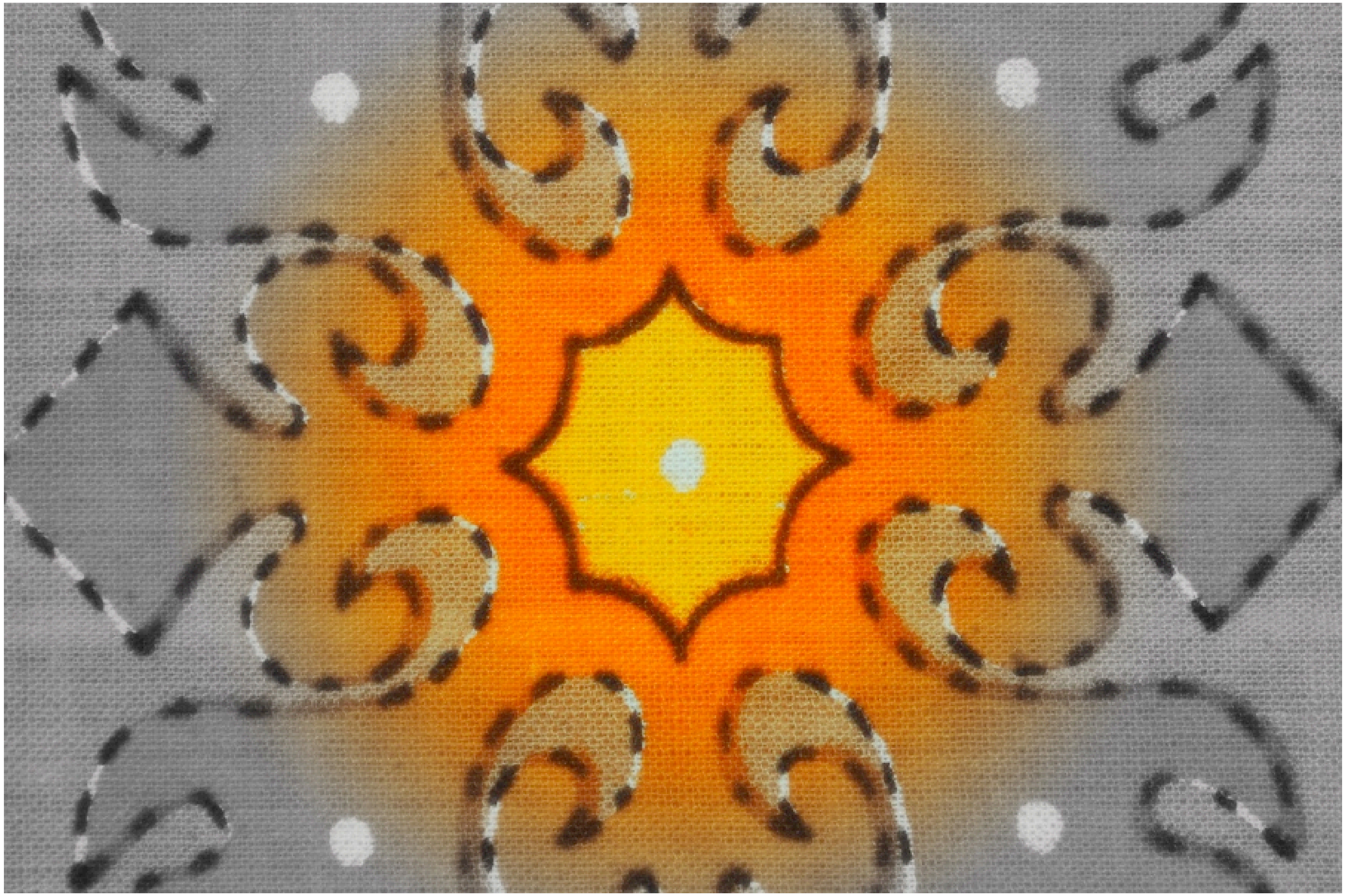
A selection of some of the patterns...



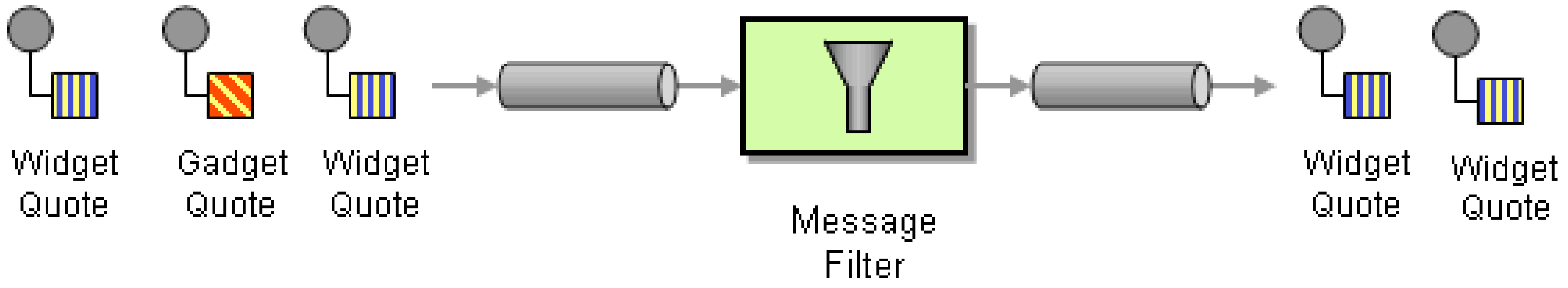
Message Routing

	Content Based Router	How do we handle a situation where the implementation of a single logical function (e.g., inventory check) is spread across multiple physical systems?
	Message Filter	How can a component avoid receiving uninteresting messages?
	Recipient List	How do we route a message to a list of dynamically specified recipients?
	Splitter	How can we process a message if it contains multiple elements, each of which may have to be processed in a different way?
	Aggregator	How do we combine the results of individual, but related messages so that they can be processed as a whole?
	Resequencer	How can we get a stream of related but out-of-sequence messages back into the correct order?
	Throttler	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?
	Delayer	How can I delay the sending of a message?

Lets look at a pattern!



Message Filter



Message Filter : XML

```
<camelContext xmlns="http://activemq.apache.org/camel/schema/spring">
  <route>
    <from uri="activemq:topic:Quotes"/>
    <filter>
      <xpath>/quote/product = ‘widget’</xpath>
      <to uri="mqseries:WidgetQuotes"/>
    </filter>
  </route>
</camelContext>
```

Message Filter : Spring XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
       http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
       http://activemq.apache.org/camel/schema/spring
       http://activemq.apache.org/camel/schema/spring/camel-spring.xsd">

 <camelContext xmlns="http://activemq.apache.org/camel/schema/spring">
    <route>
      <from uri="activemq:topic:Quotes"/>
      <filter>
        <xpath>/quote/product = ‘widget’</xpath>
        <to uri="mqseries:WidgetQuotes"/>
      </filter>
    </route>
  </camelContext>

</beans>
```

Message Filter : XML

```
<camelContext xmlns="http://activemq.apache.org/camel/schema/spring">
  <route>
    <from uri="activemq:topic:Quotes"/>
    <filter>
      <xpath>/quote/product = ‘widget’</xpath>
      <to uri="mqseries:WidgetQuotes"/>
    </filter>
  </route>
</camelContext>
```

Expressions & Predicates

BeanShell	PHP
EL	Python
Groovy	Ruby
JavaScript	SQL
JSR 223	XPath
OGNL	XQuery

URIs, Endpoints and Components

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

activemq	ibatis	mail	rmi	udp
activemq.journal	imap	mina	rnc	validation
bean	irc	mock	rng	velocity
cxf	jdbc	msv	seda	vm
direct	jetty	multicast	sftp	xmpp
event	jbi	pojo	smtp	xquery
file	jms	pop	string-template	xslt
ftp	jpa	quartz	timer	webdav
http	log	queue	tcp	

Message Filter : XML

```
<camelContext xmlns="http://activemq.apache.org/camel/schema/spring">
  <route>
    <from uri="activemq:topic:Quotes"/>
    <filter>
      <xpath>/quote/product = ‘widget’</xpath>
      <to uri="mqseries:WidgetQuotes"/>
    </filter>
  </route>
</camelContext>
```

Message Filter : Java

```
from("activemq:topic:Quotes).
  filter().xpath("/quote/product = ‘widget’").
    to("mqseries:WidgetQuotes");
```

Message Filter : Java Complete

```
package com.acme.quotes;

import org.apache.camel.builder.RouteBuilder;

public class MyRouteBuilder extends RouteBuilder {

    public void configure() {

        // forward widget quotes to MQSeries
        from("activemq:topic:Quotes).
                filter().xpath("/quote/product = ‘widget’").
                to("mqseries:WidgetQuotes");
   }
}
```

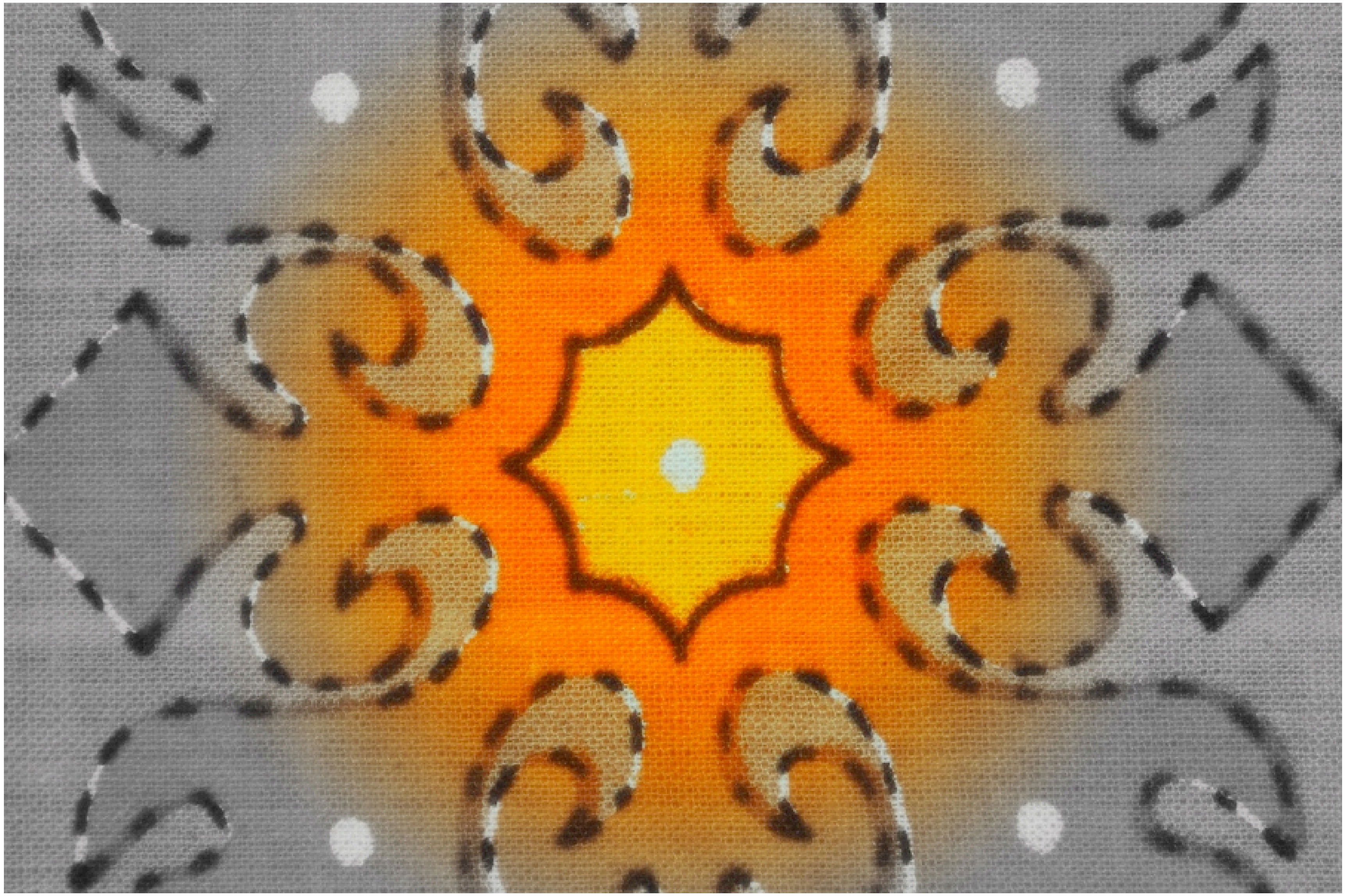
Create CamelContext in Java

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

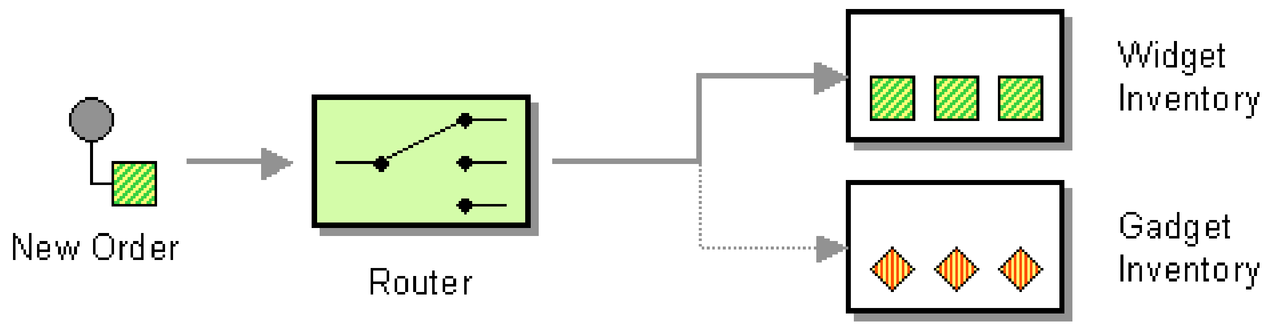
Create CamelContext in Spring

```
<camelContext xmlns="http://activemq.apache.org/camel/schema/spring">
  <package>com.acme.quotes</package>
</camelContext>
```

More Patterns!



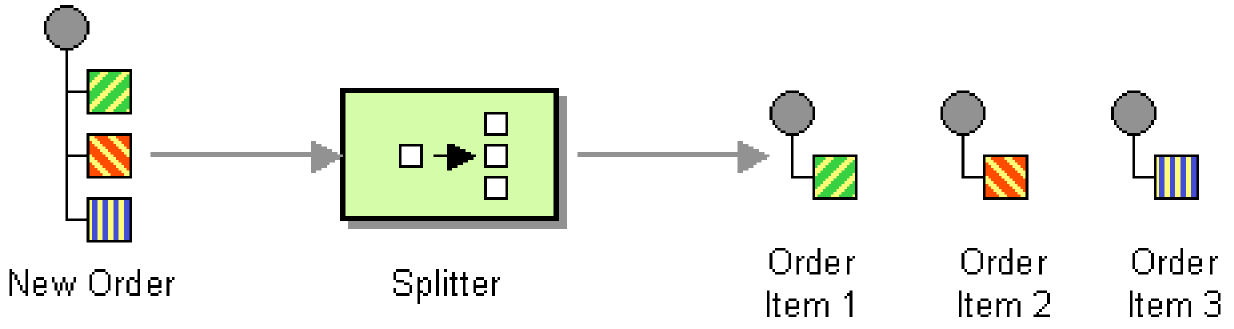
Content Based Router



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>
```

Splitter



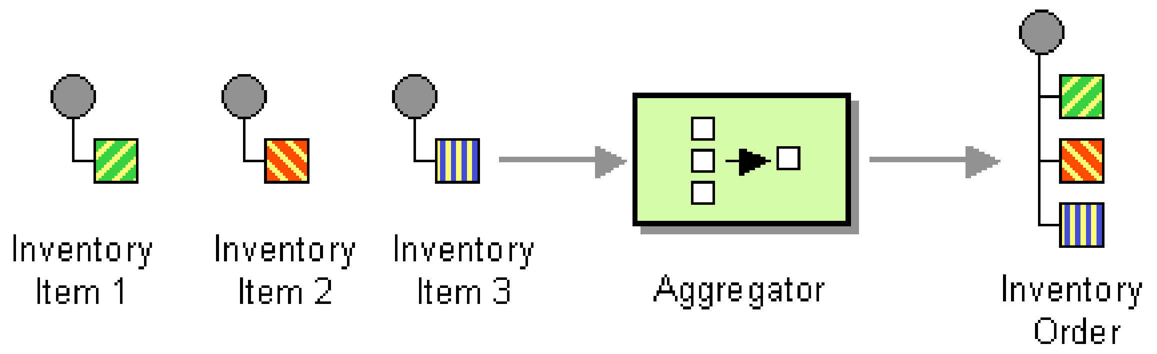
Splitter

```
from("file://orders").
  splitter(body().tokenize("\n")).
    to("activemq:Order.Items");
```

Splitter using XQuery

```
from("file://orders").
  splitter().xquery("/order/items").
    to("activemq:Order.Items");
```

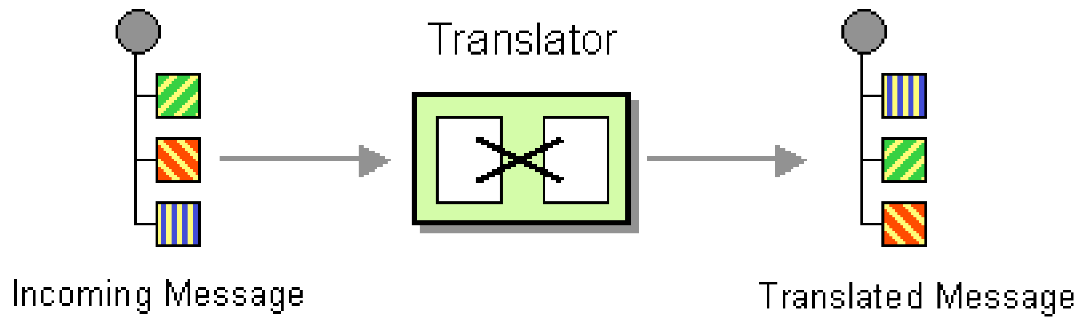
Aggregator


Inventory Item 1
Inventory Item 2
Inventory Item 3
Aggregator
Inventory Order

Aggregator

```
from("activemq:Inventory.Items").
  aggregator().xpath("/order/@id").
  to("activemq:Inventory.Order");
```

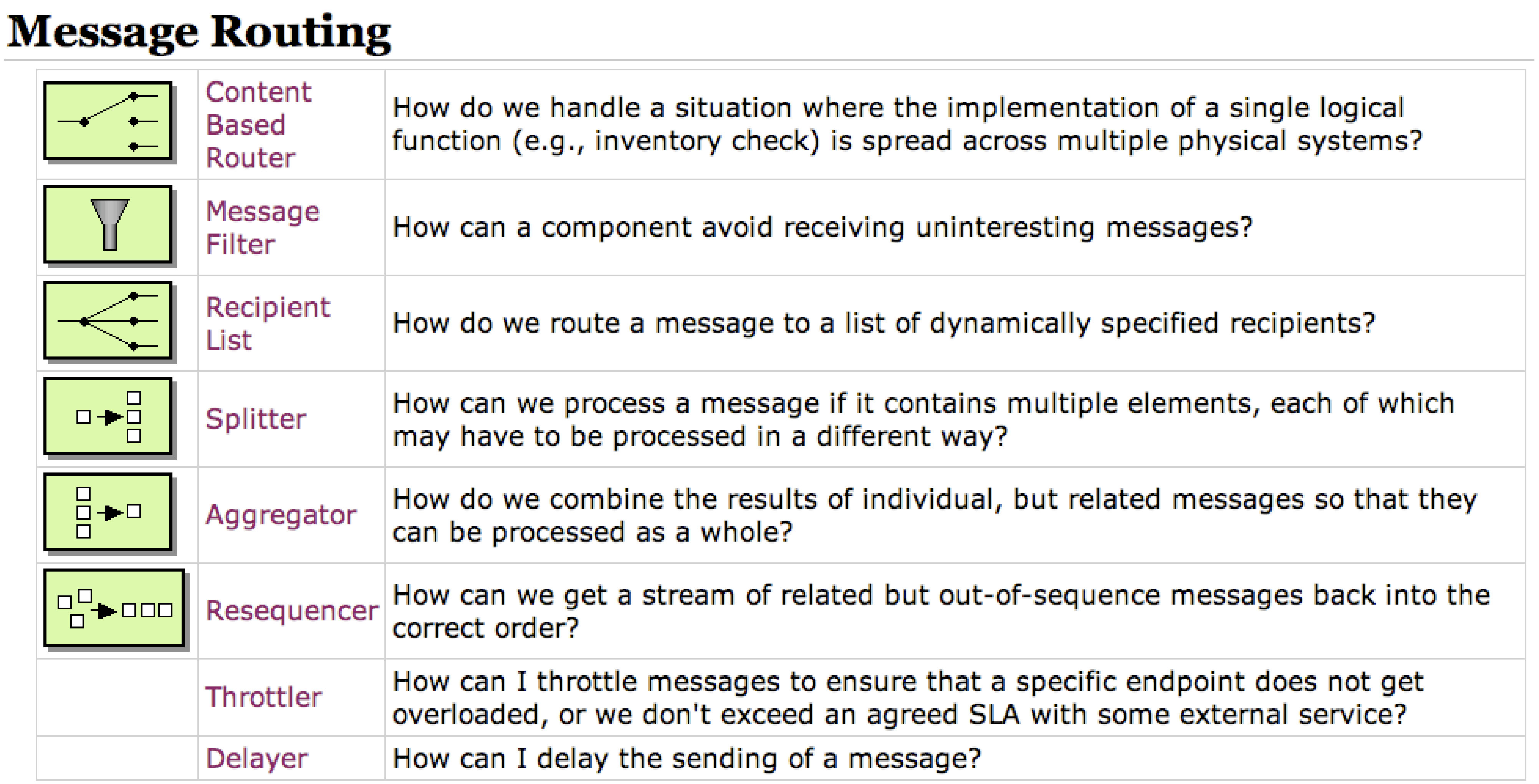
Message Translator



Message Translator

```
from("file://incoming”).
  to("xslt:com/acme/mytransform.xsl").
    to("http://outgoing.com/foo");
```

Quick recap



Message Routing

	Content Based Router	How do we handle a situation where the implementation of a single logical function (e.g., inventory check) is spread across multiple physical systems?
	Message Filter	How can a component avoid receiving uninteresting messages?
	Recipient List	How do we route a message to a list of dynamically specified recipients?
	Splitter	How can we process a message if it contains multiple elements, each of which may have to be processed in a different way?
	Aggregator	How do we combine the results of individual, but related messages so that they can be processed as a whole?
	Resequencer	How can we get a stream of related but out-of-sequence messages back into the correct order?
	Throttler	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?
	Delayer	How can I delay the sending of a message?

Beans

Bean as a Message Translator

```
from("activemq:Incoming”).
  beanRef("myBeanName").
    to("activemq:Outgoing");
```

Bean

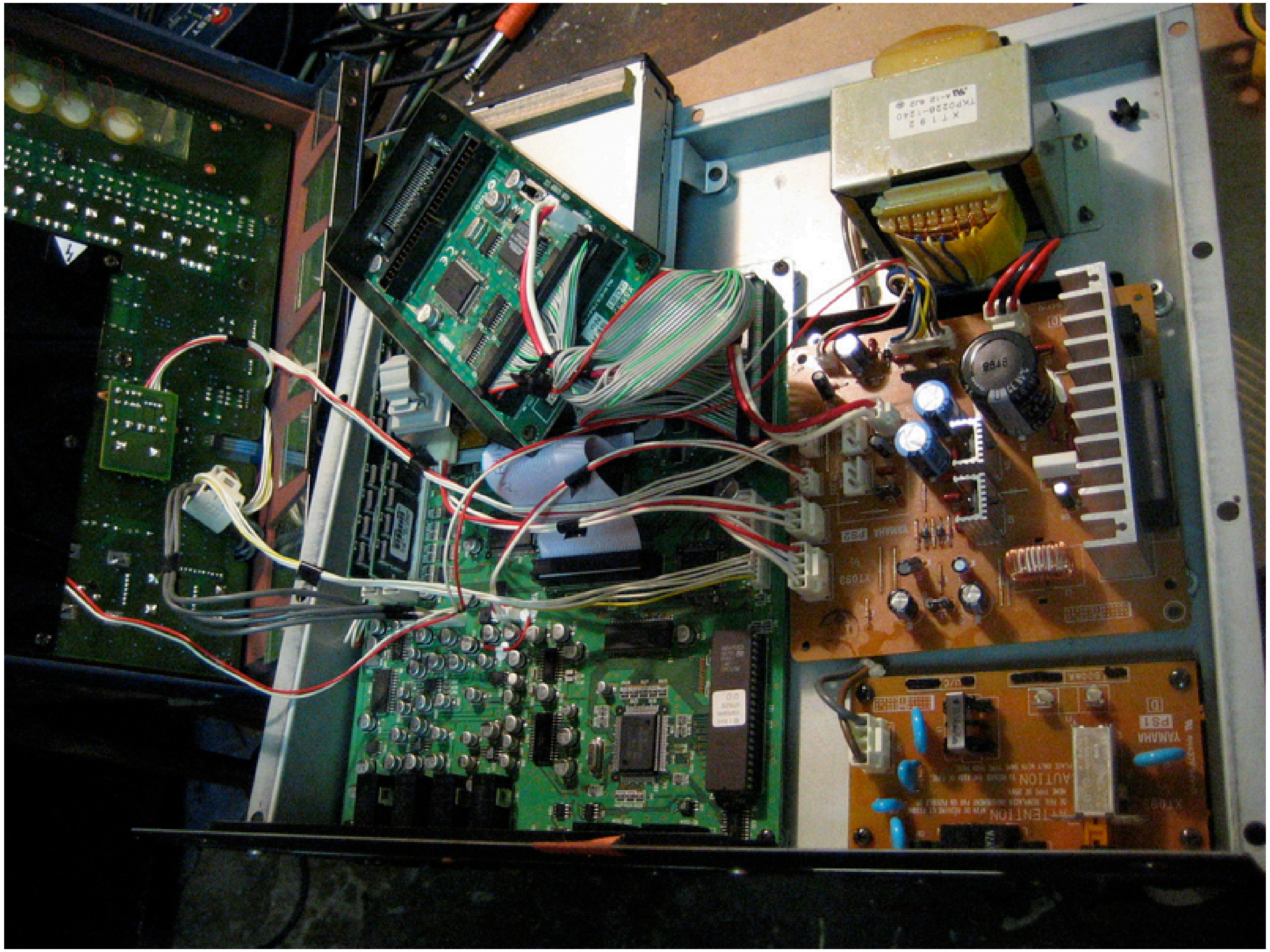
```
public class Foo {

  public void someMethod(String name) {
    ...
  }
}
```

Bean as a Message Translator with method name

```
from("activemq:Incoming”).
  beanRef("myBeanName", "someMethod").
    to("activemq:Outgoing");
```

Type Conversion



Type Conversion

```java
package com.acme.foo.converters;

import org.apache.camel.Converter;
import java.io.*;

@Converter
public class IOConverter {

    @Converter
    public static InputStream toInputStream(File file) throws FileNotFoundException {
        return new BufferedInputStream(new FileInputStream(file));
    }
}
```

```
# META-INF/services/org/apache/camel/TypeConverter

com.acme.foo.converters
```

Binding Beans to Camel Endpoints

```
public class Foo {

  @MessageDriven(uri="activemq:cheese")
  public void onCheese(String name) {
    ...
  }
}
```

Binding Method Arguments

```
public class Foo {

  public void onCheese(
    @XPath("/foo/bar") String name,
    @Header("JMSCorrelationID") String id) {
    ...
  }
}
```

for more annotations see
http://activemq.apache.org/camel/bean-integration.html

Injecting endpoints into beans

```
public class Foo {
  @EndpointInject(uri="activemq:foo.bar")
  ProducerTemplate producer;

  public void doSomething() {
    if (whatever) {
      producer.sendBody("<hello>world!</hello>");
    }
  }
}
```

Spring Remoting - Client Side

```
<camelContext xmlns="http://activemq.apache.org/camel/schema/spring">
  <proxy id="sayService" serviceUrl="activemq:MyService"
    serviceInterface="com.acme.MyServiceInterface"/>
</camelContext>
```

Spring Remoting - Server Side

```
<camelContext xmlns="http://activemq.apache.org/camel/schema/spring">
  <export id="sayService" uri="activemq:MyService" serviceRef="sayImpl"
    serviceInterface="com.acme.MyServiceInterface"/>
</camelContext>

<bean id="sayImpl" class="com.acme.MyServiceImpl"/>
```

Spring Framework

Dependency Injection

```
<camelContext xmlns="http://activemq.apache.org/camel/schema/spring">
  ...
</camelContext>

<bean id="activemq" class="org.apache.camel.component.jms.JmsComponent">
  <property name="connectionFactory">
    <bean class="org.apache.activemq.ActiveMQConnectionFactory">
      <property name="brokerURL" value="vm://localhost?broker.persistent=false"/>
    </bean>
  </property>
</bean>
```

Data Format

```
from("activemq:QueueWithJavaObjects).
  marshal().jaxb().
    to("mqseries:QueueWithXmlMessages");
```

Business Activity Monitoring (BAM)

Business Activity Monitoring (BAM)

```
public class MyActivities extends ProcessBuilder {

    public void configure() throws Exception {

        // lets define some activities, correlating on an XPath on the message bodies
        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");
    }
}
```

Riding the camel

Where would I use Camel?

- standalone or in any Spring application
- inside ActiveMQ’s JMS client or the broker
- inside your ESB such as ServiceMix via the servicemix-camel Service Unit
- inside CXF either as a transport or reusing CXF inside Camel

Camel Riding from Java

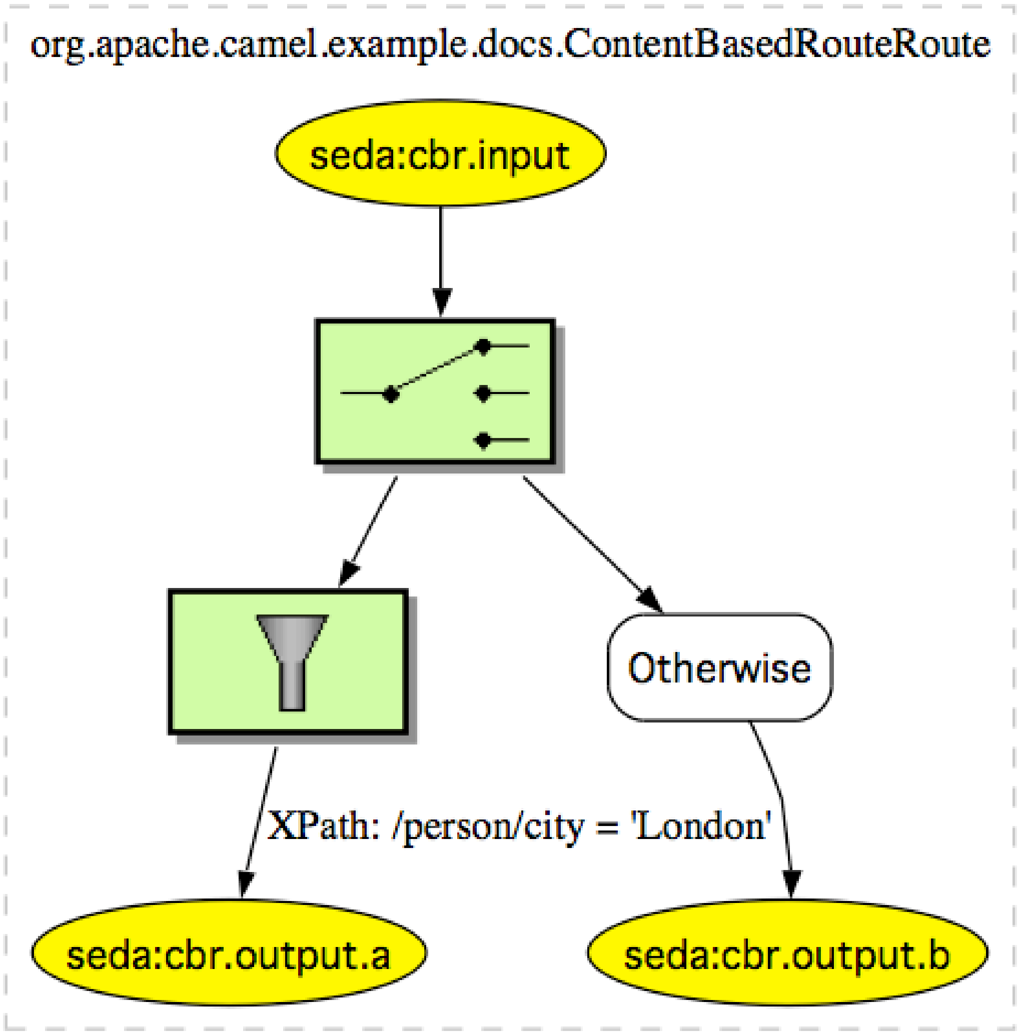
- /META-INF/spring/camelContext.xml
- set the CLASSPATH
- java org.apache.camel.spring.Main

Maven Tooling

```
<project>
...
   <build>
   <plugins>
      <plugin>
        <groupId>org.apache.camel</groupId>
        <artifactId>camel-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </build>
  <reporting>
    <plugins>
      <plugin>
        <groupId>org.apache.camel</groupId>
        <artifactId>camel-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </reporting>
</project>
```

mvn camel:run

Maven Plugin Site Report

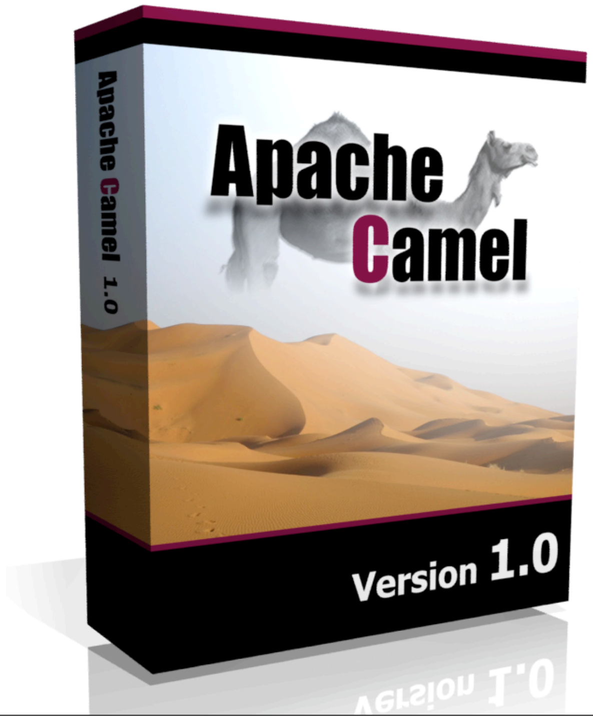

org.apache.camel.example.docs.ContentBasedRouteRoute
seda:cbr.input
Otherwise
XPath: /person/city = 'London'
seda:cbr.output.a
seda:cbr.output.b

Where do I get more info?

please do take Camel for a ride!

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

don’t get the hump! :-)



Questions?

James Strachan

blog

http://macstrac.blogspot.com/

Camel v Mule

- A Camel can carry 4 times as much load as other beasts of burden!

  http://activemq.apache.org/camel/why-the-name-camel.html

Camel v Mule

- Camel provides a higher level abstraction for EIP; making it simpler & easier & less XML
- Awesome integration with JMS + ActiveMQ (client and broker), JBI + ServiceMix and JAX-WS + CXF
- great testing with Camel's Mock Endpoint
- Business Activity Monitoring framework

Where do I get more info?

please do take Camel for a ride!

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

don’t get the hump! :-)