Enterprise Integration with Spring Integration

Agim Emruli
SpringSource
8100

JAZOON09
THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
JUNE 22 – 25, 2009 ZURICH

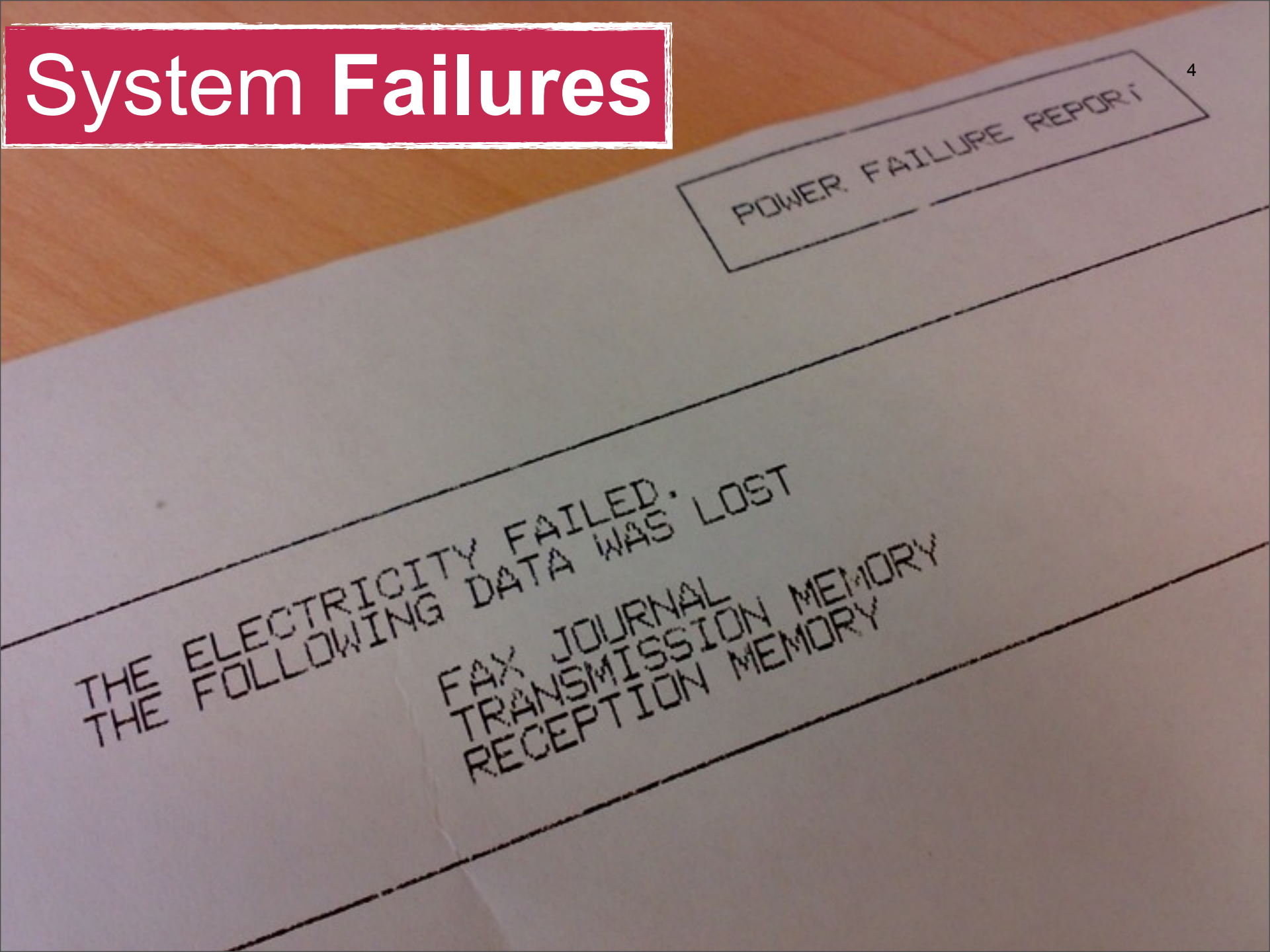
netcetera
Sun microsystems

Presentation Goal

Learn how **Spring Integration** helps to solve common **Enterprise Integration** challenges

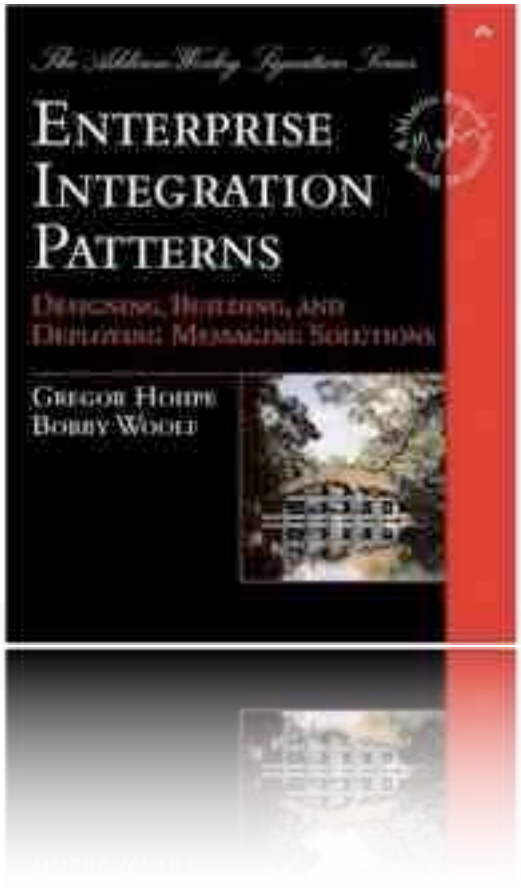
Integration **Challenges**

System **Failures**

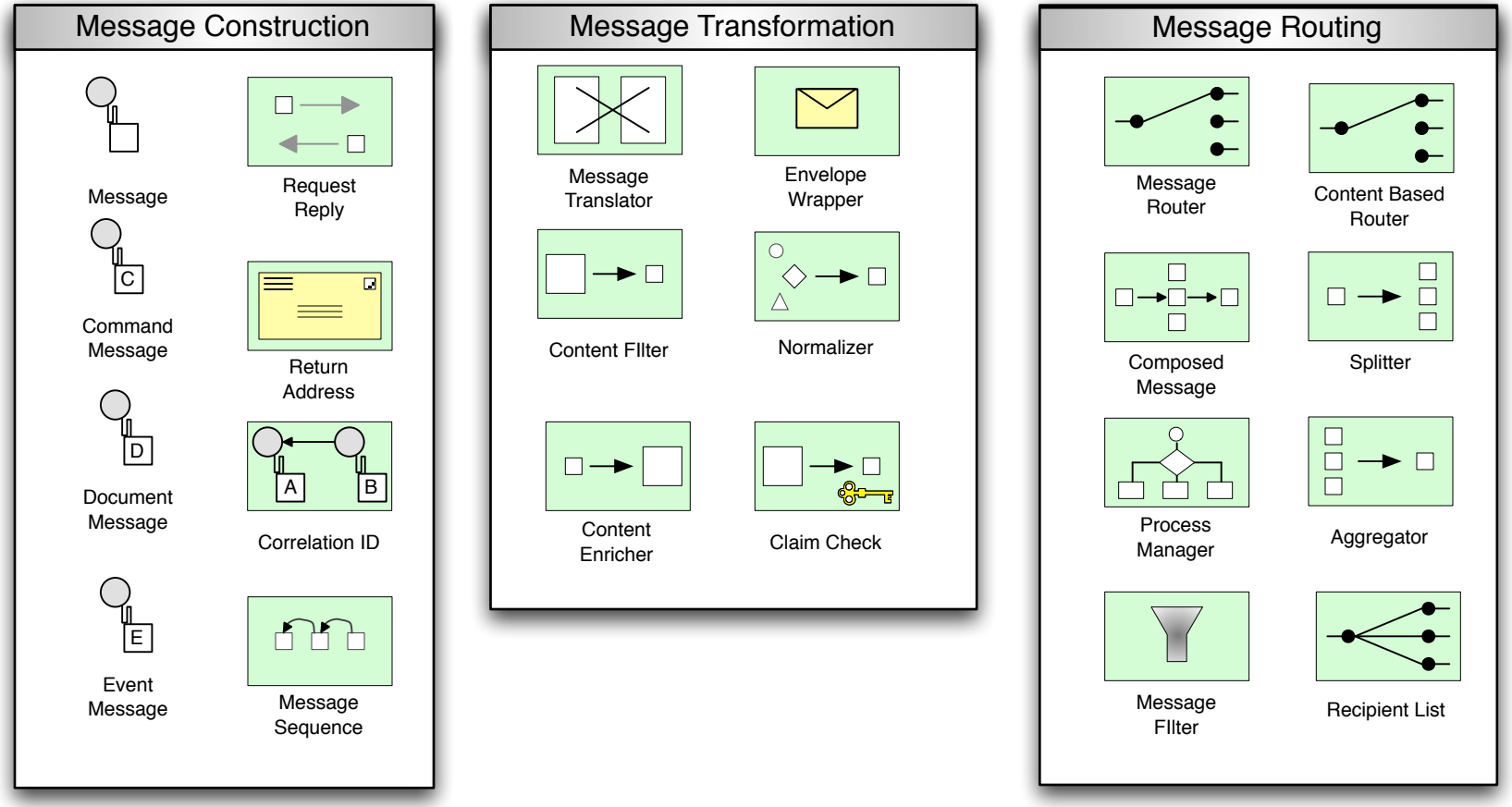


Intolerant Systems

Enterprise **Integration** Patterns

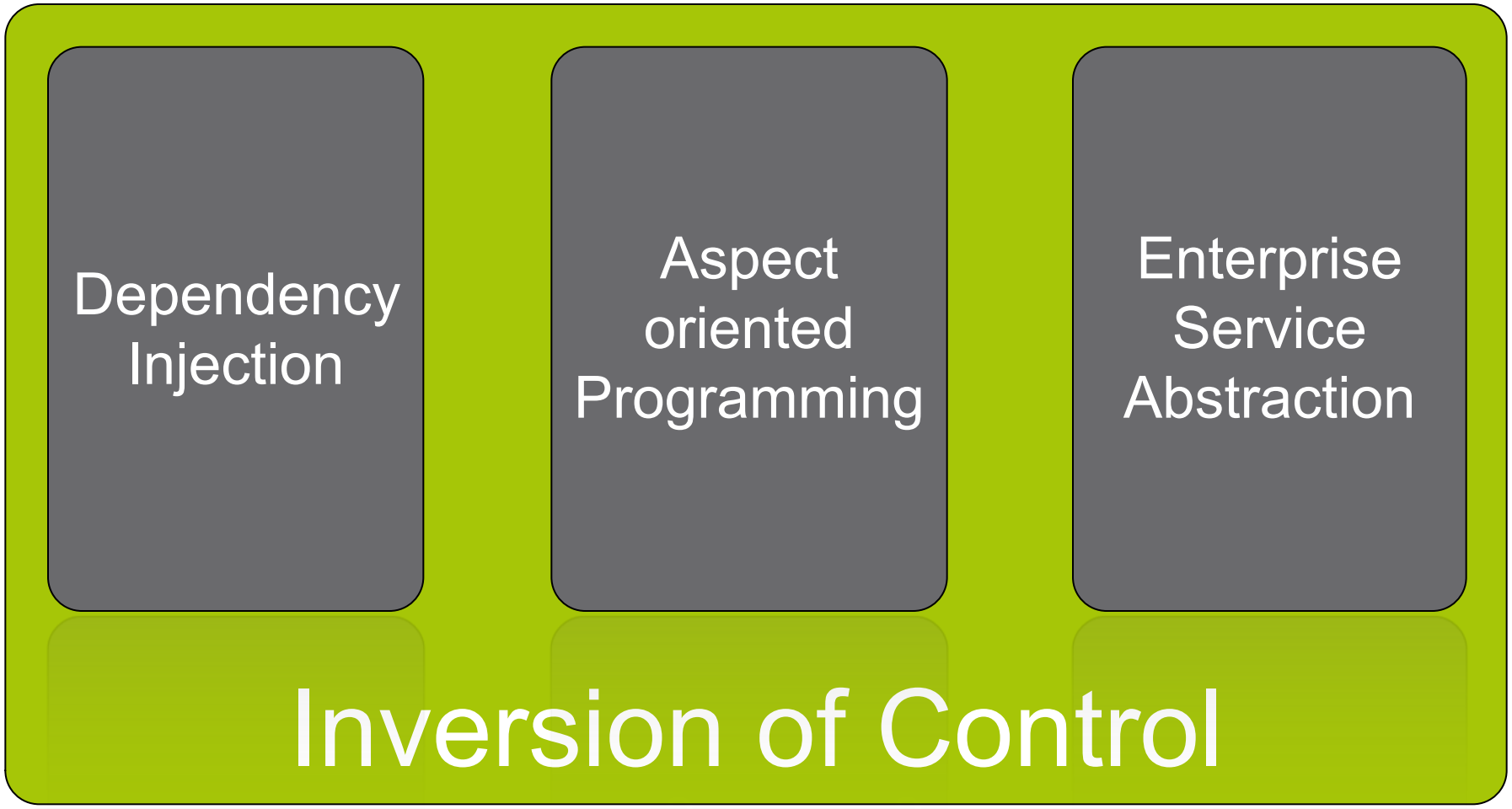


Pattern Catalog Overview



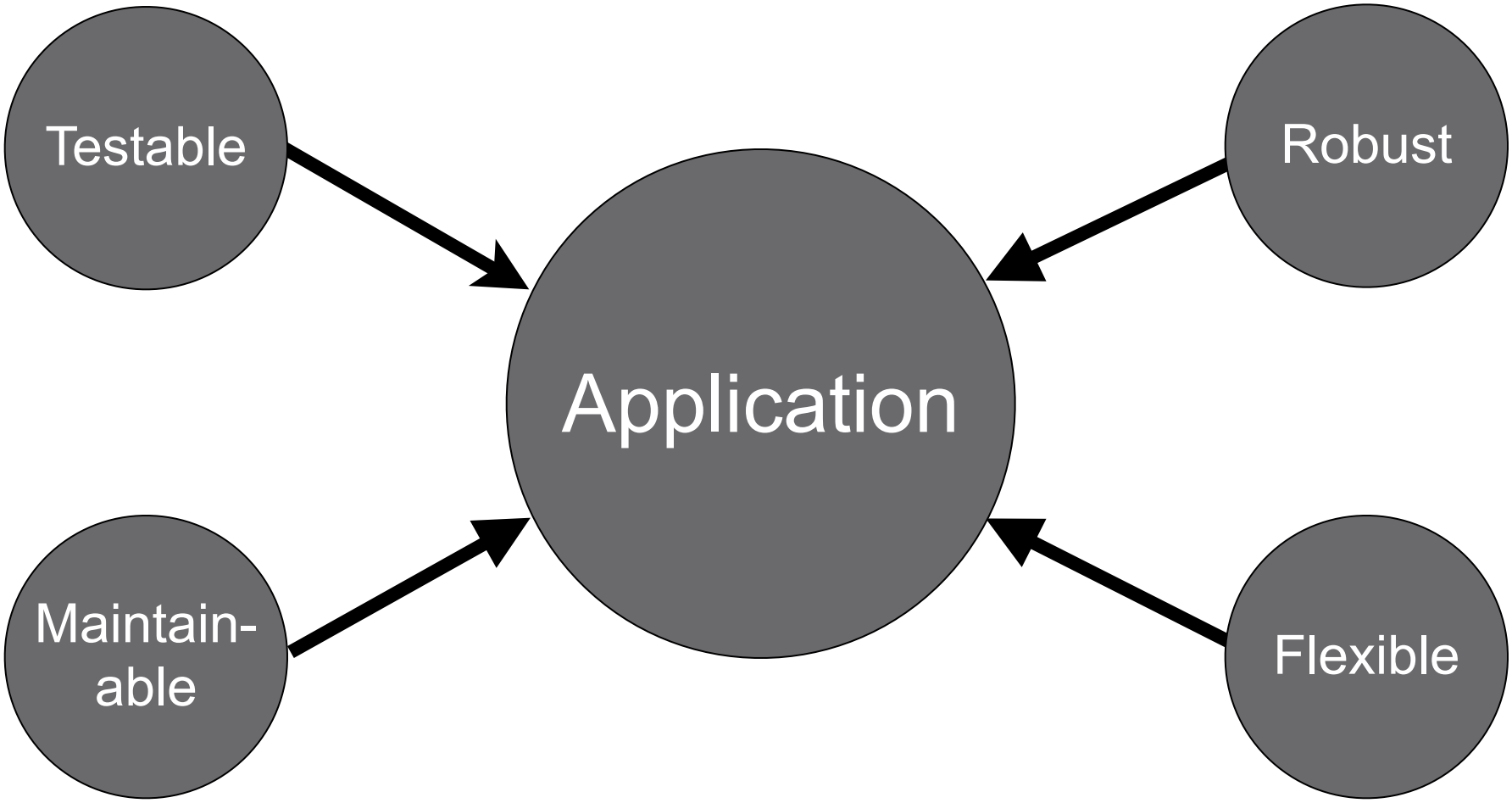
How does Spring **help** ?

Spring Big Picture



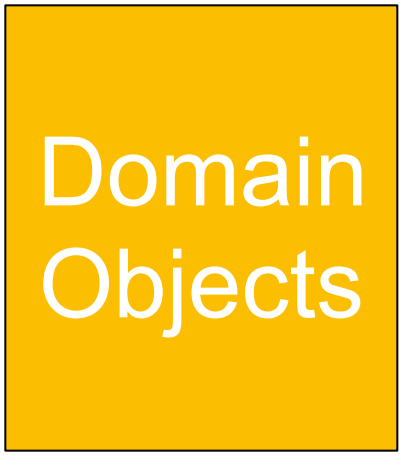
Focus on **Business** domain

Spring Powered Applications



Layered Architecture

Layered Architecture



Layered Architecture

Domain Objects

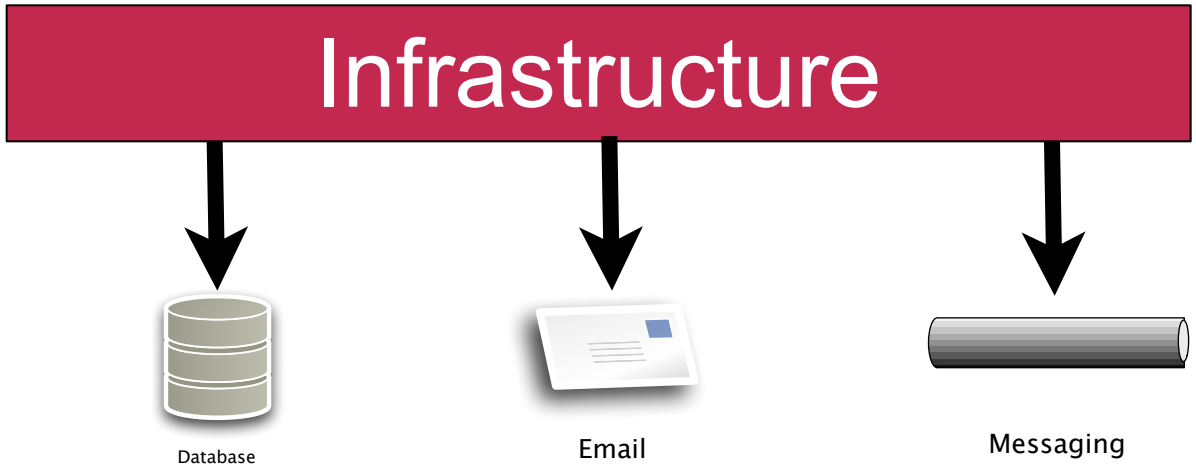
Database

Email

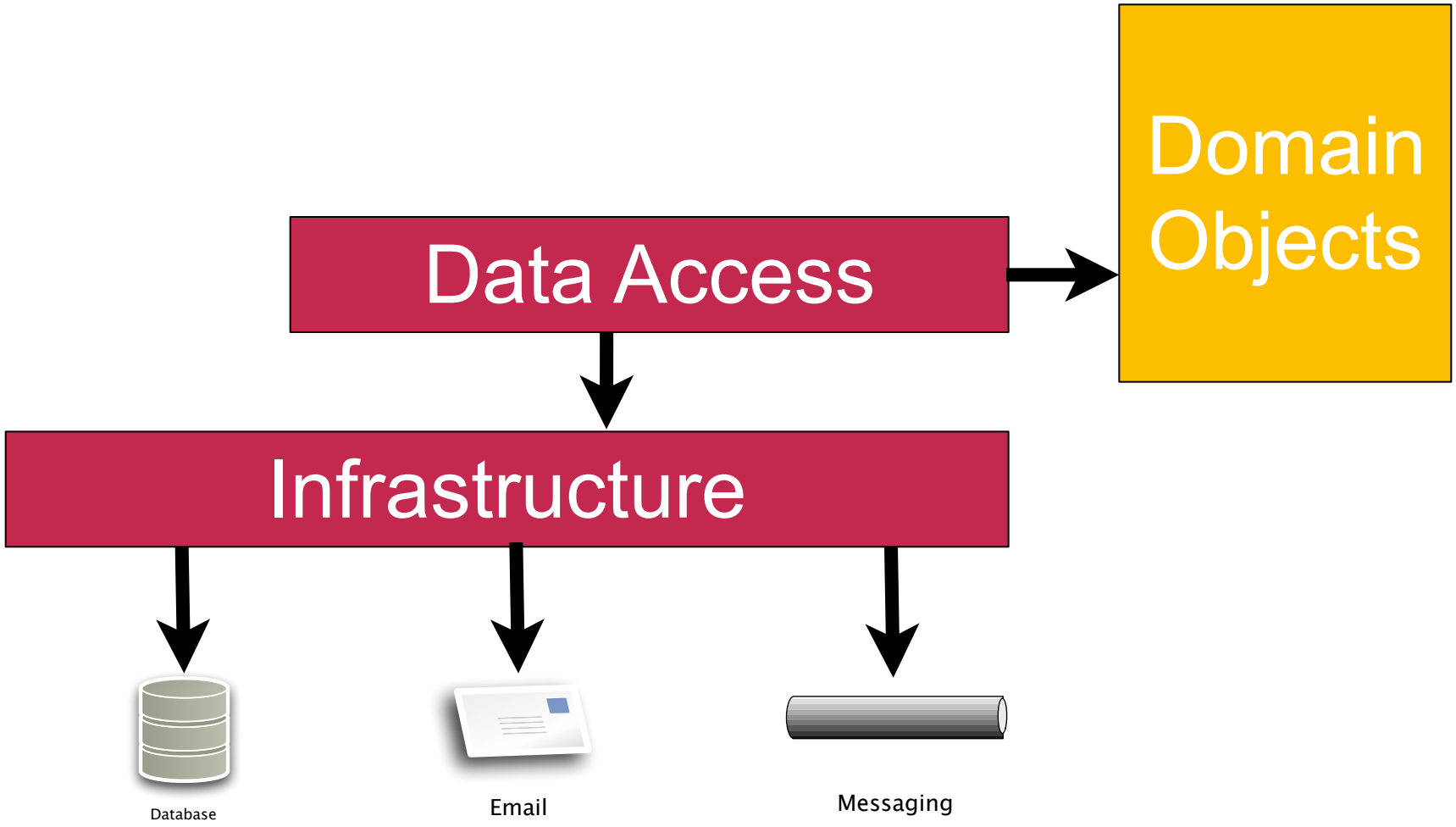
Messaging

Layered Architecture

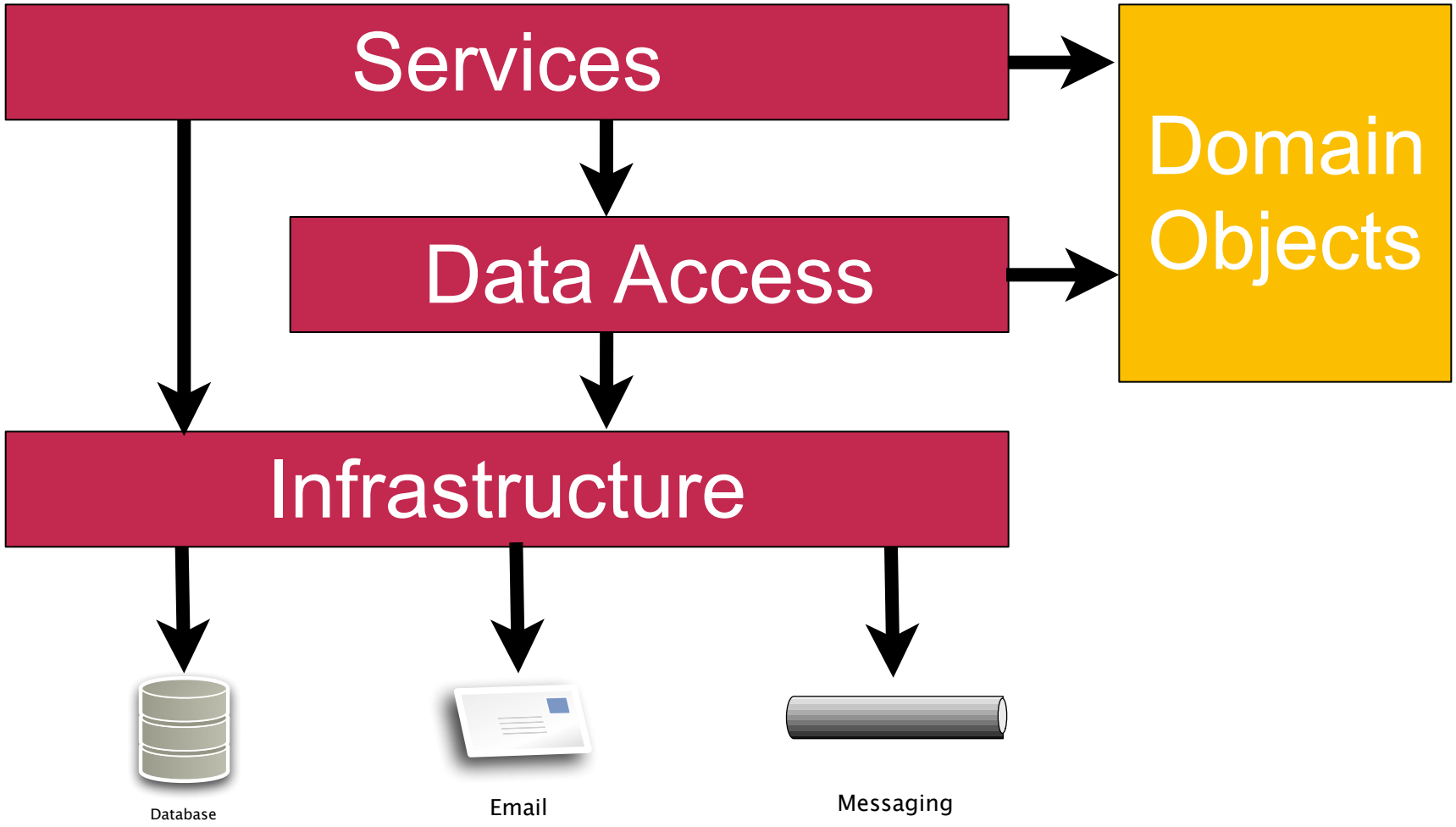
Domain Objects



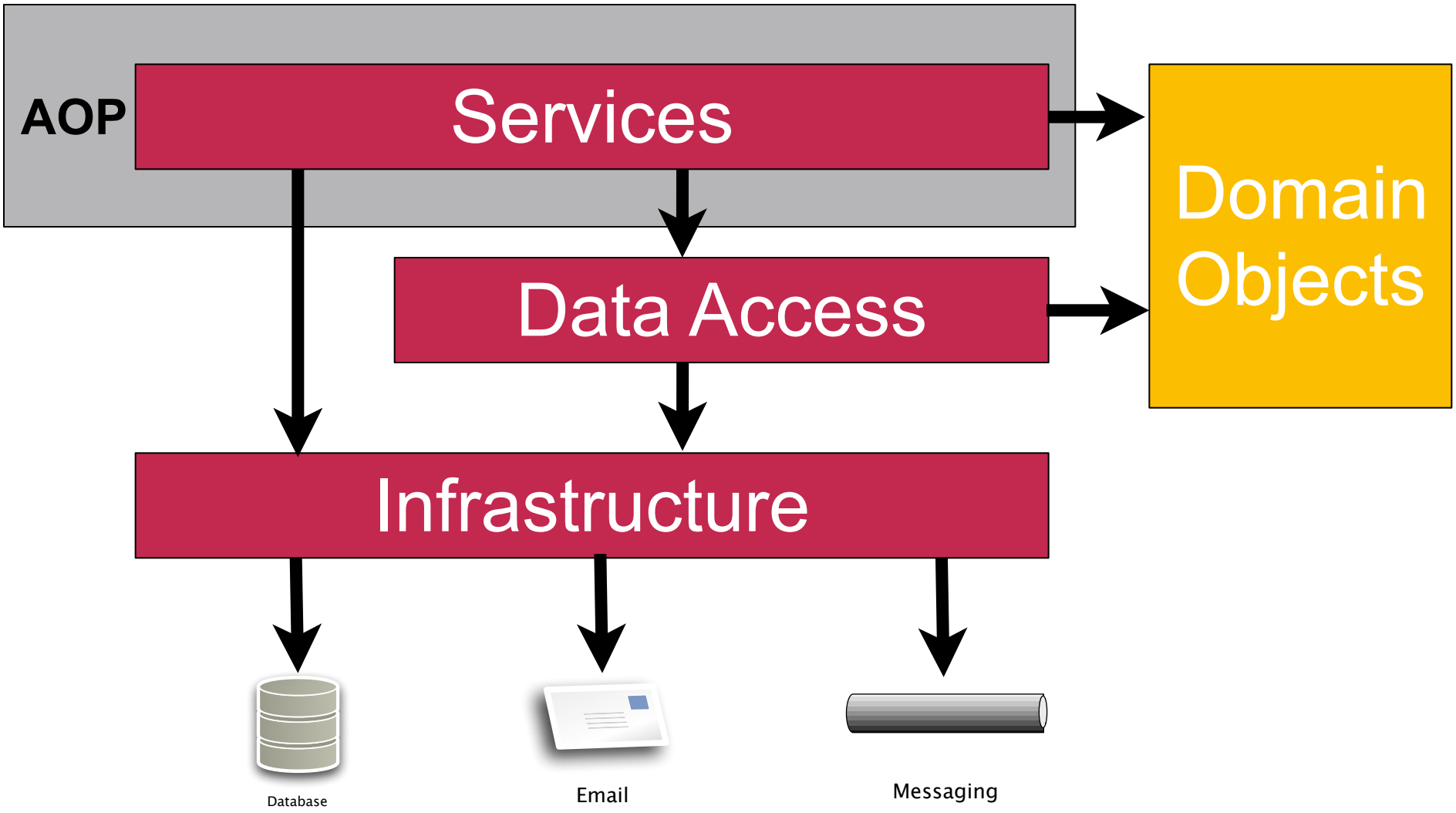
Layered Architecture



Layered Architecture

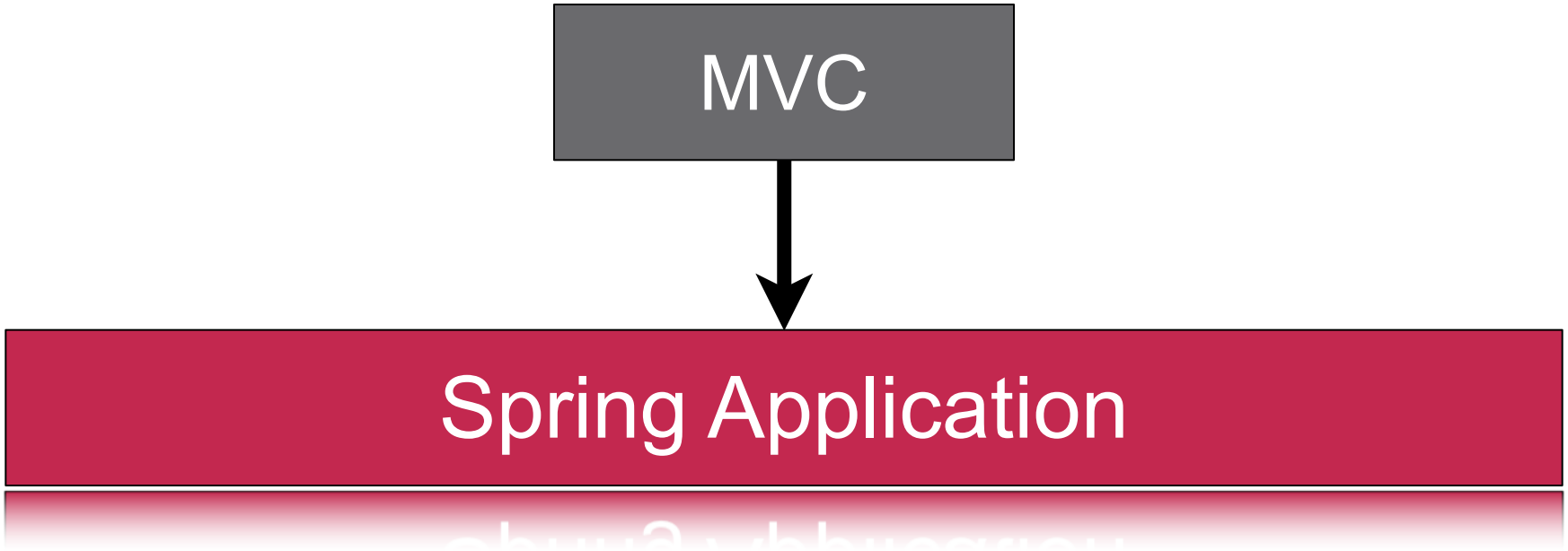


Layered Architecture

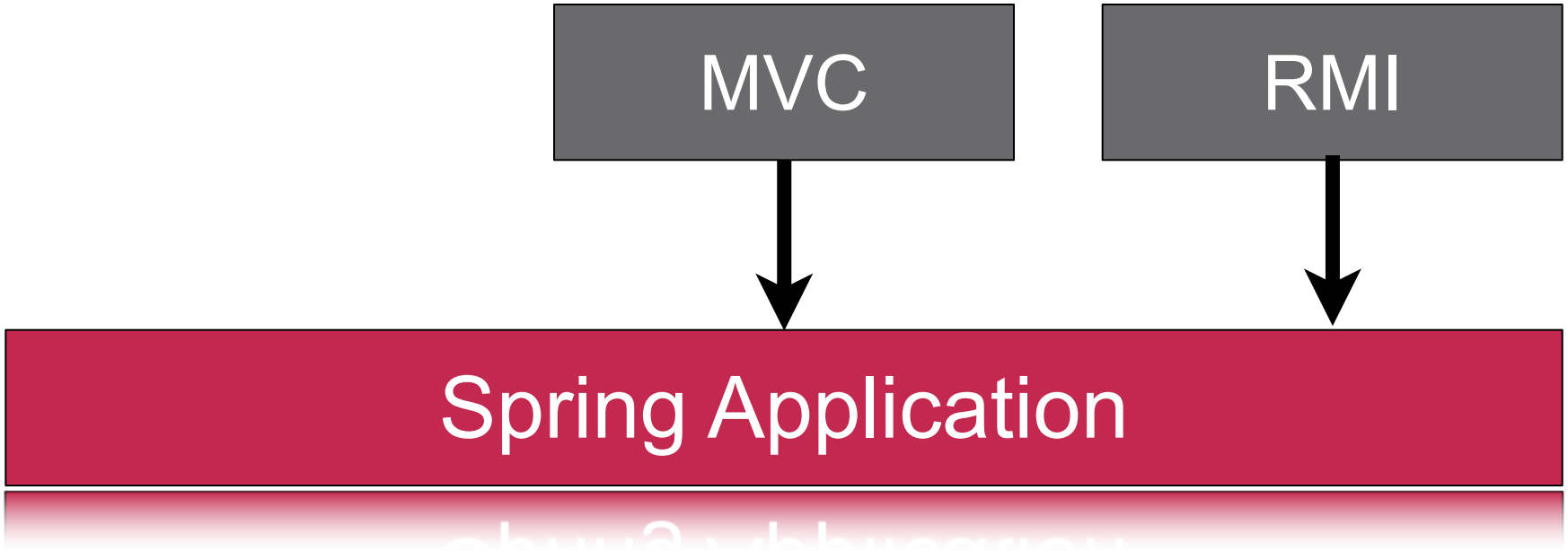


Layered Architecture

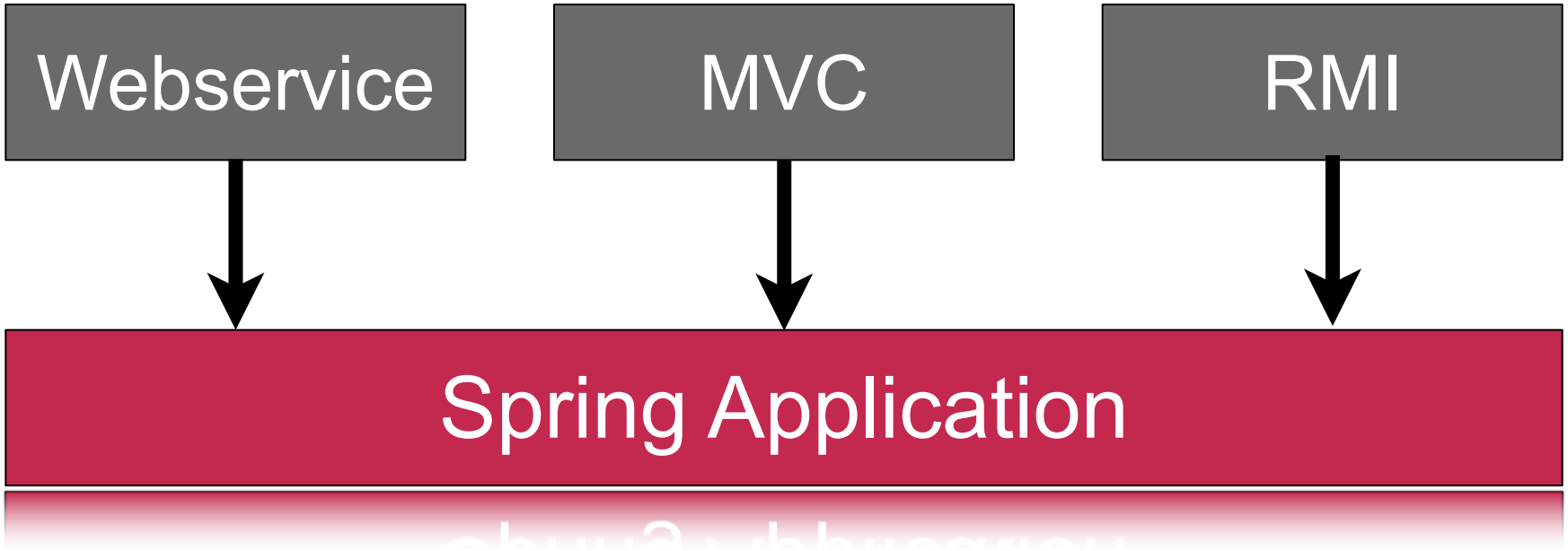
Layered Architecture



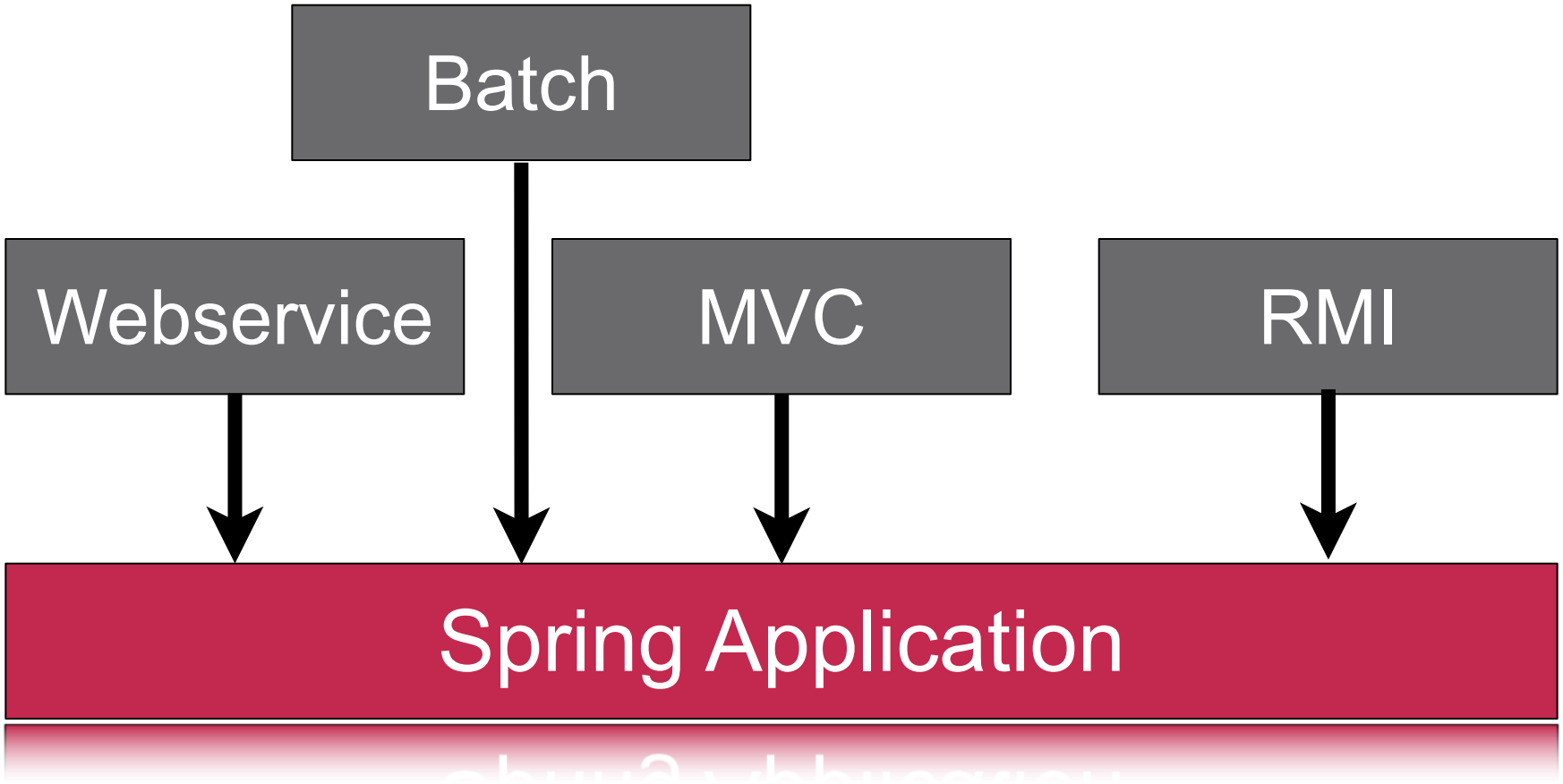
Layered Architecture



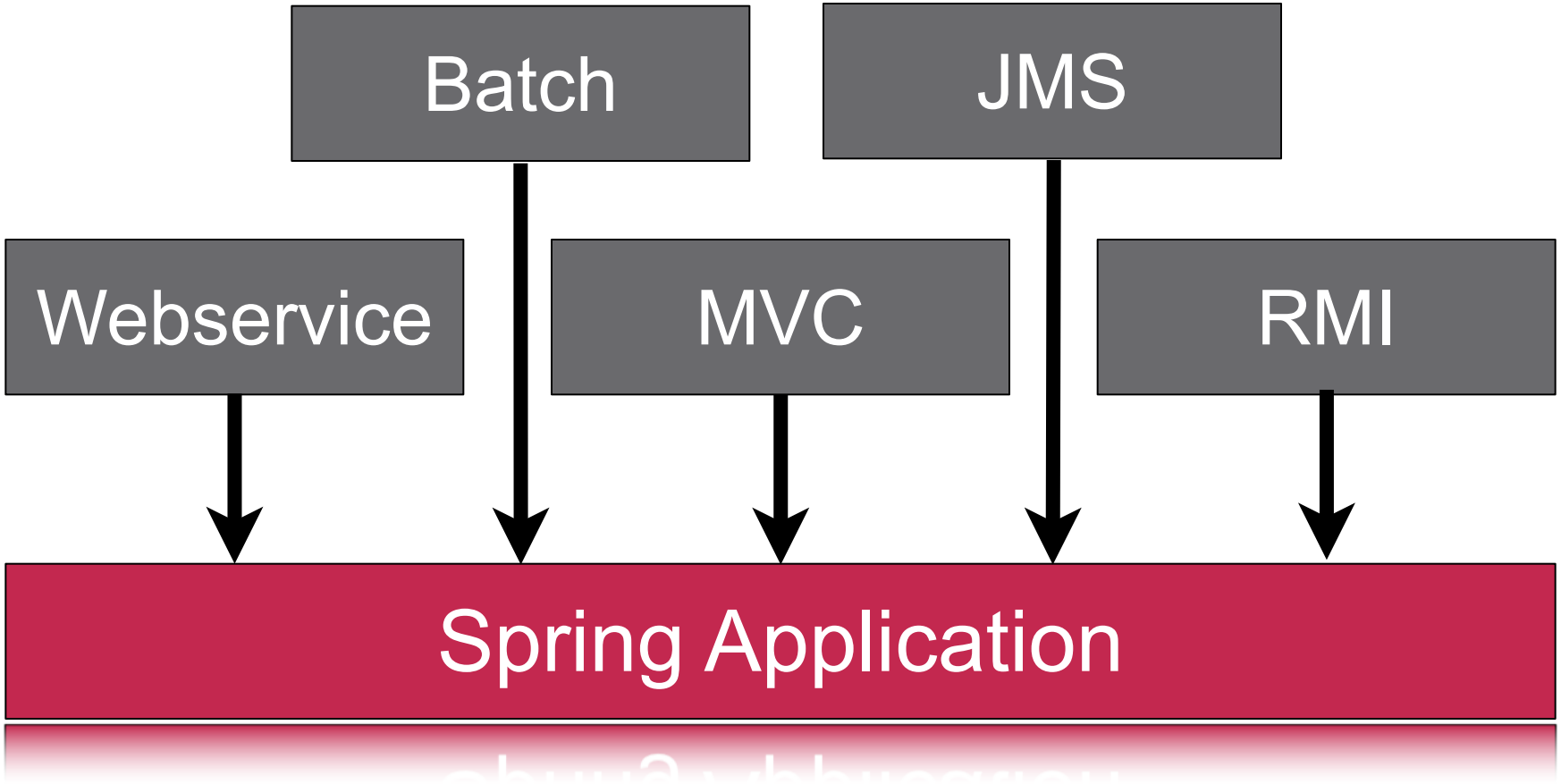
Layered Architecture



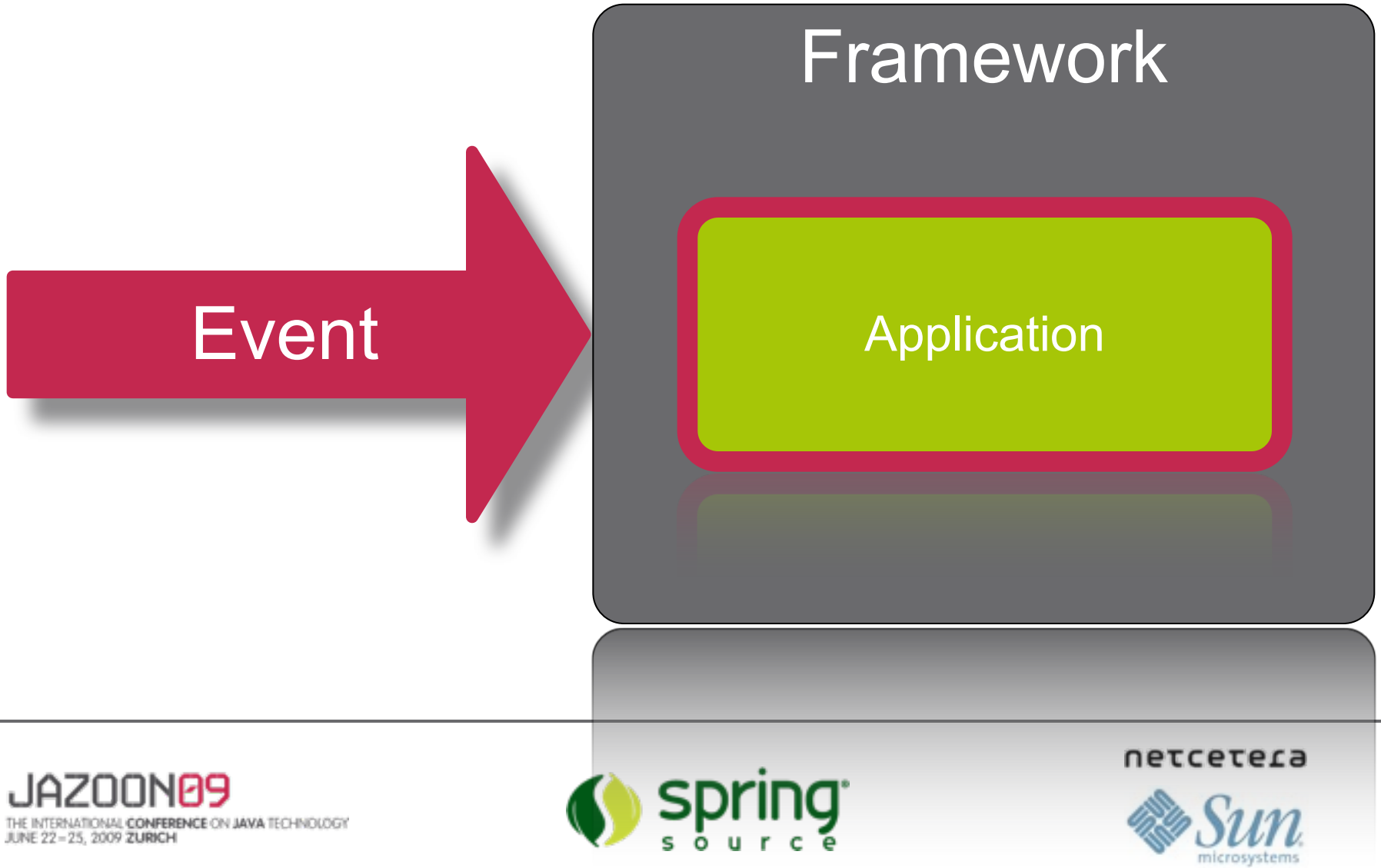
Layered Architecture



Layered Architecture



Event Driven Architecture



Spring JMS Support

```
<jms:listener-container transaction-manager="txManager">
    <jms:listener ref="orderService"
                  method="order"
                  destination="queue.orders"
                  response-destination="queue.confirmation"/>
</jms:listener-container>
```

Spring JMS Support

```
<jms:listener-container transaction-manager="txManager">
    <jms:listener ref="orderService"
                  method="order"
                  destination="queue.orders"
                  response-destination="queue.confirmation"/>
</jms:listener-container>
```

```
public class OrderService {

    public OrderConfirmation order(Order o) {..}

}
```

Introducing **Spring Integration**

JAZOON09
THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
JUNE 22 – 25, 2009 ZURICH

spring source

netcetera

Sun microsystems


Goals

Reuse Service Layer

Incremental Extension

Spring Integration Architecture

Spring Integration Architecture

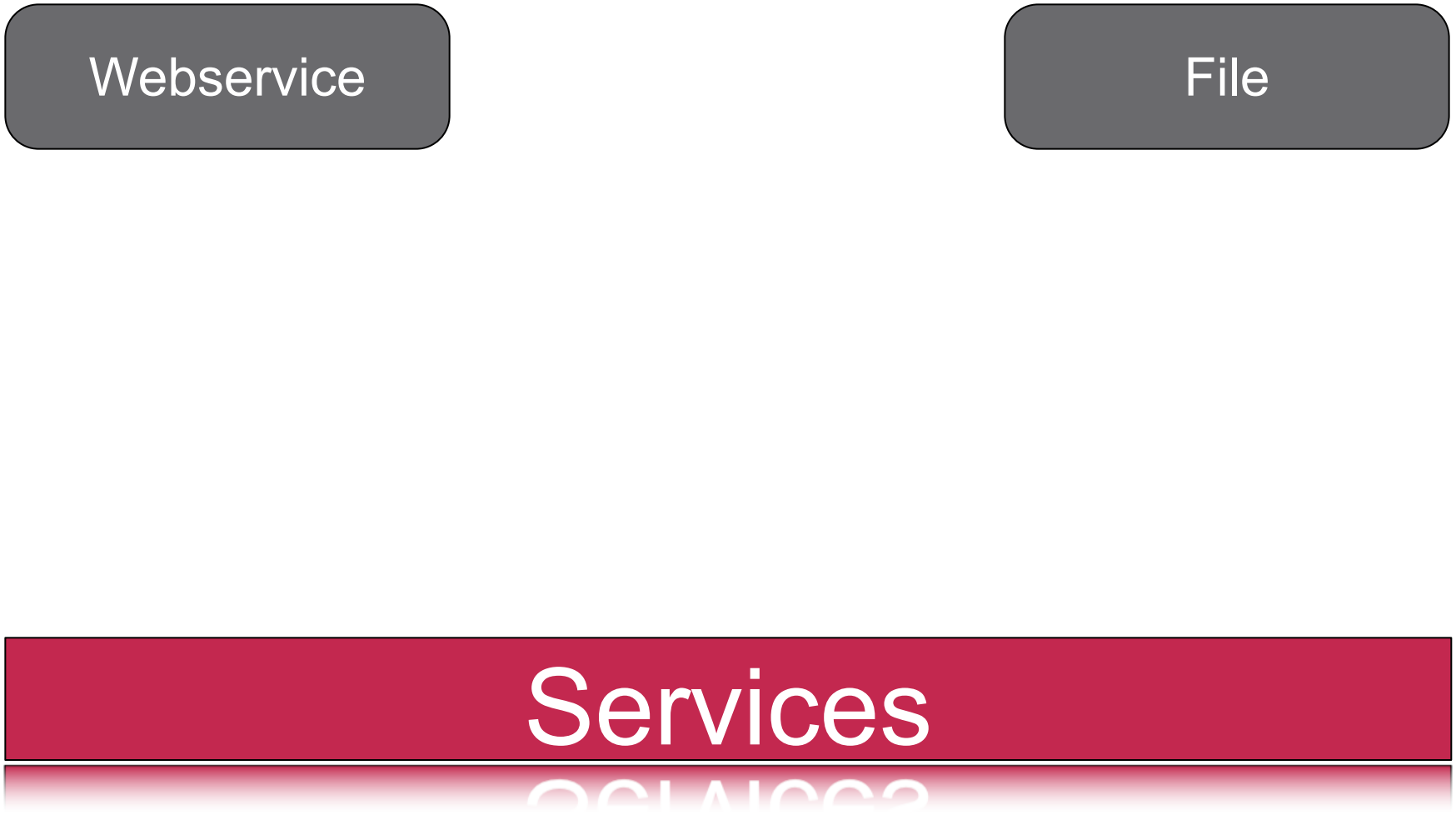
Services

Spring Integration Architecture

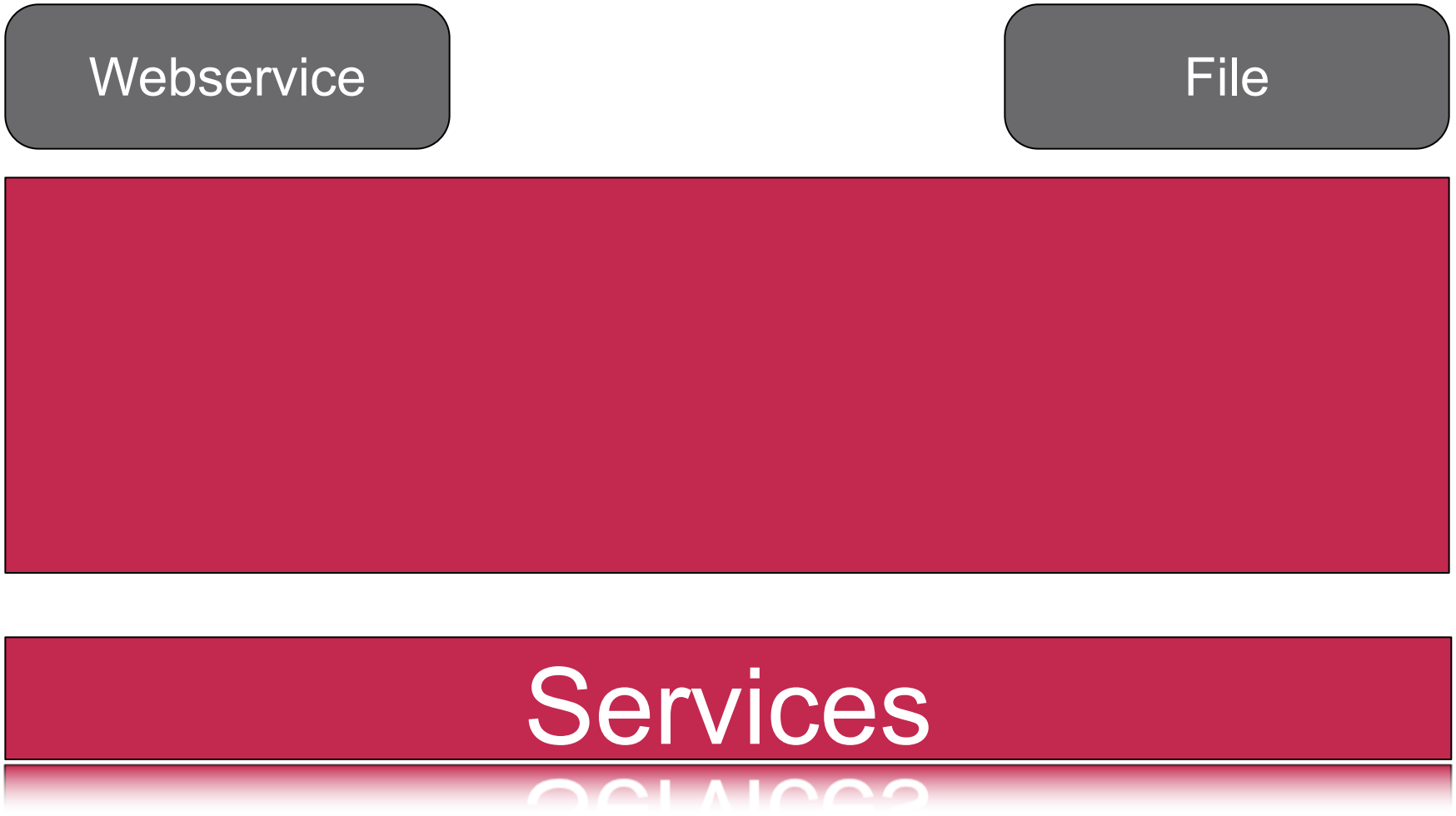
Webservice

Services

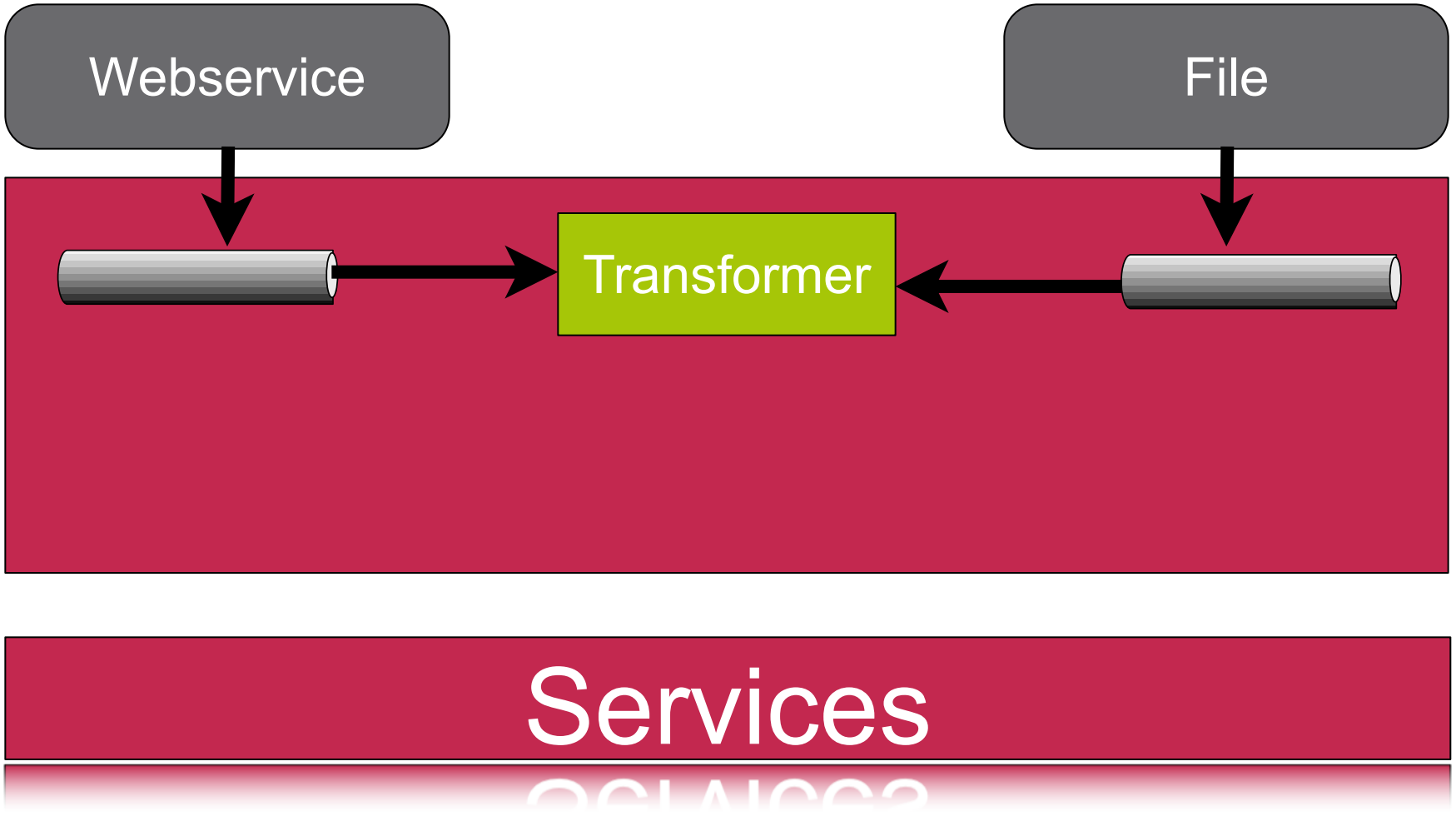
Spring Integration Architecture



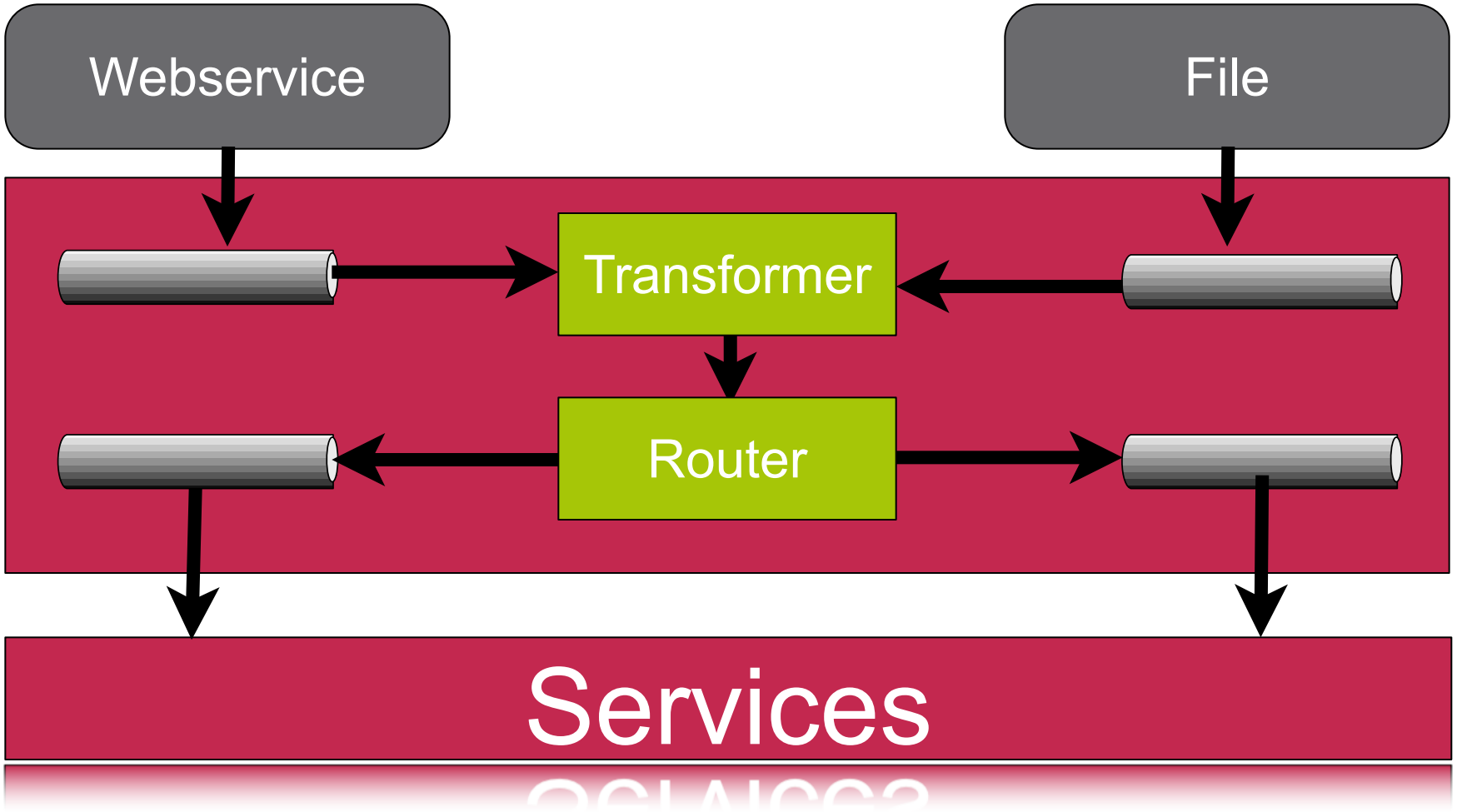
Spring Integration Architecture



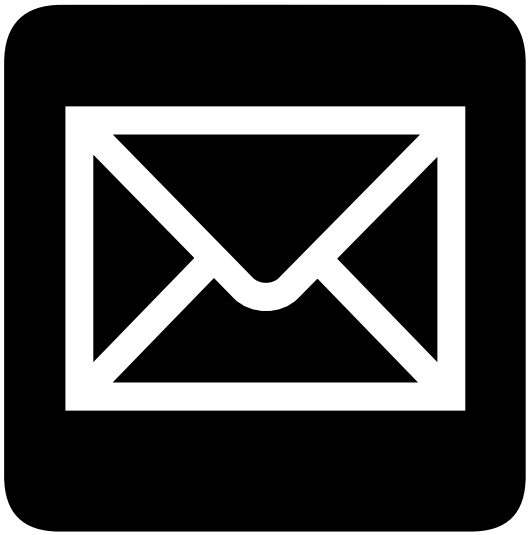
Spring Integration Architecture



Spring Integration Architecture



Message Construction

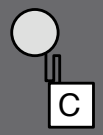


Message Structure

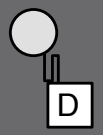
Message Header

Sequence Number
Sequence Size
Expiration Date
Correlation Identifier

Message Body

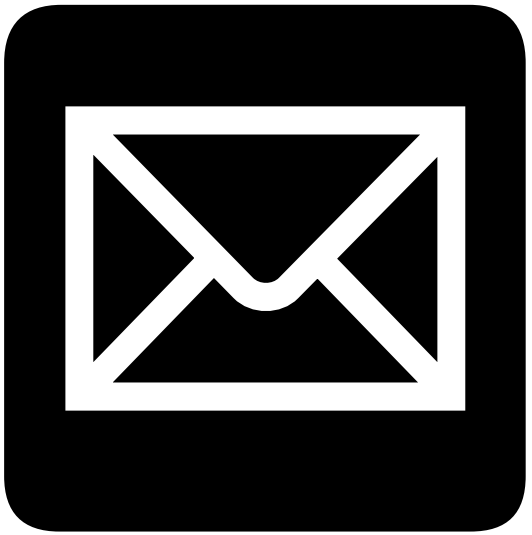


Command Message



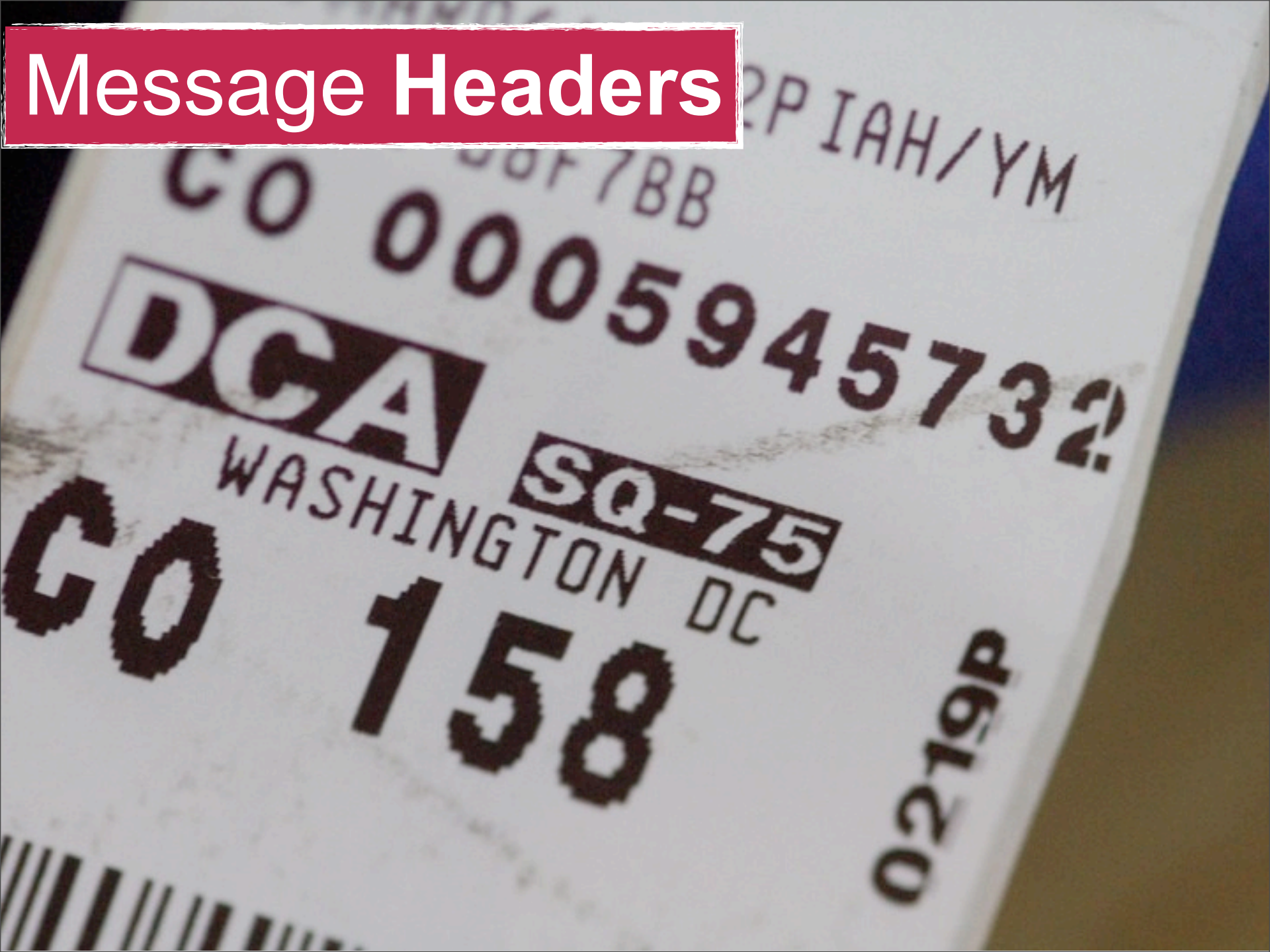
Document Message

Event Message



Message Interface

```
public interface Message<T> {

  MessageHeaders getHeaders();

  T getPayload();
}
```


Message Headers
CO 0005945732
DCA
WASHINGTON DC
SQ-75
CO 158
0219P

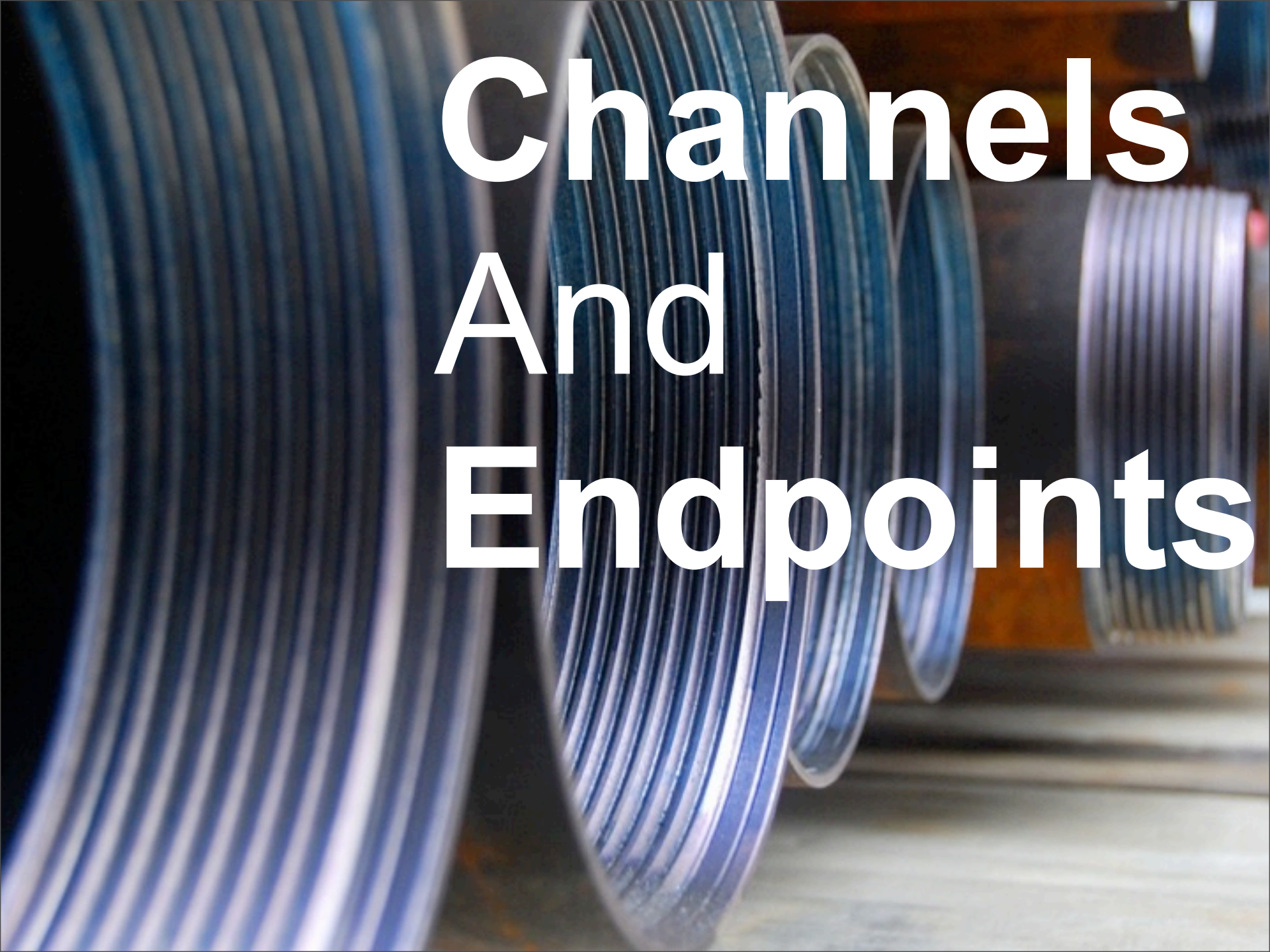
Message Headers


```
MessageHeaders headers = message.getHeaders();

String value =
headers.get("key", String.class);

Object id = headers.getId();

Long timestamp = headers.getTimestamp();

MessagePriority priority =
headers.getPriority();
```

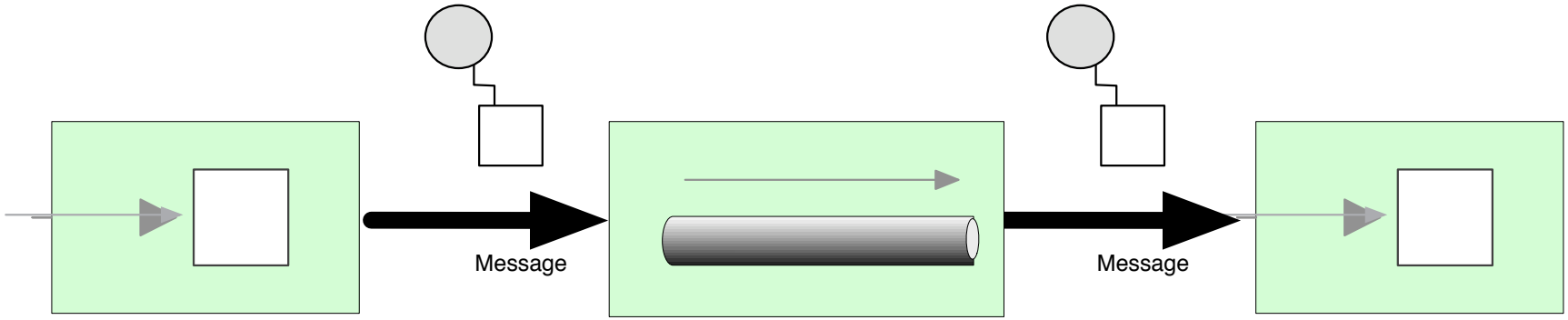
Message **Builder**

Message Builder

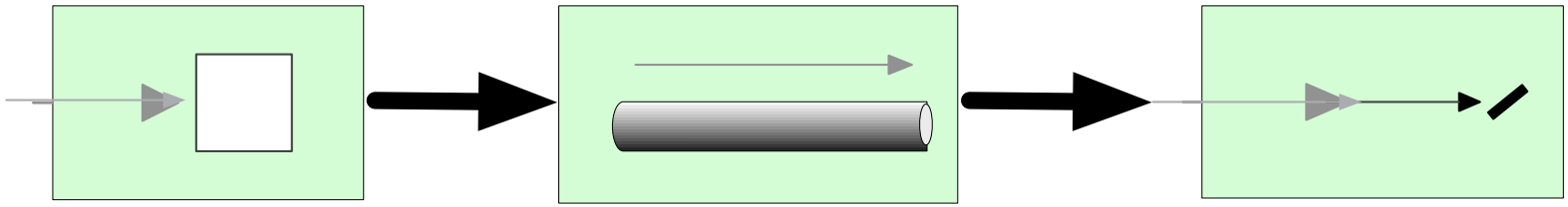

```
Message<String> message =
MessageBuilder.withPayload("test")
    .setHeader("foo", 123)
    .setPriority(MessagePriority.HIGHEST)
    .build();


Message<String> copy =
MessageBuilder.fromMessage(message)
    .setHeader("foo", 456)
    .setHeaderIfAbsent("bar", 789)
    .build();
```


Channels
And
Endpoints

Message Channel



Direct Channels

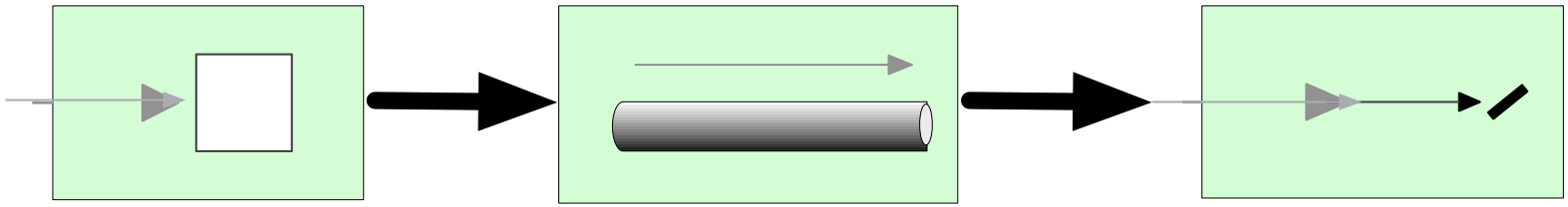


Message Endpoint

Channel

Event-Driven Consumer

Direct Channels



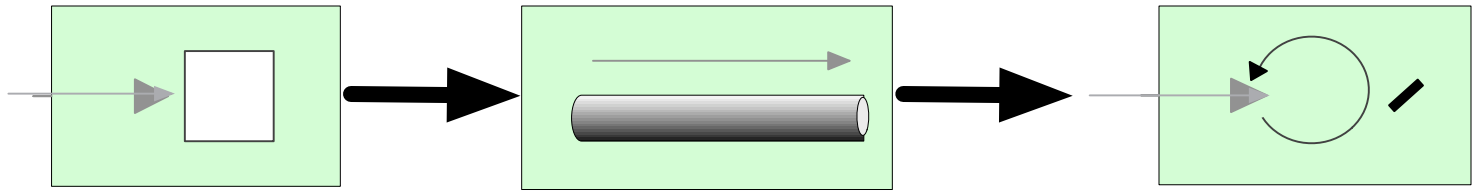
Message Endpoint

Channel

Event-Driven Consumer

```
<channel id="sync-p2p" />
```

Queue Channel

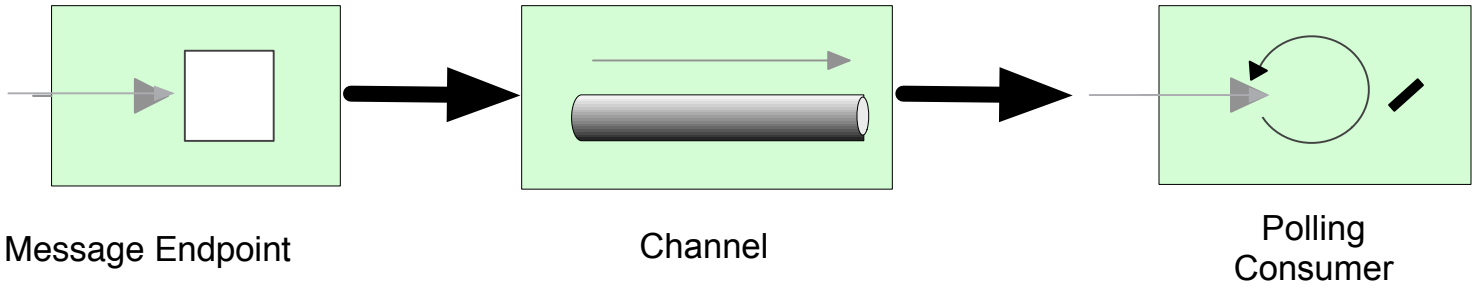


Message Endpoint

Channel

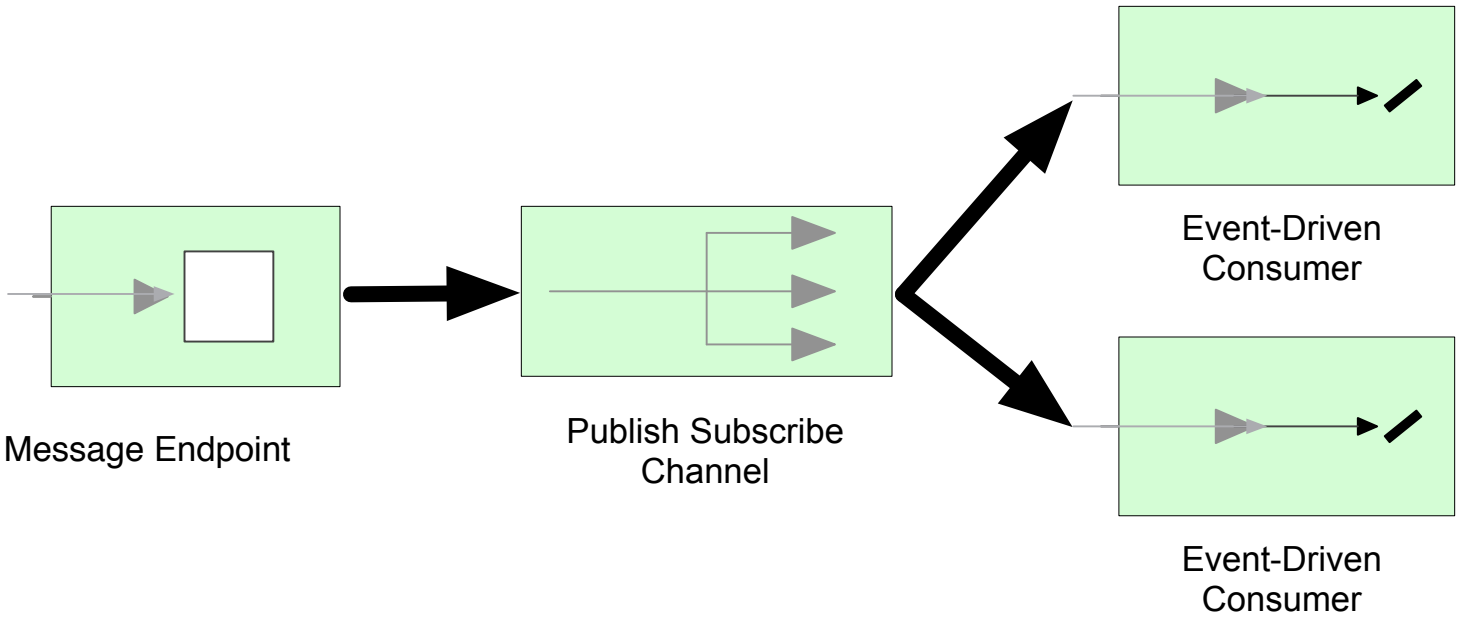
Polling Consumer

Queue Channel

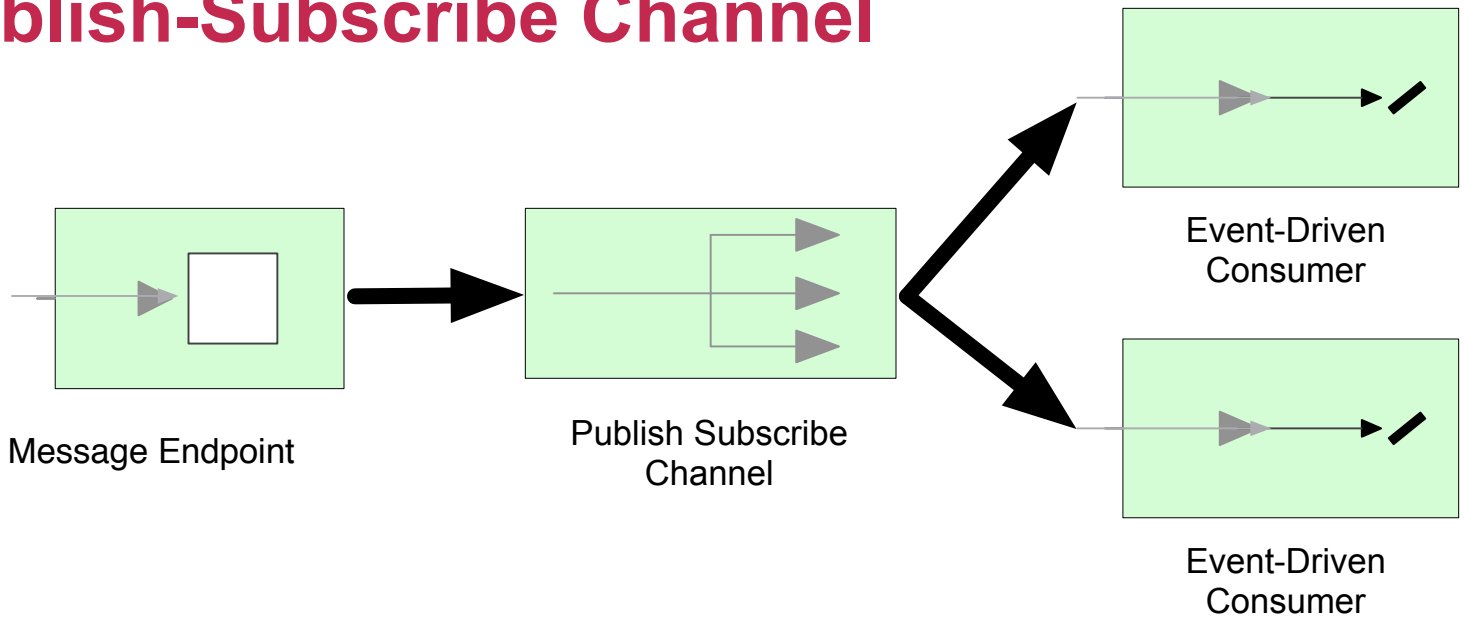



```
<channel id="async-p2p">
  <queue capacity="50" />
</channel>
```

Publish-Subscribe Channel

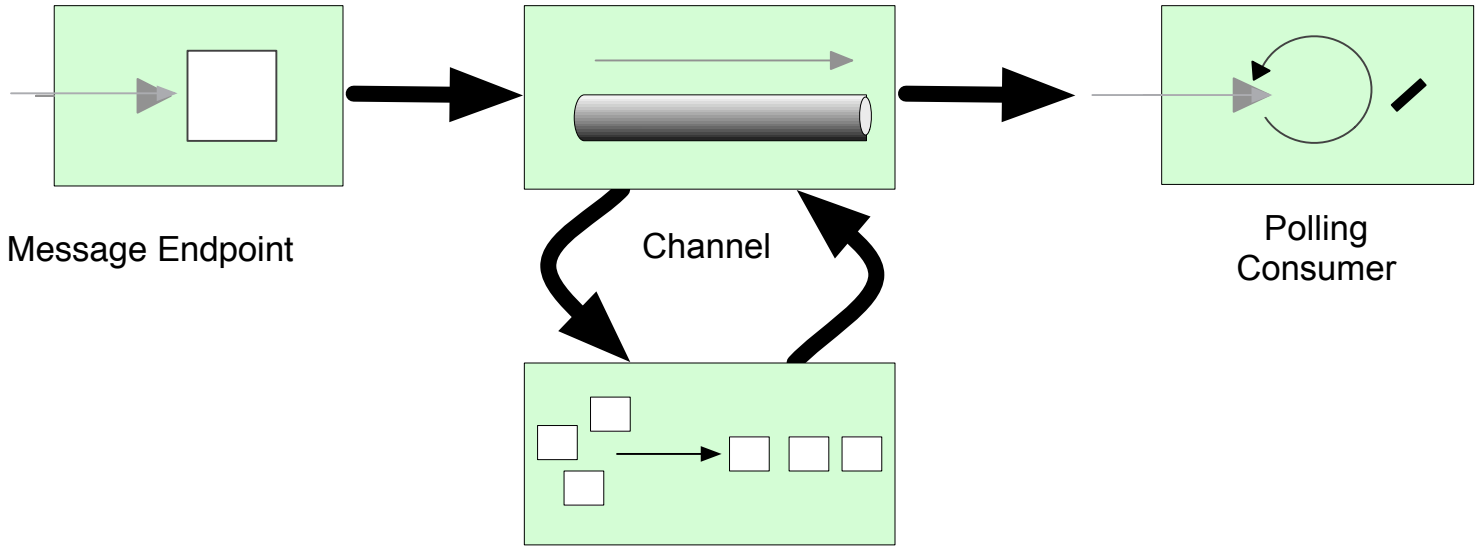


Publish-Subscribe Channel

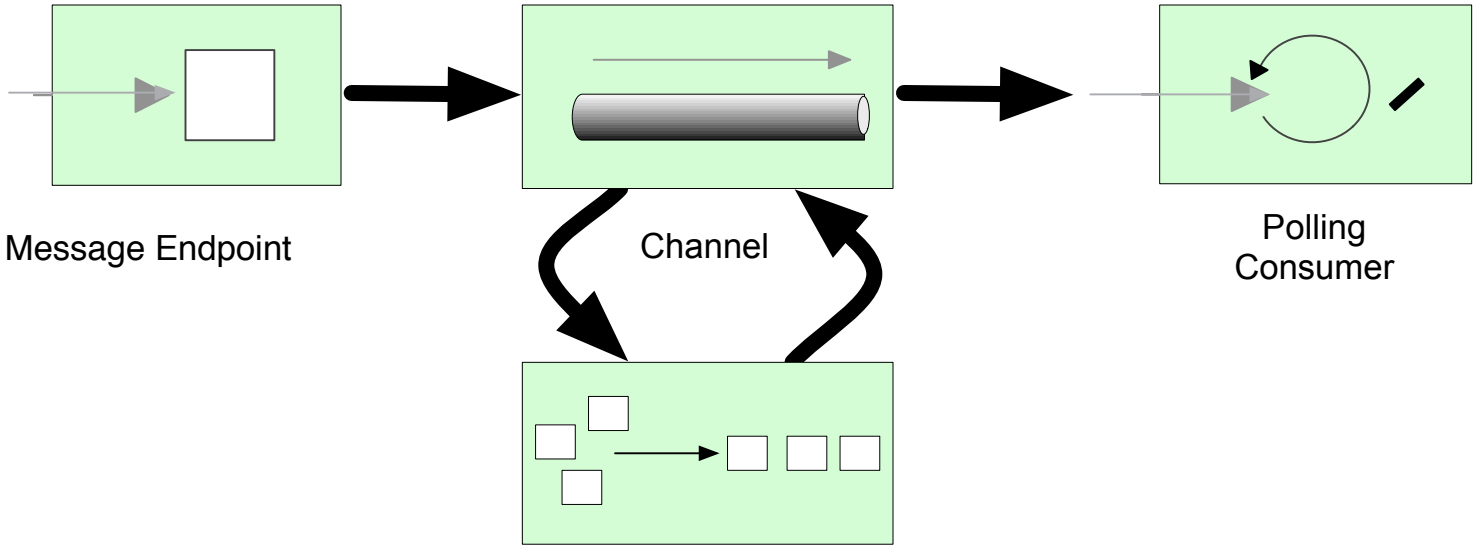



```
<publish-subscribe-channel id="pubsub" />
```

Priority Channel



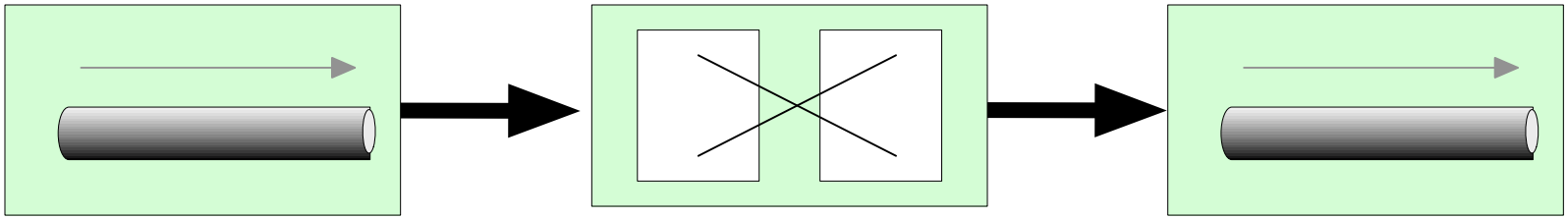
Priority Channel



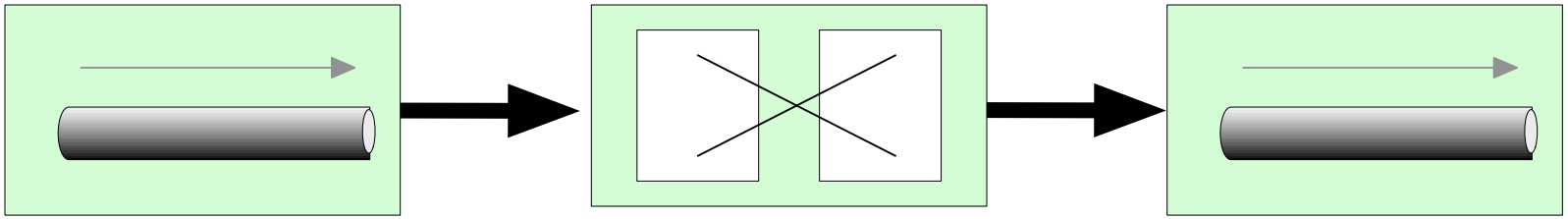

```
<channel id="priorityChannel">
   <priority-queue comparator="someComp" />
</channel>
```


Message
Transformation

Message Translator



Message Translator



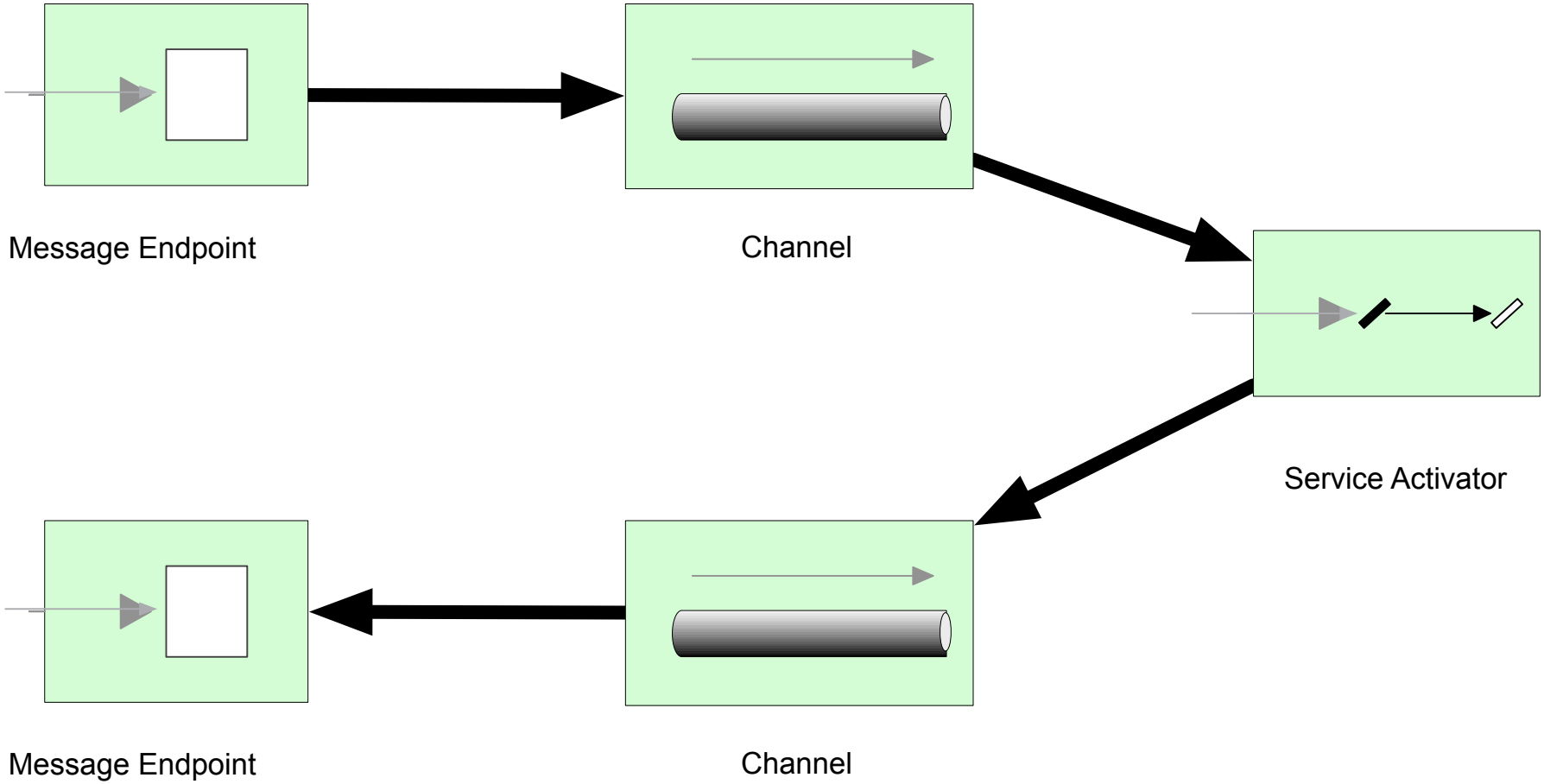

```
<transformer input-channel="input"
output-channel="output" ref="transformer"
method="transform"/>
```

Annotation Based Message Translator

```
@MessageEndpoint
public class MessageTransformer{

   @Transformer(inputChannel="in",
                outputChannel="out")
   public LoanRequest transform(Loan loan){
      return ...;
   }
}
```

Service Activator

Service Activator



Service Activator

```
<channel id="requests"/>
<channel id="quotes"/>

<service-activator input-channel="requests"
                   ref="loanBroker"
                   method="processRequest"
                   output-channel="quotes"/>

<beans:bean id="loanBroker"
class="example.LoanBroker"/>
```

Annotation Based Service Activator

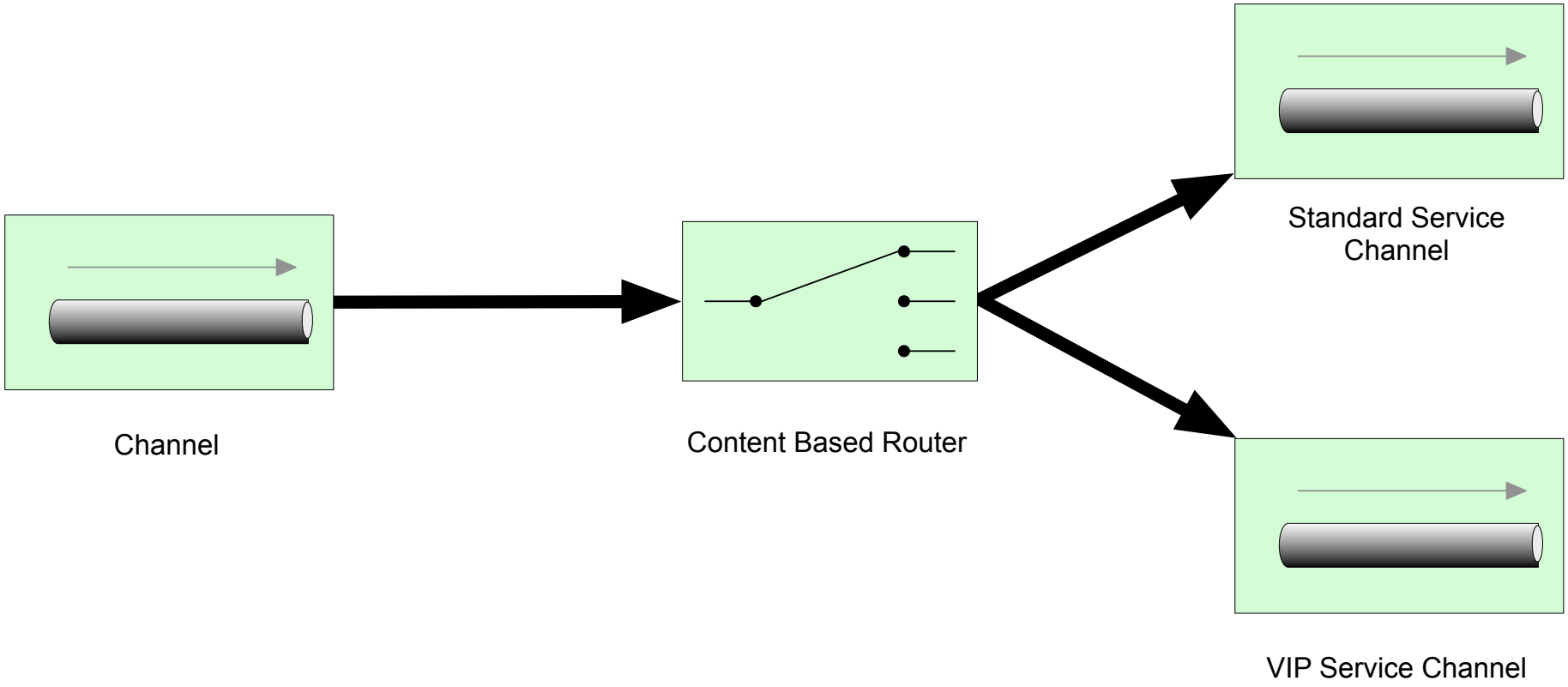
```
@MessageEndpoint
public class LoanBroker {

   @ServiceActivator(inputChannel="x",
outputChannel="y")
   public LoanQuote processRequest(
                  LoanRequest request) {
          LoanQuote quote ...;
          return quote;
   }
}
```

Polling and Transactions

```
<service-activator ref="loanBroker"
                   method="processRequest"
                   input-channel="requests"
                   output-channel="quotes">
   <poller task-executor="pool1">
      <interval-trigger interval="5000"/>
      <transactional propagation="REQUIRES_NEW"/>
   </poller>
</service-activator>
<pool-executor id="pool1" max-size="25"/>
<beans:bean id="transactionManager" ... />
```

Message **Routing**

Content Based Router



MethodInvokingRouter

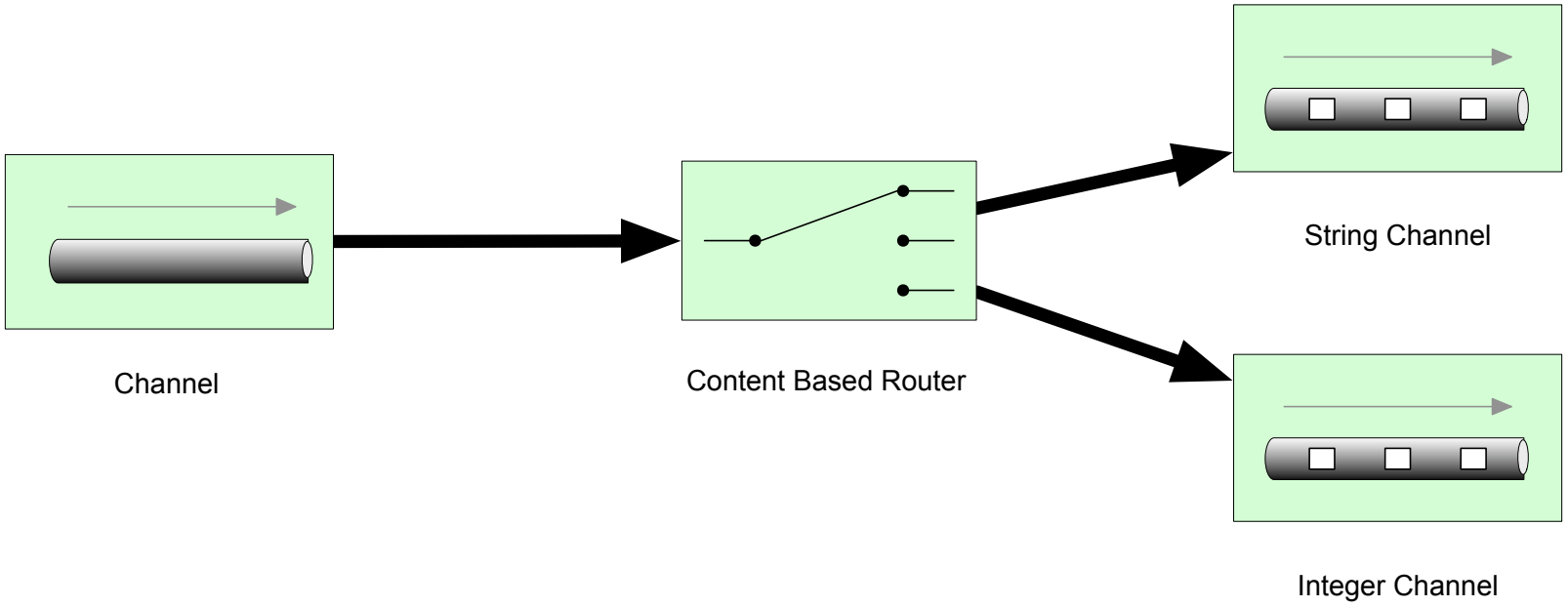
```
<channel id="even"/>

<channel id="odd"/>

<router ref="parityResolver" input-
channel="numbers"/>
```

MethodInvokingRouter

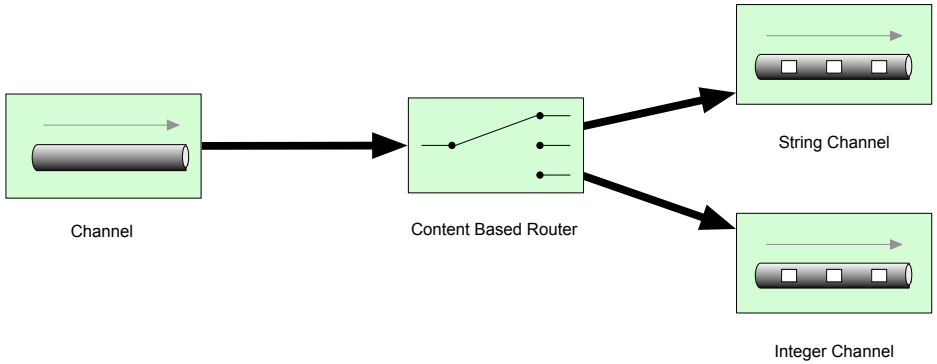
```
<channel id="even"/>

<channel id="odd"/>

<router ref="parityResolver" input-
channel="numbers"/>
```

```
@Router
public String getParity(int i) {
   return (i % 2 == 0) ? "even" : "odd";
}
```

PayloadtypeRouter

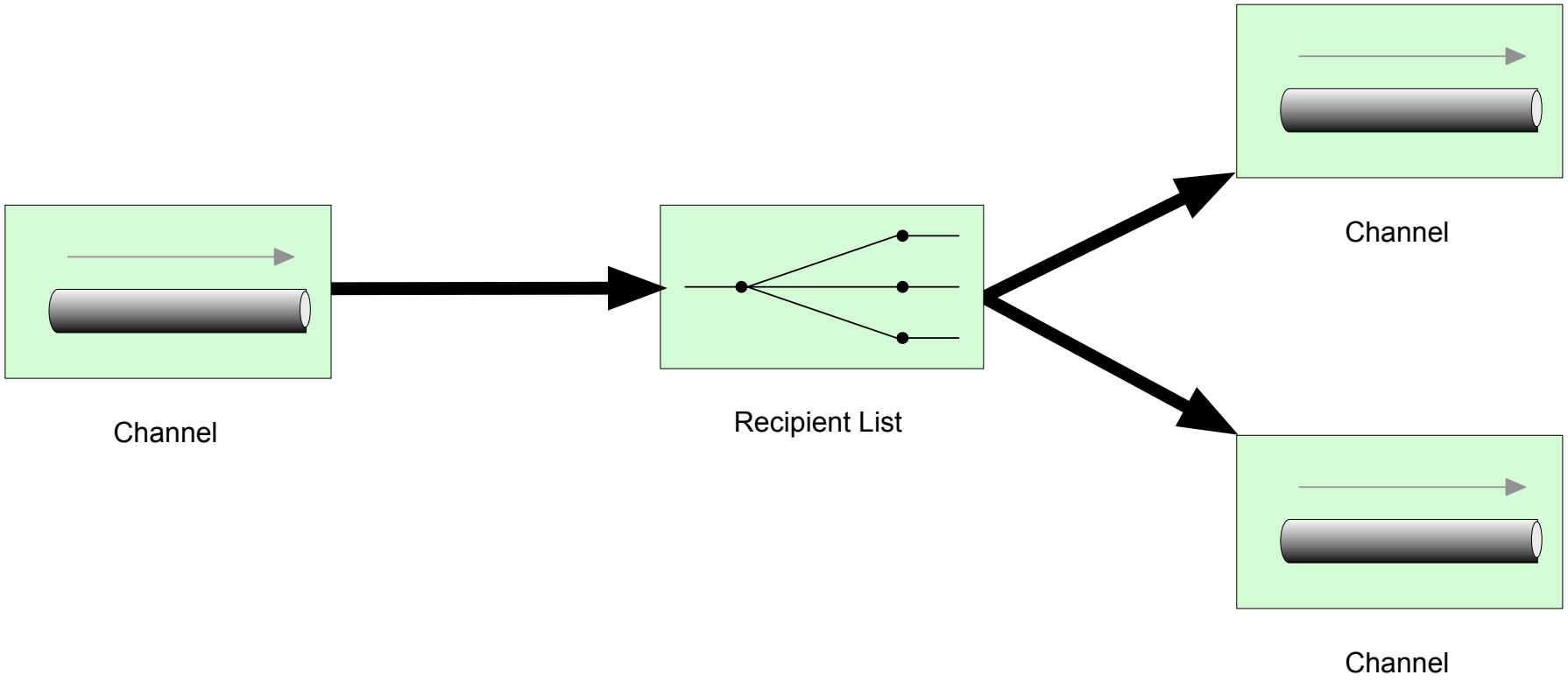


PayloadtypeRouter

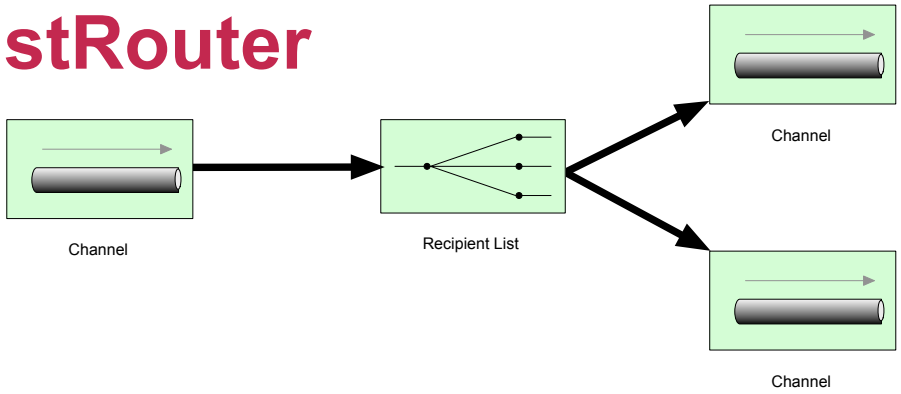



```
typeMap .put(String.class, stringChannel);
typeMap.put(Integer.class, integerChannel);

PayloadTypeRouter router = new PayloadTypeRouter();
router.setPayloadTypeChannelMap(typeMap);

router.handleMessage(new StringMessage("test"));
router.handleMessage(new GenericMessage(123));
```

ReceipientListRouter

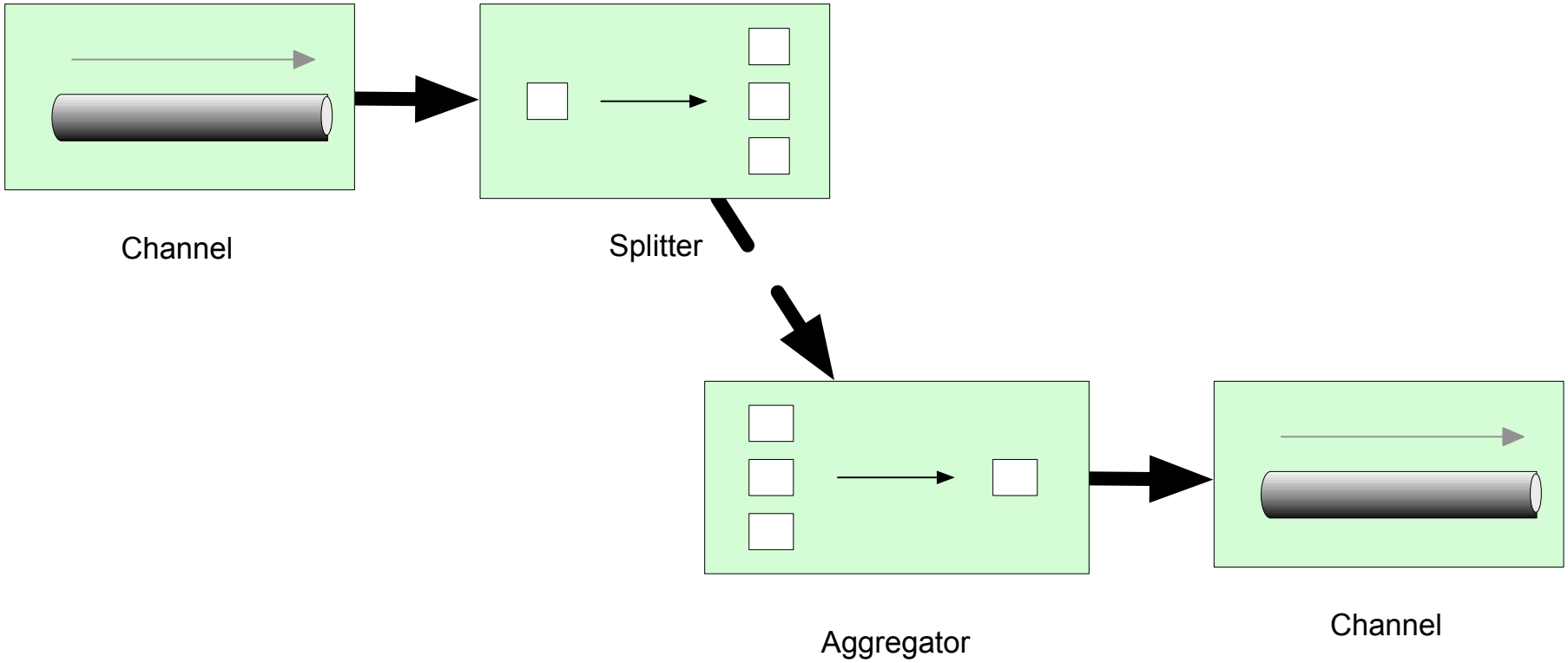


ReceipientListRouter




```
channels.add(channel1);
channels.add(channel2);

RecipientListRouter router = new RecipientListRouter();
router.setChannels(channels);
Message<String> message = new StringMessage("test");

router.handleMessage(message);
```

Splitter And Aggregator



Splitter And Aggregator

```
@Splitter
public List<OrderItem> splitOrder(PurchaseOrder order,
                    @Header("customerId") String customerId) {
        // split the purchase order into order items…
}
```

Splitter And Aggregator

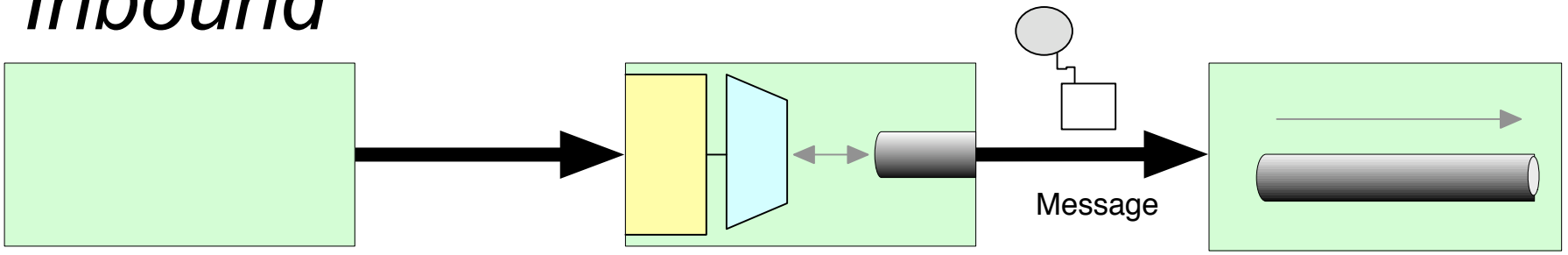
```
@Splitter
public List<OrderItem> splitOrder(PurchaseOrder order,
                    @Header("customerId") String customerId) {
        // split the purchase order into order items…
}
```

```
@Aggregator
public PurchaseOrder aggregateOrder(List<OrderItem> items) {
        // aggregate the items into a single order object...
}
```

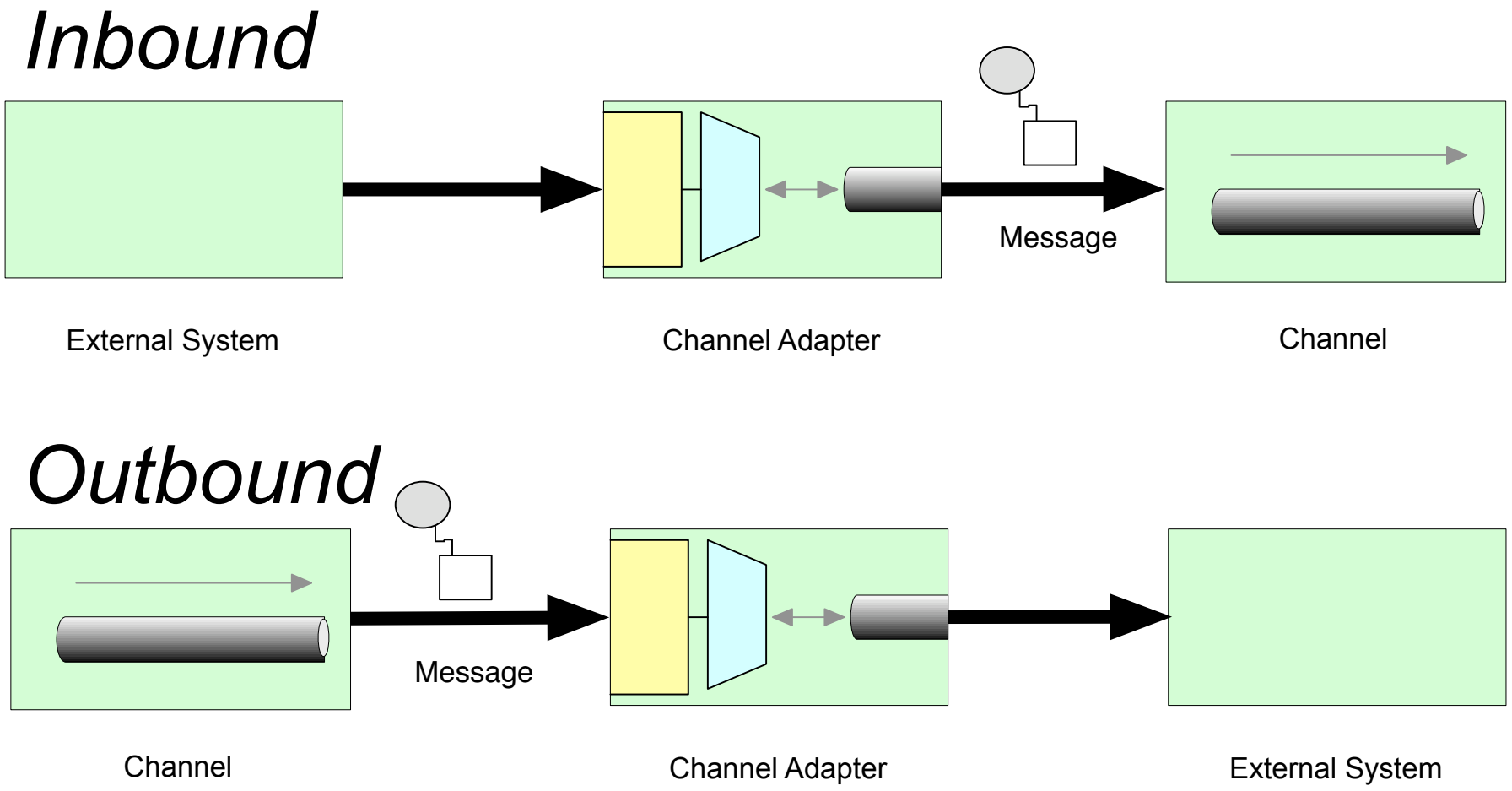
Channel **Adapter**

Channel Adapters

Inbound



Channel Adapters



File Adapter

```
<file:inbound-channel-adapter channel="filesIn"
          directory="${java.io.tmpdir}/test-input">
 <poller max-messages-per-poll="5">
       <cron-trigger expression="*/10 * * * * MON-FRI"/>
 </poller>
</file:inbound-channel-adapter>


<file:outbound-channel-adapter      channel="filesOut"
      directory="${java.io.tmpdir}/test-output"/>
```

JMS Adapter

```
<jms:inbound-channel-adapter channel="input"
      connection-factory="connectionFactory"
      destination-name="sourceQueueName"/>

<jms:outbound-channel-adapter channel="output"
      destination="targetQueue"/>

<jms:inbound-gateway request-channel="inRequests"
      destination="inboundRequestQueue"/>

<jms:outbound-gateway request-channel="outRequests"
   reply-channel="replies" jms-queue="outQueue"/>
```

Method Invoking Adapter

```
<channel id="channel"/>

<inbound-channel-adapter channel="channel"
             ref="reader" method="read">
   <poller max-messages-per-poll="1">
      <interval-trigger interval="1000"/>
   </poller>
</inbound-channel-adapter>

<outbound-channel-adapter channel="channel"
               ref="writer" method="write"/>
```

Webservice Adapter

```
<ws:outbound-gateway uri="http://..."
marshaller="someMarshaller"
unmarshaller="someMarshaller"
request-channel="req" reply-channel="rep"/>
```

```
<ws:inbound-gateway request-channel="req"
reply-channel="rep"
marshaller="someMarshaller"
unmarshaller="someMarshaller" />
```

Other Adapters

- HTTP
- Mail
- RMI
- springsource.org/extensions

Q & A

Agim Emruli
agim.emruli@springsource.com


THANK
YOU!

Picture Credits

	http://www.flickr.com/photos/scraplab/2612334635/
	http://www.flickr.com/photos/mylifestory/527847004/
	http://www.flickr.com/photos/alex-s/116511016/
	http://www.flickr.com/photos/jdan/2324469981/

Picture Credits

	http://www.flickr.com/photos/global-jet/2125557004/