Scatter-Gather

Scatter-Gather

The Scatter-Gather from the EIP patterns allows you to route messages to a number of dynamically specified recipients and re-aggregate the responses back into a single message.

blocked URL

Dynamic Scatter-Gather Example

In this example we want to get the best quote for beer from several different vendors. We use a dynamic Recipient List to get the request for a quote to all vendors and an Aggregator to pick the best quote out of all the responses. The routes for this are defined as:

Error formatting macro: snippet: java.lang.NullPointerException

So in the first route you see that the Recipient List is looking at the `listOfVendors` header for the list of recipients. So, we need to send a message like

Error formatting macro: snippet: java.lang.NullPointerException

This message will be distributed to the following Endpoints: `bean:vendor1`, `bean:vendor2`, and `bean:vendor3`. These are all beans which look like

Error formatting macro: snippet: java.lang.NullPointerException

and are loaded up in Spring like

Error formatting macro: snippet: java.lang.NullPointerException

Each bean is loaded with a different price for beer. When the message is sent to each bean endpoint, it will arrive at the `MyVendor.getQuote` method. This method does a simple check whether this quote request is for beer and then sets the price of beer on the exchange for retrieval at a later step. The message is forwarded on to the next step using POJO Producing (see the @Produce annotation).

At the next step we want to take the beer quotes from all vendors and find out which one was the best (i.e. the lowest!). To do this we use an Aggregator with a custom aggregation strategy. The Aggregator needs to be able to compare only the messages from this particular quote; this is easily done by specifying a correlationExpression equal to the value of the quoteRequestId header. As shown above in the message sending snippet, we set this header to `quoteRequest-1`. This correlation value should be unique or you may include responses that are not part of this quote. To pick the lowest quote out of the set, we use a custom aggregation strategy like

Error formatting macro: snippet: java.lang.NullPointerException

Finally, we expect to get the lowest quote of $1 out of $1, $2, and $3.

Error formatting macro: snippet: java.lang.NullPointerException

You can find the full example source here:

camel-spring/src/test/java/org/apache/camel/spring/processor/scattergather/
camel-spring/src/test/resources/org/apache/camel/spring/processor/scattergather/scatter-gather.xml

Static Scatter-Gather Example

You can lock down which recipients are used in the Scatter-Gather by using a static Recipient List. It looks something like this

```
from("direct:start").multicast().to("seda:vendor1", "seda:vendor2", "seda:vendor3");

from("seda:vendor1").to("bean:vendor1").to("seda:quoteAggregator");
from("seda:vendor2").to("bean:vendor2").to("seda:quoteAggregator");
from("seda:vendor3").to("bean:vendor3").to("seda:quoteAggregator");

from("seda:quoteAggregator")
    .aggregate(header("quoteRequestId"), new LowestQuoteAggregationStrategy()).to("mock:result")
```

A full example of the static Scatter-Gather configuration can be found in the Loan Broker Example.

Using This Pattern

If you would like to use this EIP Pattern then please read the Getting Started, you may also find the Architecture useful particularly the description of Endpoint and URIs. Then you could try out some of the Examples first before trying this pattern out.