Datatype Channel

This section explains, through an example scenario, how the Datatype Channel EIP can be implemented using the WSO2 ESB. The following topics are covered:

- Introduction to Datatype Channel
- Example scenario
  - The ESB configuration
- Simulating the example scenario
  - Setting up the environment
  - Executing the sample
  - Analyzing the output

Introduction to Datatype Channel

This EIP creates a separate channel for each type of data so that all the messages on a given channel will contain the same data type. The sender, who knows the data type, should select the appropriate channel on which to send the message. The receiver identifies the type of data a message contains, based on the channel in which it is received.

For more information, go to Data Type Channel.

![Datatype Channel EIP Diagram](image)

Figure 1: Datatype Channel EIP

Example scenario

This example scenario depicts a Stock Quote service deployed in Axis2 server. It offers several service operations to the user. The ESB uses the filter mediator to identify each action that is specified by the sender and diverts the request into the appropriate sequence. Each sequence acts as a separate channel. The sender experiences the decomposition of channels through a log message indicated by the ESB. There will be a different log message for each operation the sender requests.

The following diagram depicts how this example scenario can be depicted using the ESB.
Before digging into implementation details, let's take a look at the relationship between the example scenario and the Datatype Channel EIP by comparing their core components.

<table>
<thead>
<tr>
<th>Datatype Channel EIP (Figure 1)</th>
<th>Datatype Channel Example Scenario (Figure 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sender</td>
<td>Client</td>
</tr>
<tr>
<td>Datatype Channel</td>
<td>Filter and Sequence mediators of the ESB.</td>
</tr>
<tr>
<td>Receiver</td>
<td>Stock Quote Service</td>
</tr>
</tbody>
</table>

**The ESB configuration**

Given below is the ESB configuration for simulating the example scenario explained above.
<definitions xmlns="http://ws.apache.org/ns/synapse">
  <endpoint name="StockQuoteReceiver">
    <address uri="http://localhost:9000/services/SimpleStockQuoteService"/>
  </endpoint>
  <sequence name="MarketActivity">
    <log level="custom">
      <property name="Messaging_Channel" value="MARKET_ACTIVITY"/>
    </log>
    <send>
      <endpoint key="StockQuoteReceiver"/>
    </send>
  </sequence>
  <sequence name="FullQuote">
    <log level="custom">
      <property name="Messaging_Channel" value="FULL_QUOTE"/>
    </log>
    <send>
      <endpoint key="StockQuoteReceiver"/>
    </send>
  </sequence>
  <sequence name="StockQuote">
    <log level="custom">
      <property name="Messaging_Channel" value="STOCK_QUOTE"/>
    </log>
    <send>
      <endpoint key="StockQuoteReceiver"/>
    </send>
  </sequence>
  <proxy name="datatype-channel-proxy" startOnLoad="true" transports="http https">
    <target>
      <log/>
      <inSequence>
        <switch source="get-property('Action')">
          <case regex="/urn:getQuote/*">
            <sequence key="StockQuote"/>
          </case>
          <case regex="/urn:getFullQuote/*">
            <sequence key="FullQuote"/>
          </case>
          <case regex="/urn:getMarketActivity/*">
            <sequence key="MarketActivity"/>
          </case>
        </switch>
      </inSequence>
      <outSequence>
        <respond/>
      </outSequence>
    </target>
  </proxy>
</definitions>
Simulating the example scenario

Now, let's try out the example scenario explained above.

Setting up the environment

You need to set up the ESB, and the back-end service:

1. Download the `Datatype_1.0.0.zip` file, which includes the ESB configuration described above.
2. See Setting up the Environment for instructions on setting up the ESB and the back-end service.

When you set up the environment, note that you only need to start **one instance** of the back-end service (Stock Quote Service) to simulate this example.

Executing the sample

Let's send a request to the ESB using the **Stock Quote Client** application. Find out more about the Stock Quote Client from the ESB documentation.

1. Open a new terminal, and navigate to the `<ESB_HOME>/samples/axis2Client/` directory. The Stock Quote client application is stored in this directory.
2. Execute the following command to send the request to the ESB.

```
ant stockquote -Dtrpurl=http://localhost:8280/services/datatype-channel-proxy -Dmode=quote
```

Analyzing the output

When you execute the command above, the ESB first receives the message and then routes it to the back-end service (StockQuoteService). The following output will be printed on the Axis2 server's console:

**Stock Client Console output :**

Standard :: Stock price = $172.81050109499768

**Axis2 server console output:**

samples.services.SimpleStockQuoteService :: Generating quote for : IBM

**ESB Profile logs:**

INFO - LogMediator Messaging_Channel = STOCK_QUOTE

Also execute the below commands, and observe the ESB profile log of the corresponding values: STOCK_QUOTE, MARKET_ACTIVITY and FULL_QUOTE.

- `ant stockquote -Dtrpurl=http://localhost:8280/services/datatype-channel-proxy -Dmode=marketactivity`
- `ant stockquote -Dtrpurl=http://localhost:8280/services/datatype-channel-proxy -Dmode=fullquote`

Notice the following respective output.

**Stock Client Console output :**

- Activity :: Average price = $123.67485145988432
- Full :: Average price = $125.67875616729333
Axis2 server console output:

- samples.services.SimpleStockQuoteService :: Generating Market activity report for: [JLN, FDZ, EQR, XNV, RDR, CZC, LIY, ZEP, ZJX, GWO, STS, NQU, RMA, UUR, PFL, ZEF, IYU, ZLV, KTW, PUN, IOZ, PZJ, HAE, PSL, CQM, CLX, BWI, UYF, QWC, EKB, LMM, UQI, GZA, KRC, GFB, DWM, ETA, SRS, VEP, ZTS, TNE, FJF, LNV, QBY, ZIO, HBS, IIW, SNO, MNO, BTY, OGJ, OUW, CLW, OZT, MXB, HNK, FQC, VEI, BLD, LUP, PHR, JUQ, MZM, GIL, EVE, UAH, SHV, WYS, MAG, XBX, ZYB, MUX, MUO, DAM, DVR, RDF, LGB, KGS, DWP, ZAG, SDF, BSF, CTX, MKG, YTO, RRX, OVJ, MEZ, ODU, JGU, GQB, SLW, UCQ, GDI, DIO, BKV, UUQ, JES, TAZ, AAU]
- samples.services.SimpleStockQuoteService :: Full quote for: IBM

ESB Profile logs:

- INFO - LogMediator Messaging_Channel = MARKET_ACTIVITY
- INFO - LogMediator Messaging_Channel = FULL_QUOTE