



# **Test Guide**

## **CEF e-Delivery**

Author(s) : Maarten DANIELS  
Reviewed by :

Approved by :  
1.02 version :

Date: 17/12/2015

## Document history

Version	Date	Comment	Modified pages
1.00	05/11/2015	Creation for demo	All pages
1.01	18/11/2015	Updated localUrl to remoteUrl	Page 7
1.02	17/12/2015	Updated for 3.2 release	All pages

<b>1. Overview .....</b>	<b>3</b>
<b>2. Prerequisites .....</b>	<b>3</b>
2.1. Complete the Quick Start Guide .....	3
2.2. Install SoapUI .....	3
2.3. Configure SoapUI .....	3
<b>3. Test Scenarios .....</b>	<b>6</b>
3.1. Submit Request, push notification and submit Response .....	6
<b>4. Data Structure .....</b>	<b>15</b>

## 1. OVERVIEW

This document contains information to set up the test and demo framework for the AS4 component of CEF e-Delivery.

First the test environment will be set up and the test tool will be configured.

Then the test scenario is explained and guidance is provided on how to run it.

## 2. PREREQUISITES

### 2.1. Complete the Quick Start Guide

Ensure that you have a working environment by following the steps defined in the Quick Start Guide.

### 2.2. Install SoapUI

Install SoapUI, an open source test tool for web service testing. A free version without restrictions is available and this free version is sufficient to run the tests in this package.

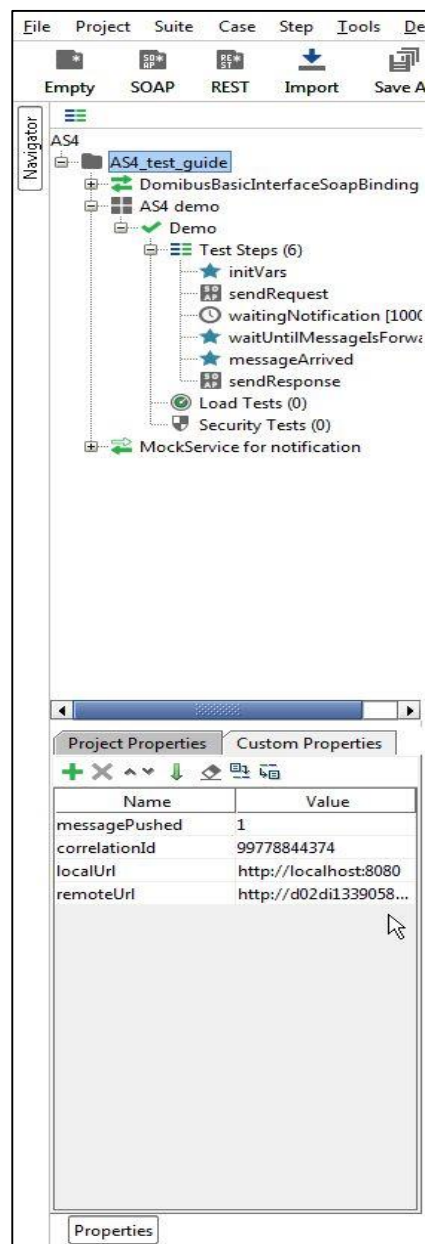
Download SoapUI free via <http://www.soapui.org/> and follow the installation instructions on the website. The version used to create the tests can be downloaded via <http://sourceforge.net/projects/soapui/files/soapui/5.2.1/>.

### 2.3. Configure SoapUI

Inside SoapUI, create a new workspace (File->New Workspace) and load the project in this package (AS4-test-guide-soapui-project.xml) using the import functionality of SoapUI (File->Import Project).

In the Navigator in SoapUI, left-click on the project "AS4\_test\_guide" and select "Custom Properties" in the Properties panel.

- Set the Property named "localUrl" to the IP address and port of the machine that is running instanceA. If this is your local machine, then the value can remain "http://localhost:8080".
- Set the Property named "remoteUrl" to the IP address and port of the machine that is running instanceB.

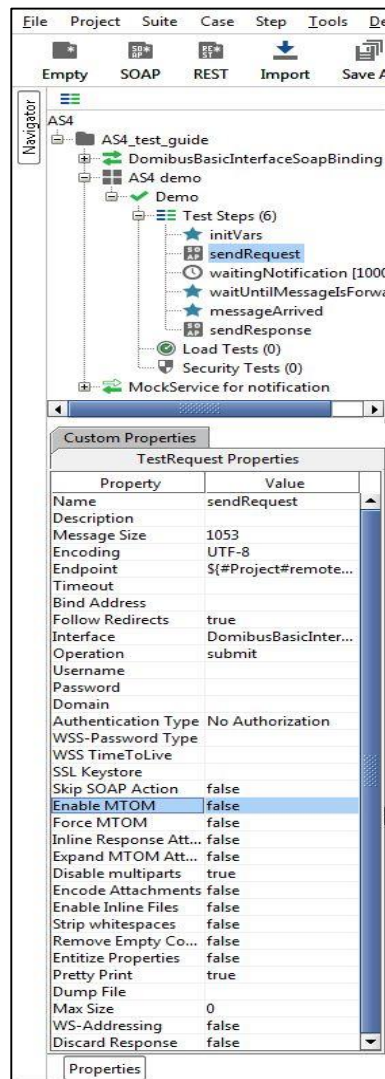


**Figure 1 - Custom Properties**

Now the service endpoints in the test steps will automatically be created from these values and no further configuration is required.

The WSDL of the services were loaded when creating the test project and they are stored in the project file. If you want to reload them, ensure that your local AS4 gateway is up and running. Otherwise the URL of the WSDL (e.g <http://localhost:8080/domibus-app/services/basic?wsdl>) will not be accessible.

MTOM (Message Transmission Optimization Mechanism) is a method to efficiently send binary data to and from web services. This option is already pre-configured in the test request properties that need it. Changing this value might result in failures when transmitting the binary content of the exchanged messages.



**Figure 2 – Test Request properties**

No authentication or authorization is required in the current version of the setup. The default setup of the SoapUI project will be sufficient to run the test case.

## 3. TEST SCENARIOS

### 3.1. Submit Request, push notification and submit Response

In this scenario, the party instanceBid1 will first send a request to the party instanceAid1. Then, after being notified of this message via the push mechanism, the party instanceAid1 will send a response to party instanceBid1.

The detailed steps are listed below:

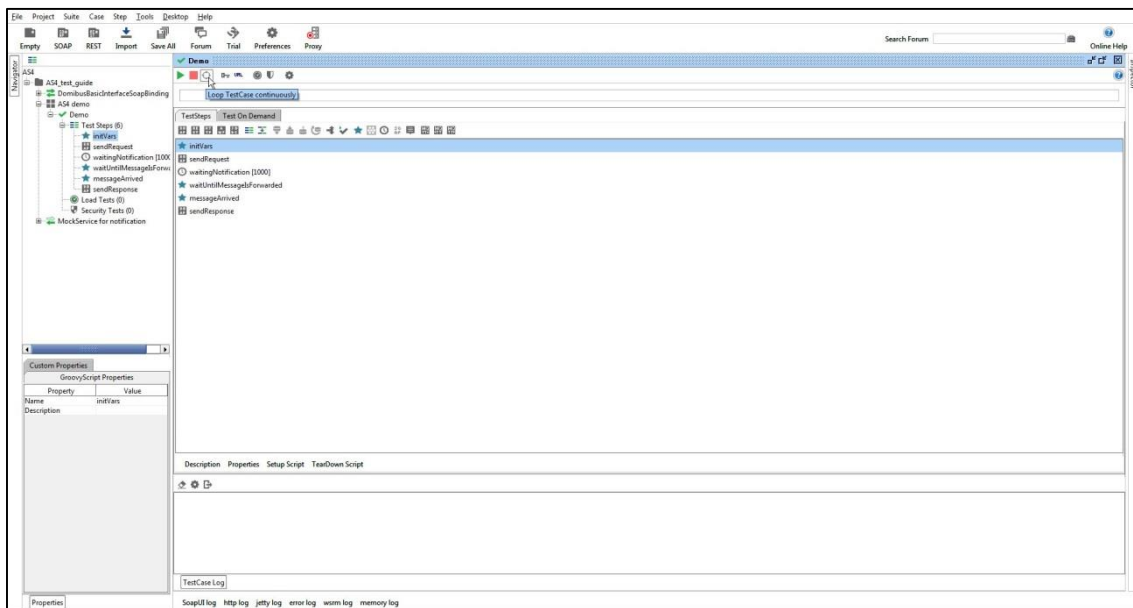
1. In step "initVars", a property "messagePushed" is initialized to "0". This property is used to indicate that a message has been pushed by the mock service that is listening for the push notification.
2. In step "sendRequest", a message is submitted to the AS4 gateway that is located at remoteUrl. The message contains a Payload and an Attachment. The Payload is mandatory and contains the XML data of the request. Its content type must be set to "text/xml". The Attachment is optional and can contain any type of binary attachment. Regardless of the specific type of attachment used, its content type must be set to "application/octet-stream". In this specific test step, the Attachment contains a PDF. To modify the Payload or Attachment, refer to 3.1.2 Updating the test data.
3. In the background, the AS4 gateway running at remoteUrl uses the message from step 2 as an input to construct an AS4 message and send it to the AS4 gateway that is running at localUrl.
4. In the background, the AS4 gateway that is running at localUrl will receive the message that is sent in step 3. Upon reception, the message will first be stored in the database of the AS4 gateway. Afterwards, the AS4 gateway will try to notify a preconfigured web service endpoint by pushing the message to this endpoint. If the push notification succeeds, the AS4 gateway will delete the message from the database. If the push notification fails, the AS4 gateway will retry pushing the message according to a retrial policy. If the retrial policy times out and the message cannot be forwarded to the web service endpoint, an error is logged in the logfile and the message is sent to the application server's (e.g. JBoss) error queue.
5. In the background, the mock service "MockService for notification" is running and listening for pushed messages that arrive on localUrl. When a new message arrives, the mock service will log the actual message in the script log window of SoapUI, extract the Conversation Id from the message in step 2 and store it in a project property called "conversationId". After the

message has arrived, the mock service will set the project property "messagePushed" to "1" so that the demo scenario knows that the message has been pushed. To modify the mock service, refer to 3.1.2 Updating the test data.

6. The demo scenario will wait for the message to be arrived to localUrl and forwarded to the preconfigured web service endpoint at localUrl. The steps "waitingNotification" and "waitUntilMessageIsForwarded" will loop until the project property "messagePushed" is set to "1" by the mock service "MockService for notification". Then the demo scenario will continue with the step "messageArrived" and "sendResponse".
7. In step "sendResponse", a message is submitted to the AS4 gateway that is located at localUrl. The message contains a Payload and a Attachment. The Payload is mandatory and contains the XML data of the response. Its content type must be set to "text/xml". The Attachment is optional and contains any type of binary attachment. Regardless of the specific type of attachment used, its content type must be set to "application/octet-stream". In this specific test step, the Attachment contains a PDF. Since this message is a response to the message sent in step 2, the "ConversationId" which is extracted in step 5 will be set in the "ConversationId" element of the message in this step 7.
8. In the background, the AS4 gateway running at localUrl will use the message from step 7 as input to construct an AS4 message and send it to the AS4 gateway that is running at remoteUrl. Note that the mock service "MockService for notification" that is running and listening for pushed messages that arrive on localUrl will not pick up the message from step 7 since this message is pushed on the remoteUrl.

To run this test scenario, open the "demo" test case and click the green "play" button. Each test step will be executed and a green progress bar will indicate the successful completion of the test steps.

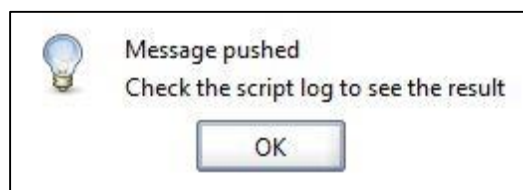
## Test Guide – CEF e-Delivery



**Figure 3 - Run test scenario**

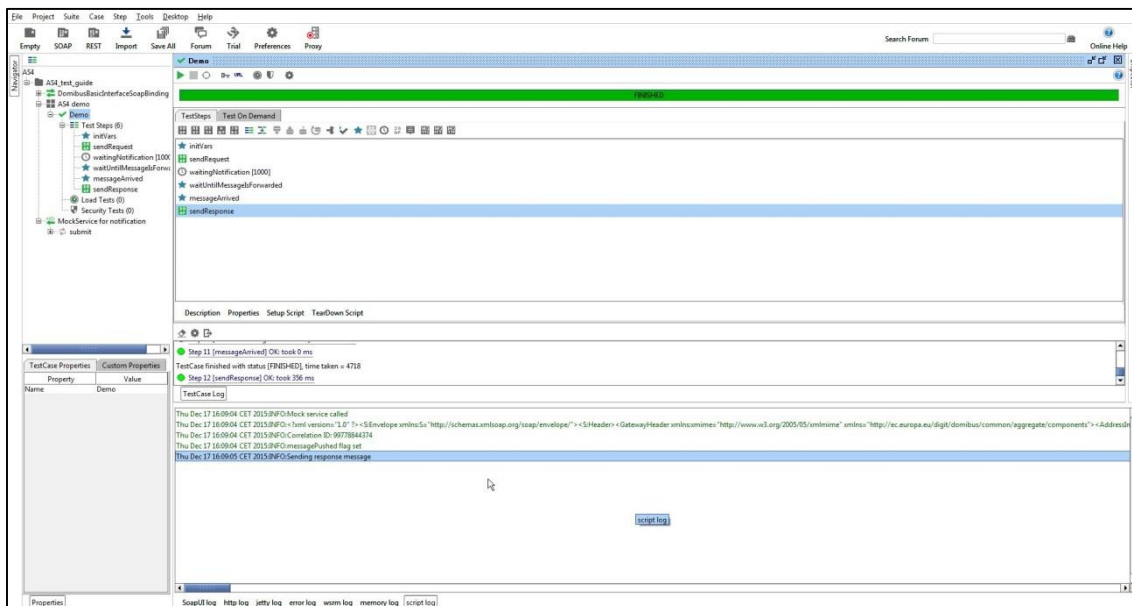
Beware that before executing the test case, the mock service "MockService for notification" needs to be running in the background. This can be done by right-clicking the mock service and selecting "Start Minimized".

Additionally, if the mock service receives a document, a pop-up is shown that the message is found and its contents are printed to the script log in SoapUI.



**Figure 4 - Pop-up showing that the web service forwarded message is found**





**Figure 5 – The contents of the web-service forwarded message are printed on the script log in SoapUI.**

**Note:** if the web service forwarding was enabled without the mock service running, several messages can arrive at once when starting up the mock service. This is due to a retrieval policy on the forwarding inside the AS4 gateway.

After running this test case, the mock service can be stopped in SoapUI by right-clicking "MockService for notification" and selecting "Stop".

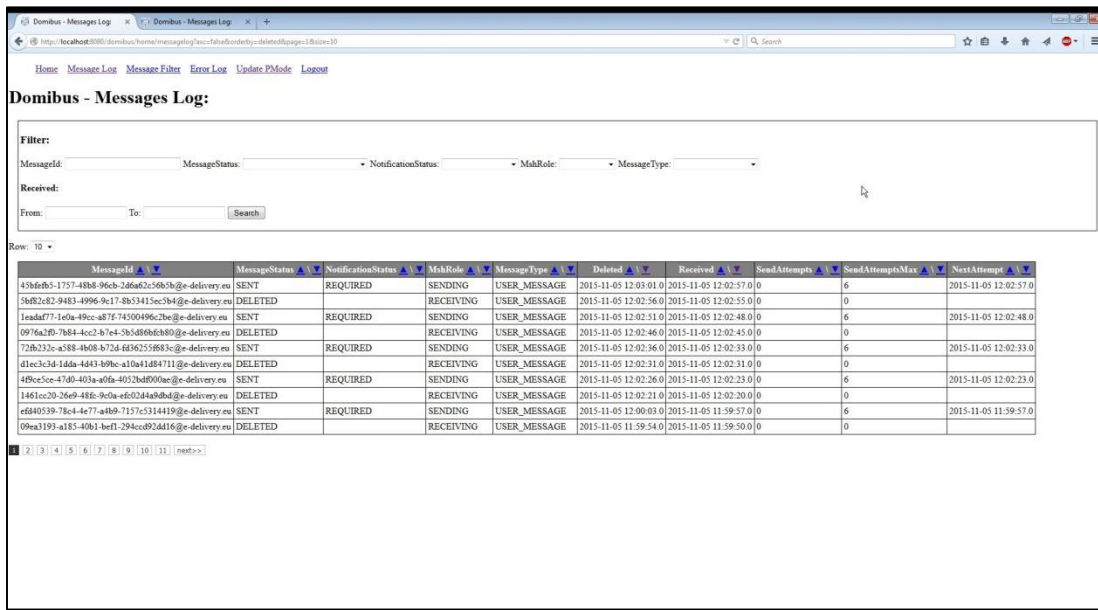
### 3.1.1. Verifying the message status in the administration console

The status of messages that are sent or received can be verified in the administration console. Both administration consoles of the local and remote instances can be used for this purpose. Upon calling the "sendMessage" operation in the "sendRequest" or "sendResponse" test step, a "messageID" is synchronously returned. This "messageID" can be used to verify the status of the message in the local or remote instance.

As an example, the figure below shows that the message with ID "5bf82c82-9483-4996-9c17-8b53415ec5b4@e-delivery.eu" has the status "DELETED". This indicates that the message has been received successfully from the other instance, pushed to the preconfigured web service endpoint and deleted from the database at the local instance.

## Test Guide – CEF e-Delivery

As another example, the same figure below shows that the message with ID "45bfebf5-1757-48b8-96cb-2d6a62c56b5b@e-delivery.eu" has the status "SENT". This indicates that the message has been successfully sent to the other instance.

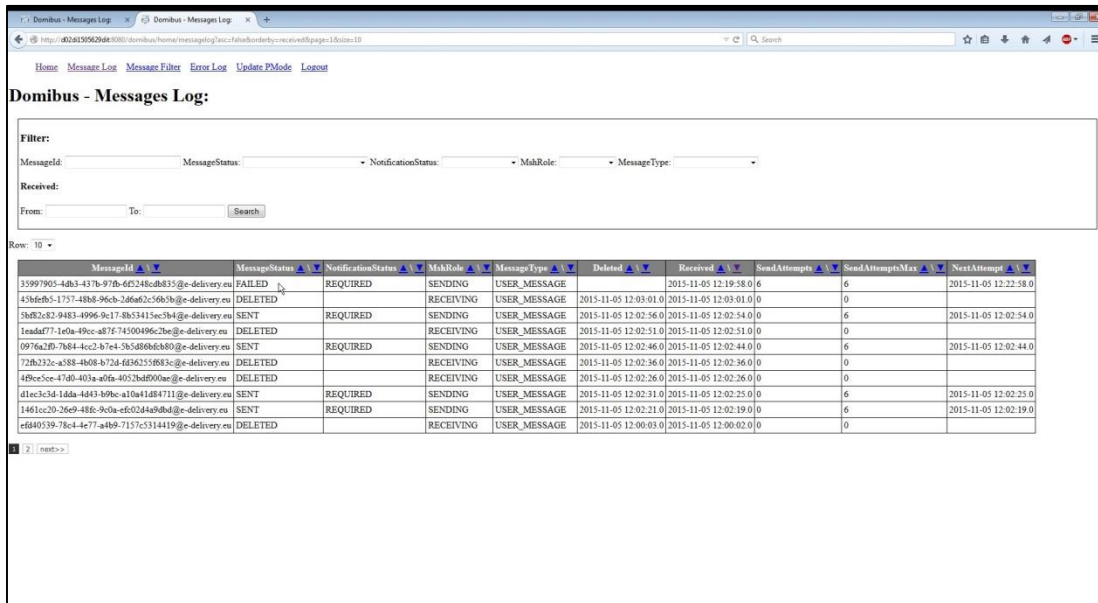


MessageId	MessageStatus	NotificationStatus	MailRole	Message Type	Deleted	Received	SendAttempts	SendAttemptsMax	NextAttempt
45bfebf5-1757-48b8-96cb-2d6a62c56b5b@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-11-05 12:03:01	2015-11-05 12:02:57	0	6	2015-11-05 12:02:57
5b82c82-9483-4996-9c17-8b53415ec5b4@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE	2015-11-05 12:02:56	2015-11-05 12:02:55	0	0	
1eada777-7e0a-49cc-a87e-74500496c2be@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-11-05 12:02:51	2015-11-05 12:02:48	0	6	2015-11-05 12:02:48
0976a2f0-7b64-4cc2-b7e4-5b5486b6c880@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE	2015-11-05 12:02:46	2015-11-05 12:02:45	0	0	
72b232c-a588-4b08-b7d4-6436255883c@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-11-05 12:02:36	2015-11-05 12:02:33	0	6	2015-11-05 12:02:33
d1ec3c3d-1dda-4443-b9bc-a10a1d84711@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE	2015-11-05 12:02:31	2015-11-05 12:02:31	0	0	
4f9cc5ce-47d0-403a-a0b0-4052bd000ae@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-11-05 12:02:26	2015-11-05 12:02:23	0	6	2015-11-05 12:02:23
1461cc20-26e9-486c-9c0a-ef6024a9dbd@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE	2015-11-05 12:02:21	2015-11-05 12:02:20	0	0	
ef640539-78c4-4e77-a8b9-7157c5314419@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-11-05 12:00:03	2015-11-05 11:59:57	0	6	2015-11-05 11:59:57
09ea3193-a185-40b1-bef1-294cc092d416@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE	2015-11-05 11:59:54	2015-11-05 11:59:50	0	0	

Figure 6 – Verifying the message status in the administration console.

If a message cannot be sent because for example the other instance is not available, then the administration console will indicate that the message is in state "FAILED" once the number of "SendAttempts" has reached the value in "SendAttemptsMax". Note that with the current component configuration, the message sender is not notified in any other way of the message failure (since the "sendMessage" operation synchronously returned an HTTP 200 OK with the "messageID").

## Test Guide – CEF e-Delivery



Domibus - Messages Log:

Filter:

MessageId: MessageStatus: NotificationStatus: MailRole: MessageType:

Received:

From: To: Search

Row: 10

MessageId	MessageStatus	NotificationStatus	MailRole	Message Type	Deleted	Received	SendAttempts	SendAttemptsMax	NextAttempt
15997905-4db3-437b-97b6-4f246c0b815@e-delivery.eu	FAILED	REQUIRED	SENDING	USER_MESSAGE		2015-11-05 12:19:58.0	6	6	2015-11-05 12:22:58.0
43b6db5-1757-48b8-96cb-26662c56b5b4@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE		2015-11-05 12:03:01.0	0	0	
5b82c82-9483-4996-9c17-8b53415ec3b4@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE		2015-11-05 12:02:56.0	0	6	2015-11-05 12:02:54.0
1eadaf77-1e0a-49cc-a87c-74500496c2be@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE		2015-11-05 12:02:51.0	0	0	
0976a2b6-7b64-4cc2-b7c4-5b548d66c880@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE		2015-11-05 12:02:46.0	0	6	2015-11-05 12:02:44.0
72b232c-a588-4b08-b73d-6f6255f883c@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE		2015-11-05 12:02:36.0	0	0	
4f8cc5cc-47d0-403a-a6b6-4052bd600a0e@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE		2015-11-05 12:02:36.0	0	0	
d1ec3c3d-1dda-4443-b9bc-a10a41d84711@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE		2015-11-05 12:02:31.0	0	6	2015-11-05 12:02:25.0
1461cc20-26e9-486c-9c0a-e8c024a8dbd@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE		2015-11-05 12:02:21.0	0	6	2015-11-05 12:02:19.0
e640539-78c4-4e77-a4b9-7157c5314419@e-delivery.eu	DELETED		RECEIVING	USER_MESSAGE		2015-11-05 12:00:03.0	0	0	

Figure 7 – A message in status FAILED in the administration console.

### 3.1.2. Updating the test data

To change the Payload or Attachment in the Request or Response message in the test scenario, ensure that the "Attachments" settings in SoapUI are correctly defined (see figure below):

- The Payload contains the XML data of the document. Its "Part" must be matching the info from the XML body. Its "Content type" must be set to "text/xml".
- The Attachment can contain any type of binary attachment. Its "Part" must be matching the info from the XML body. Its "Content type" must be set to "application/octet-stream".

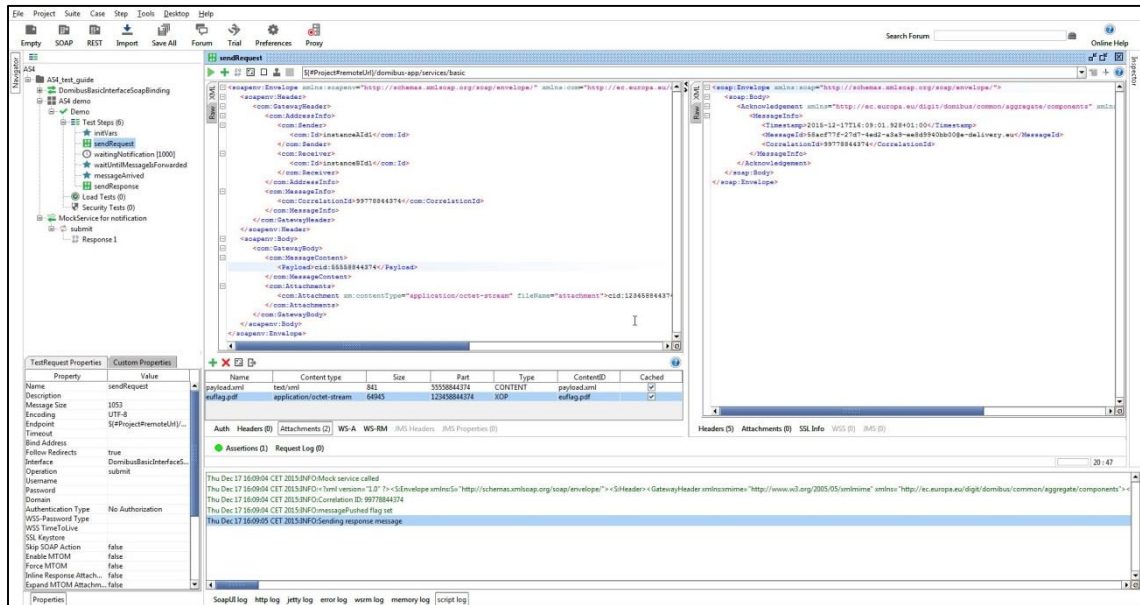


Figure 8 – Verifying the "Attachments" settings in SoapUI.

### 3.1.3. Updating the mock service

To update the mock service options (e.g. path or port), click the settings icon in the mock editor while the mock service is stopped.

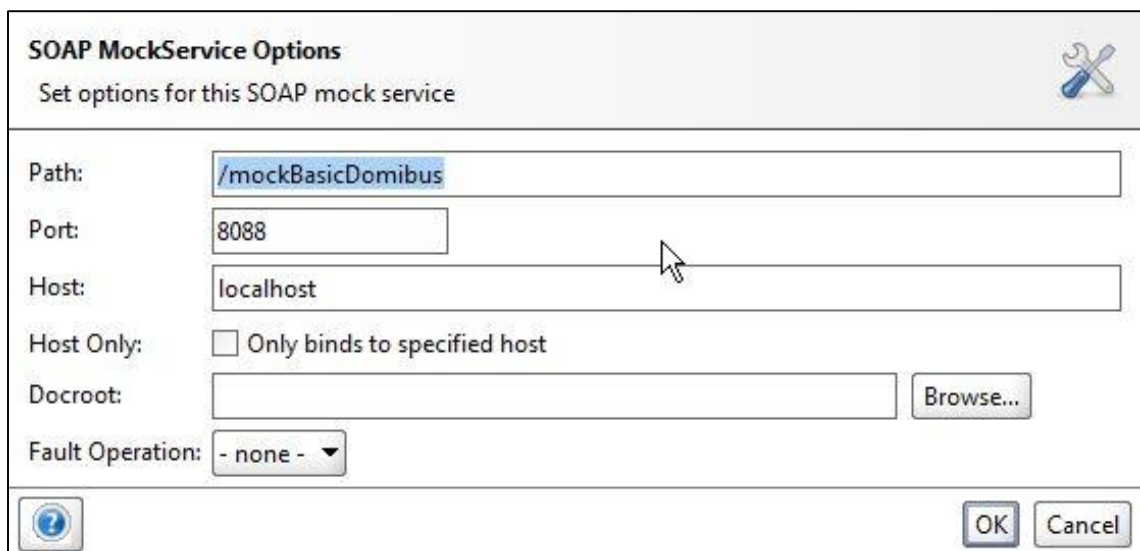


Figure 9 – Updating the mock service options.

To update the response message that the mock service returns when the web service endpoint is called, edit the Response1 data in the sendMessage operation.

## Test Guide – CEF e-Delivery

By default, the dummy message that is returned is sufficient for the correct functioning of the push notification.

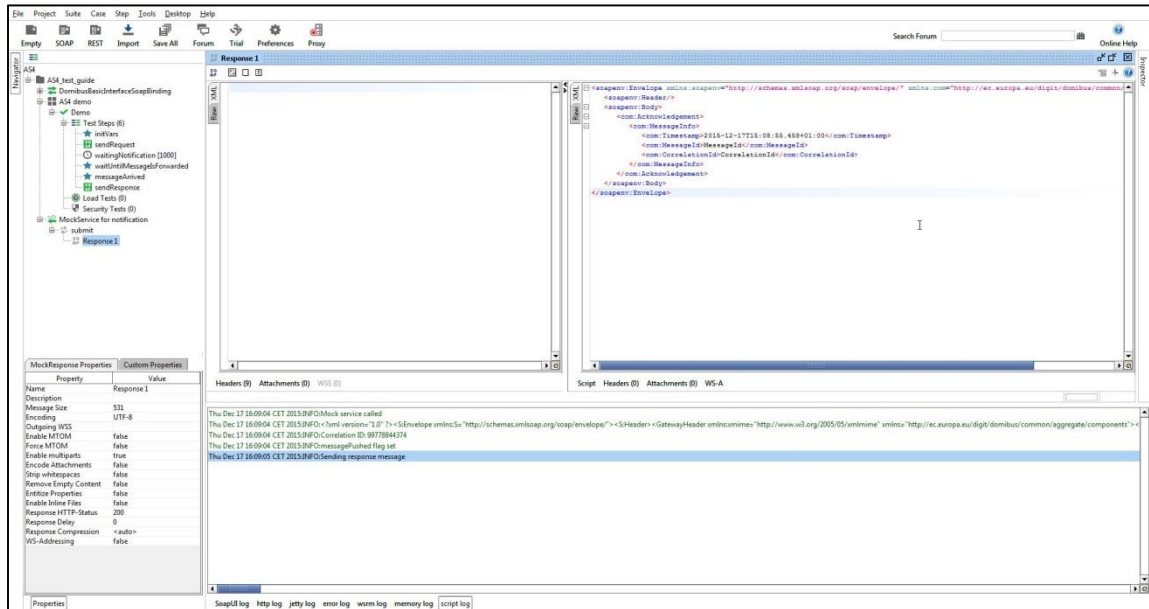


Figure 10 – Updating the mock response.

To update the script that is called each time the mock service receives a new message, edit the "OnRequest Script" in the mock editor.

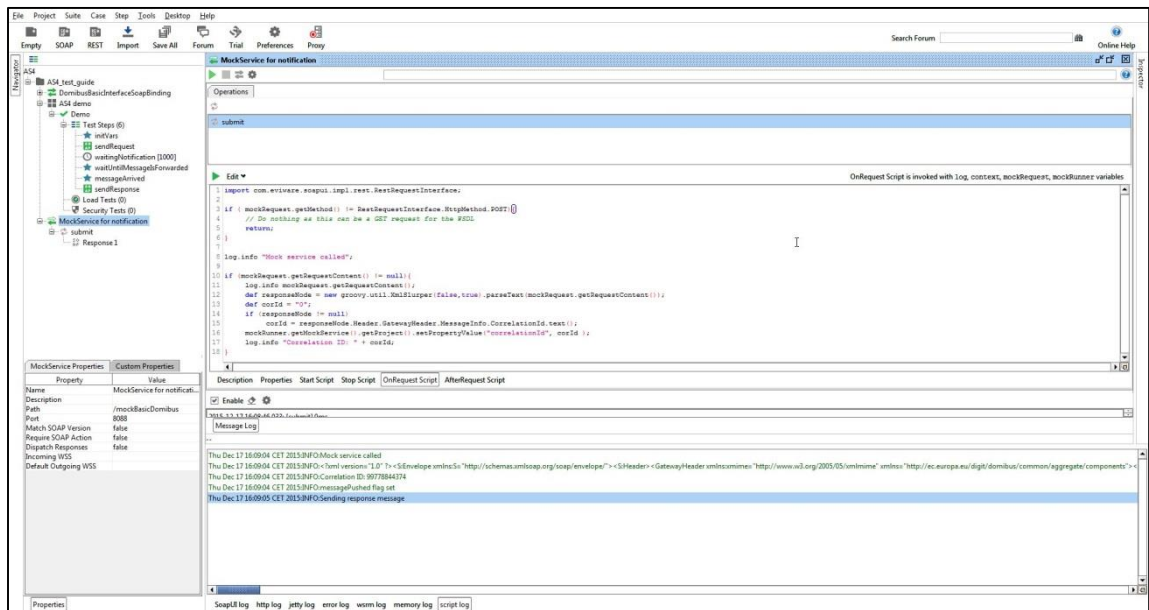
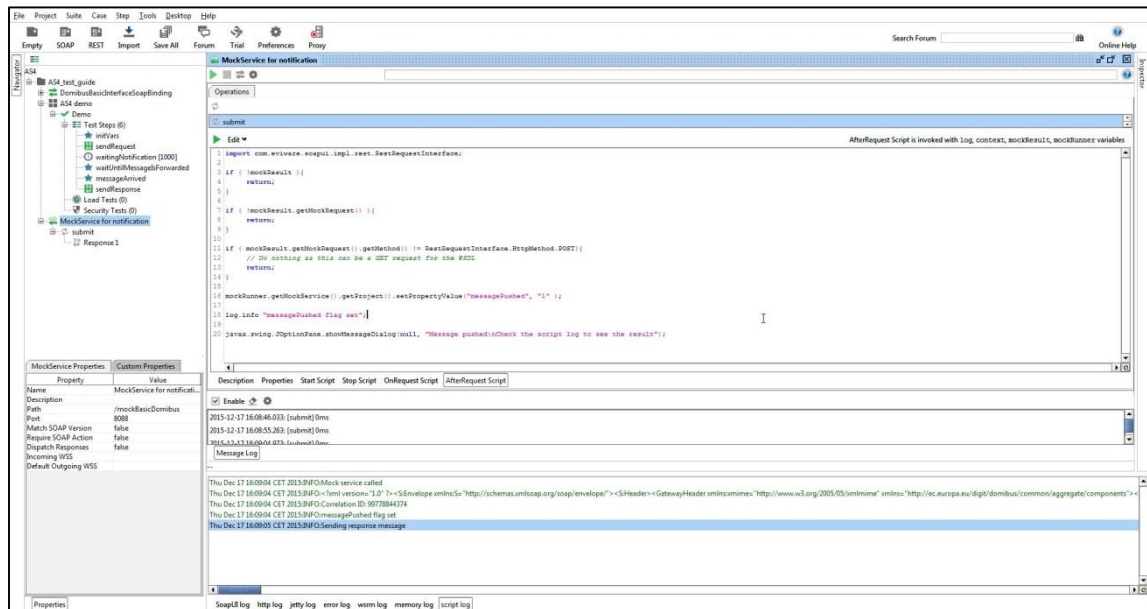


Figure 11 – Updating the OnRequest Script.

To update the script that is called each time the mock service has finished receiving a new message, edit the "AfterRequest Script" in the mock editor.



**Figure 12 – Updating the AfterRequest Script.**

## 4. DATA STRUCTURE

SOAP REQUEST	XSD Element	Comment
<pre> &lt;soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:com="http://ec.europa.eu/digit/domibus/common/aggregate/components" xmlns:xm="http://www.w3.org/2005/05/xmlmime"&gt;   &lt;soapenv:Header&gt;     &lt;com:GatewayHeader&gt;       &lt;com:AddressInfo&gt;         &lt;com:Sender&gt;           &lt;com:Id&gt;Sender Identifier&lt;/com:Id&gt;         &lt;/com:Sender&gt;         &lt;com:Receiver&gt;           &lt;com:Id&gt;Receiver Identifier&lt;/com:Id&gt;         &lt;/com:Receiver&gt;       &lt;/com:AddressInfo&gt;       &lt;com:MessageInfo&gt;         &lt;com:CorrelationId&gt;CorrelationId&lt;/com:CorrelationId&gt;       &lt;/com:MessageInfo&gt;     &lt;/com:GatewayHeader&gt;   &lt;/soapenv:Header&gt;   &lt;soapenv:Body&gt;     &lt;com:GatewayBody&gt;       &lt;com:MessageContent&gt;         &lt;Payload&gt;Payload&lt;/Payload&gt;       &lt;/com:MessageContent&gt;       &lt;com:Attachments&gt;         &lt;com:Attachment xm:contentType="application/octet-stream" fileName="attachment"&gt;Attachment&lt;/com:Attachment&gt;       &lt;/com:Attachments&gt;     &lt;/com:GatewayBody&gt;   &lt;/soapenv:Body&gt; &lt;/soapenv:Envelope&gt; </pre>	GatewayHeader	Gateway Header provides semantic information needed for the routing of documents exchanged over the AS4 Access Points.
	AddressInfo	It contains the identifier of Sender and Reciever Partner.
	GatewayHeader AddressInfo Sender Id	Sender Identifier and Receiver Identifier represent the organizations that send and receive the business documents. They are both used in the authorization process (PMode) and in the public key infrastructure (PKI). They must be a unique identification key in the eDelivery network.
	GatewayHeader AddressInfo Receiver Id	
	MessageInfo CorrelationId	CorrelationId is a unique identifier in the context of documents exchanged. Conversations are a way to group and manage a related set of messages and this can be achieved using this correlationId field.
	GatewayBody	GatewayBody contains the application-defined data (document and attachment) being exchanged in a business context.
	GatewayBody MessageContent Payload	Payload is the actual Business message. It is a mandatory field in the message.
	GatewayBody Attachment	Attachment is any type of binary file that can be sent along with the Business message. It is an optional field. Only one attachment is accepted and transferred.