GlassFish ESB v2.2 Field Notes
Processing Explicit HL7 v2 Accept Acknowledgements

Michael.Czapski@sun.com
January 2010, Release 1.0.0.0

Table of Contents

Introduction

The HL7 v2 standard mandates the use of acknowledgments to ensure message delivery, critical in Healthcare. There are the “Original Mode” acknowledgements and “Enhanced Mode” acknowledgements. Within the enhanced mode acknowledgements there are “Accept Acknowledgements” and “Application Acknowledgements”.

This Note walks through development of two BPEL Module-based solutions that cooperate in generating and processing Enhanced Accept Acknowledgements using HL7 v2.3.1 messages. This discussion should apply to any v2.x, greater then v2.2, where the Enhanced Mode acknowledgements were introduced. In addition, the solutions are used to illustrate receiving HL7 BC ACK generation, when receiving an invalid HL7 message.

HL7 Accept Acknowledgment Rules

The underlying material comes from the “Health Level 7 – Standard Version 2.3.1”, ANSI/HL7 V2.3.1—1999, April 14, 1999. HL7 standards documents are not free, and the terms of the copyright claim prevent reproduction of any part of the standard without written permission. I am paraphrasing a small part of the standard to foster understanding of the discussion in this Note.

HL7 v2.3.1, Chapter 2.12, “Application (Level 7) Processing Rules”, section 2.12.1, “Original and enhanced processing rules”, says this, about enhanced acknowledgements:

There are two parties to message exchange: an initiator and a responder.
Both initiator and responder can send and receive messages. The initiator first sends and then receives. The responder first receives then sends.

The initiator generates and sends a HL7 message.

The responder (enhanced acknowledgement) receives the message and “saves” it, accepting the responsibility for the message so that the sending system is no longer required to be in a position to re-send the message.

The receiving system checks the MSH-15 field to see if the sender requires Accept Acknowledgement to be sent back. If it does, the receiving system generates the Accept Acknowledgement message and sends it back.

Field MSH-15, Application Acknowledgement Type in the HL7 message, when not empty, can contain one of the following values:

AL	- Always send application acknowledgement
NE	- Never send application acknowledgement
ER	- Only send acknowledgements with errors/rejection
SU	- Only send acknowledgements with success

The accept acknowledgment is constructed as an instance of the general acknowledgment message, consisting of the MSH segment, MSA segment and possibly ERR segment, if the acknowledgement conveys error information.

The MSH segment in the response can be constructed as a copy of the MSH segment in the original message except:

MSH-7, Date/time of message	is the date/time of the acknowledgement generation
MSH-10, Message Control ID	is the message control id of the acknowledgment message
MSH-5, Receiving Application	is a copy of the MSH-3, Sending Application in the original message
MSH-6, Receiving Facility	is a copy of the MSH-4, Sending Facility in the original message

MSA-1, Acknowledgement Code, can be:

CA	(Commit Accept) message accepted for processing
CR	(Commit Reject) message rejected if it failed to pass certain MSH validations
CE	(Commit Error) message can not be accepted for any other reason

The following fields of the MSA segment have to be valued according to the associated rules:

MSA-2, Message Control ID	MSH-10-message control ID from MSH segment of incoming message
MSA-3, Text Message	Text description of error, if any

The standard also discusses the rules that are invoked if MSH-16, Application Acknowledgment field, is valued. This Note only deals with Accept Acknowledgment so MSH-16 is ignored.

Sample HL7 Data and HL7 v2 XSDs

While any HL7 v2.3.1 message, with MSH-15 correctly set to AL will do, there are sample ADT A03 messages which can be used for testing at http://mediacast.sun.com/users/Michael.Czapski-Sun/media/ADT_A03_raw.zip/details. Download these or obtain your own.

Obtain HL7 v2 XML Schema documents, from which we will need ADT A03 and related schemas, from http://wiki.open-esb.java.net/attach/HL7/hl7v2xsd.zip. Unzip it to a convenient location.

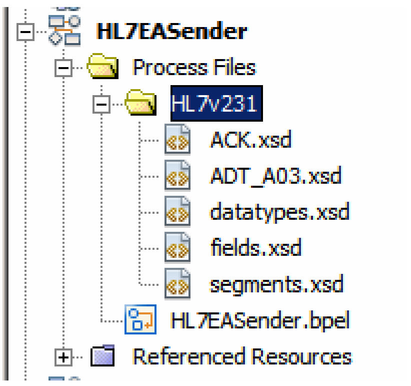
Construct HL7 Sender Project

The sender will poll a file system directory for a file containing HL7 v2.3.1 ADT A03 messages. For each message it will invoke a BPEL Process, which will send the message to an external system using the HL7 Binding Component (HL7 BC), will wait for the accept acknowledgment and will write the accept acknowledgement to a file in the same directory as the original message file. The acknowledgement file will be named using the MSH-10, Message Control ID, of the original message, with the timestamp and the literal ".hl7" appended.

Create a "New Project" → "SOA" → "BPEL Module project", named HL7EASender.

Create a sub-folder, HL7v231, in the "Process Files" folder. Right click the name of the subfolder and choose "New" → "External XML Schema Document(s)". Locate the ACK.xsd and ADT_A03.xsd in the hl7 v2.3.1 folder hierarchy hl7v2xsd/2.3.1, and choose them.

Supplementary files, included in ACK.xsd and ADT_A03.xsd, were added as well.



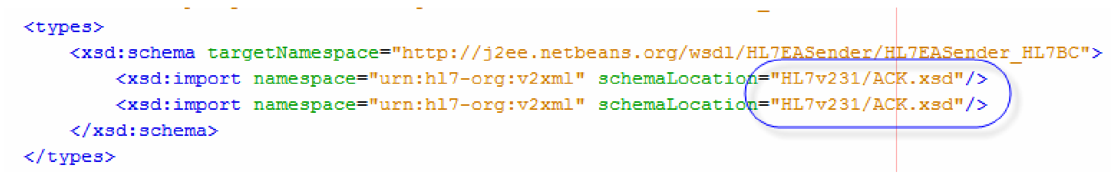
We expect to read one or more HL7 version 2.3.1 delimited ADT A03 messages and write out HL7 v 2.3.1 delimited ACK messages. This requires a File BC to read and write records. We could use one File BC configuration for reading and one for writing, but we can also use a single File BC configuration to both read records from a file and to write records to a different file in the same directory.

Create a "New" → "WSDL Document…", named HL7EASender_File, as follows:

WSDL Type:	Concrete WSDL Document
Binding:	FILE
Type:	Poll and Write Back Reply
Request Configuration:
	File Polling:
		File Name:	ADT_A03_raw_%d.hl7

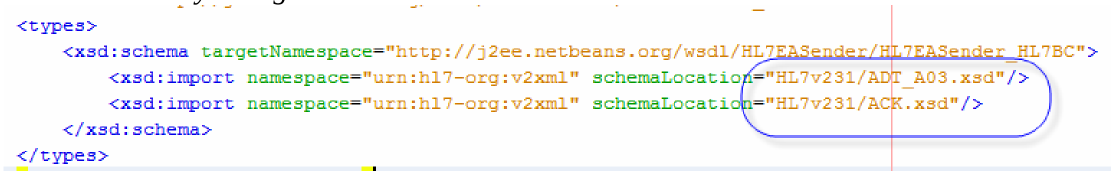
Polling Directory:	/GFESBv22Projects/GFESB_HA_LB/data (or whatever directory you unzipped files into)
Record Processing:	
Multiple Record:	true
Delimited By:	\r\n
Payload Processing:	
Message Type:	encoded data
XSD Element/Type:	ADT_A03
Encoded Type:	hl7encoder-1.0
Remove Trailing EOL:	true
Response Configuration:	
File Write:	
File Name:	output_%t.hl7
Payload Processing:	
Message Type:	encoded data
XSD Element/Type:	ACK
Encoded Type:	hl7encoder-1.0

Expect to hit trouble at this point. The NetBeans tooling in GlassFish ESB v2.2 apparently has a bug that is triggered when configuring a Request/Reply File BC WSDL. Validate XML to see an error.
We asked for ADT_A03 as the request and ACK as the response. What was produced contained, in source mode, something similar to the following (ADT_A03 missing and ACK relplicated):



```
<types>
    <xsd:schema targetNamespace="http://j2ee.netbeans.org/wsdl/HL7EASender/HL7EASender_HL7BC">
        <xsd:import namespace="urn:hl7-org:v2xml" schemaLocation="HL7v231/ACK.xsd"/>
        <xsd:import namespace="urn:hl7-org:v2xml" schemaLocation="HL7v231/ACK.xsd"/>
    </xsd:schema>
</types>
```

The ACK.xsd was referenced twice and ADT_A03.xsd not at all. You will need to manually change this to:



```
<types>
    <xsd:schema targetNamespace="http://j2ee.netbeans.org/wsdl/HL7EASender/HL7EASender_HL7BC">
        <xsd:import namespace="urn:hl7-org:v2xml" schemaLocation="HL7v231/ADT_A03.xsd"/>
        <xsd:import namespace="urn:hl7-org:v2xml" schemaLocation="HL7v231/ACK.xsd"/>
    </xsd:schema>
</types>
```

Once done, Validate XML again. It now should validate.

We will use the HL7 BC to interact with the external systems, the HL7 v2 receiver. GlassFish ESB v2.2 distribution does not include the HL7 BC, and as of that release, HL7 BC for v2.2 is no longer a free component, it seems. It is possible to download and install the HL7 BC component from the v2.1 distribution at http://download.java.net/jbi/binaries/installers/single-component/v2.1/nightly/latest/hl7bc-component-installer.jar.

If you are a support customer and have the license and support for the Healthcare Pack, then you can install the HL7 BC in the GalssFish ESB v2.2.

You may need to obtain the HL7 BC form the v2.1 distribution anyway because the HL7 BC in the Healthcare Pack, as at release of GlassFish ESB v2.2, implements incorrect Accept Acknowledgment logic so what is discussed here will not work. I put the issue, with the rationale, to the developers and they are thinking about it.

Create a “New” → “WSDL Document…”, named HL7EASender_HL7BC, as follows:

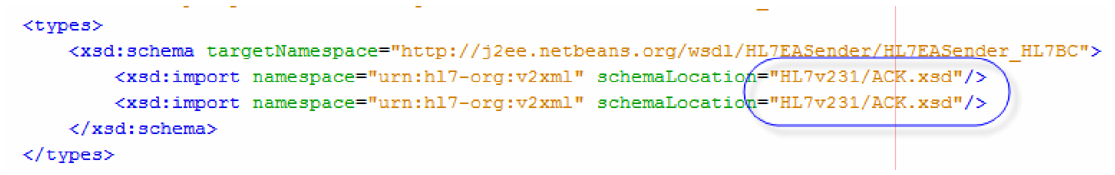
WSDL Type:	Concrete WSDL Document
Binding:	HL7
Type:	HL7 Version 2 – Outbound

Click the “Add two-way operation” button then enter and change, from default.

Operation Name:	opDoA03
Request Message:	ADT_A03
Response message:	ACK

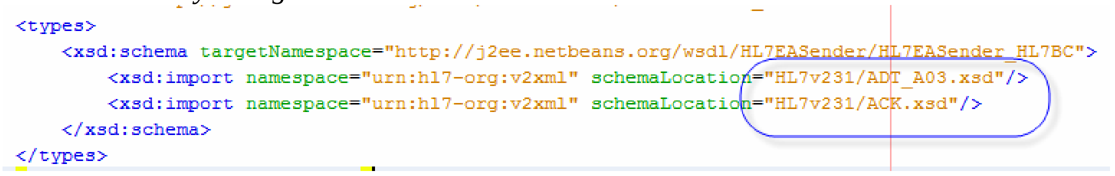
General Properties:
  Endpoint Properties:
    Location: hl7://locahost:34001 (or whatever host and port you use)
HL7 Version 2 Properties:
  HL7 Version 2 Properties:
    Acknowledgement Mode: enhanced
  MSH Properties:
    Validate MSH: enabled
    Sending Application: SystemA
    Sending Facility: HosA
  Communication Controls:
    Outbound Communication Controls:
      TIME_TO_WAIT_FOR_A_RESPONSE: 10000
        Recourse Action: Reset
      MAX_NO_RESPONSE: 1
        Recourse Action: Reset

Expect to hit trouble at this point. The NetBeans tooling in GlassFish ESB v2.2 apparently has a bug that is triggered when configuring a Request/Reply HL7 BC WSDL. Validate XML to see an error.
We asked for ADT_A03 as the request and ACK as the response. What was produced contained, in source mode:



```
<types>
    <xsd:schema targetNamespace="http://j2ee.netbeans.org/wsdl/HL7EASender/HL7EASender_HL7BC">
        <xsd:import namespace="urn:hl7-org:v2xml" schemaLocation="HL7v231/ACK.xsd"/>
        <xsd:import namespace="urn:hl7-org:v2xml" schemaLocation="HL7v231/ACK.xsd"/>
    </xsd:schema>
</types>
```

The ACK.xsd was referenced twice and ADT_A03.xsd not at all. You will need to manually change this to:

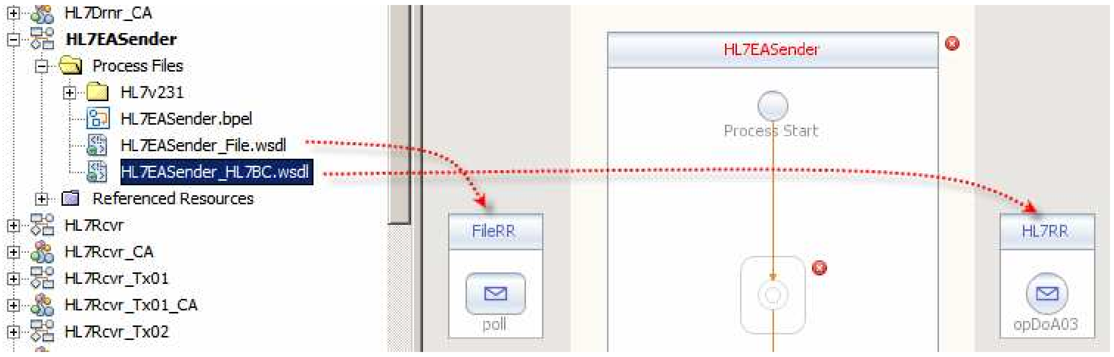


```
<types>
    <xsd:schema targetNamespace="http://j2ee.netbeans.org/wsdl/HL7EASender/HL7EASender_HL7BC">
        <xsd:import namespace="urn:hl7-org:v2xml" schemaLocation="HL7v231/ADT_A03.xsd"/>
        <xsd:import namespace="urn:hl7-org:v2xml" schemaLocation="HL7v231/ACK.xsd"/>
    </xsd:schema>
</types>
```

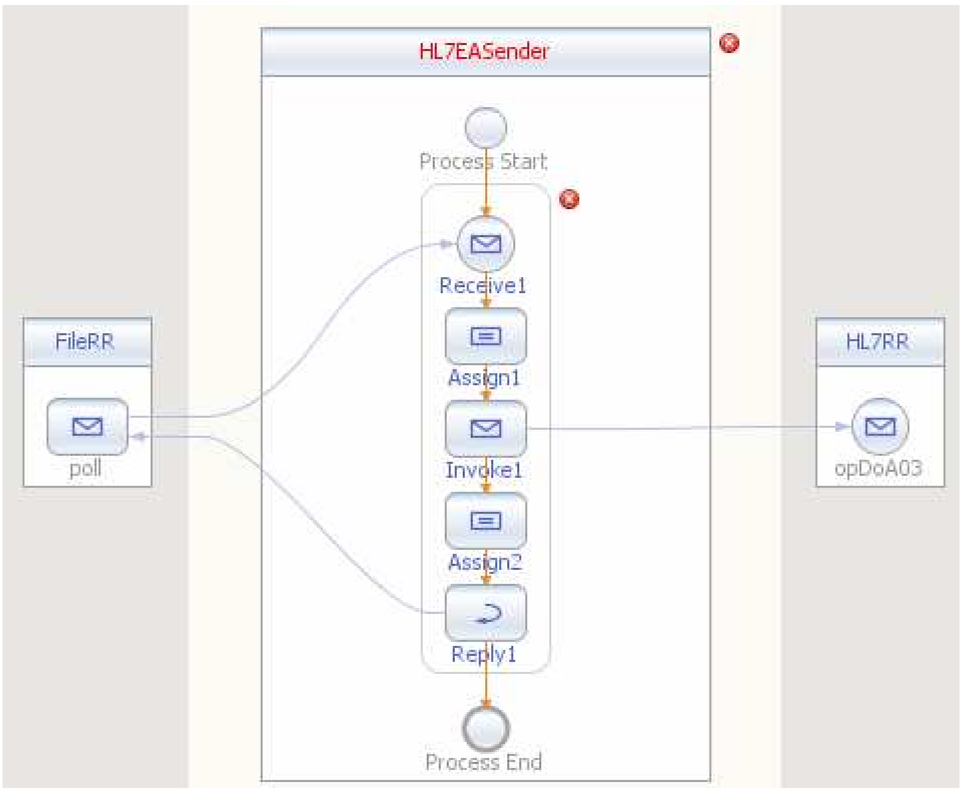
Once done, Validate XML again. It now should validate.

With both binding component configurations complete we can proceed to build the BPEL process to process HL7 messages.

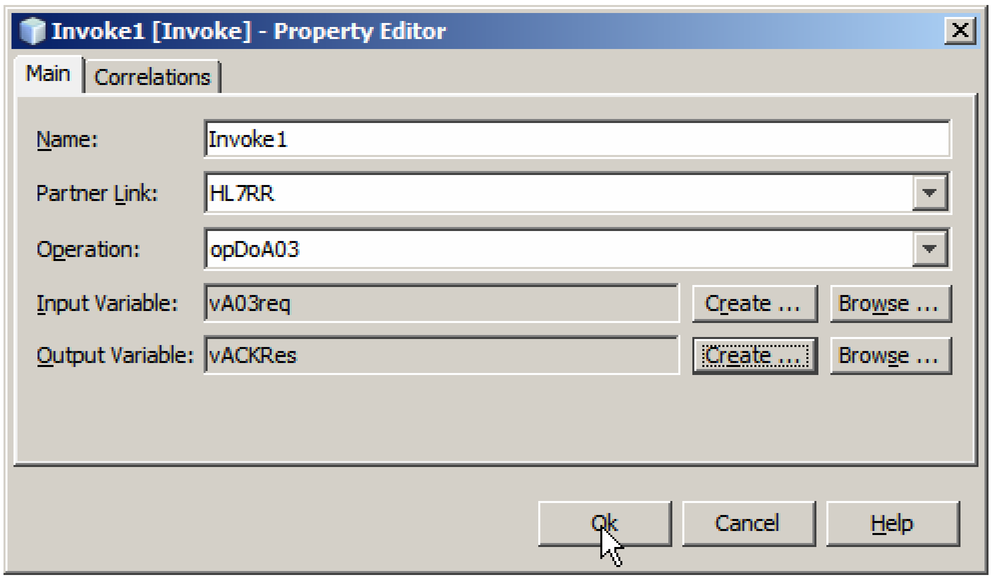
Open the HL7EASender.bpel process. Drag the HL7EASender_File WSDL onto the left hand swim line, renaming the partner link to FileRR, and HL7EASender_HL7BC to the right hand swim line, renaming the partner link to HL7RR.



Add Receive, Assign, Invoke, Assign and Reply, and connect as shown.



For Receive1 activity, create an Input Variable, vA03In.
For Invoke1 Activity, create an Input Variable vA03Req and an Output Variable vACKRes.


Invoke1 [Invoke] - Property Editor

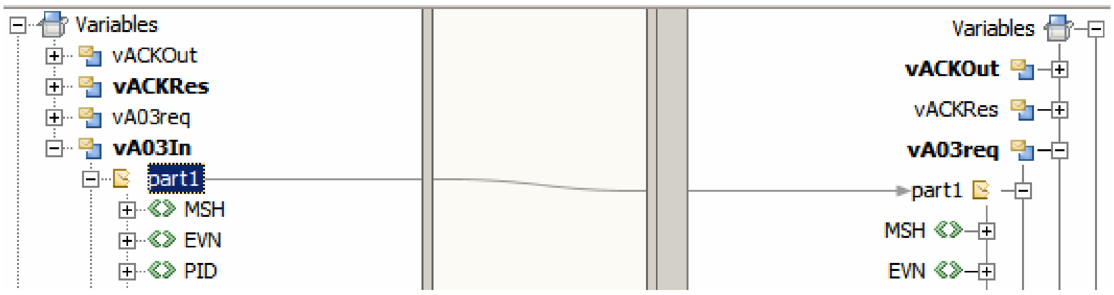
Main | Correlations

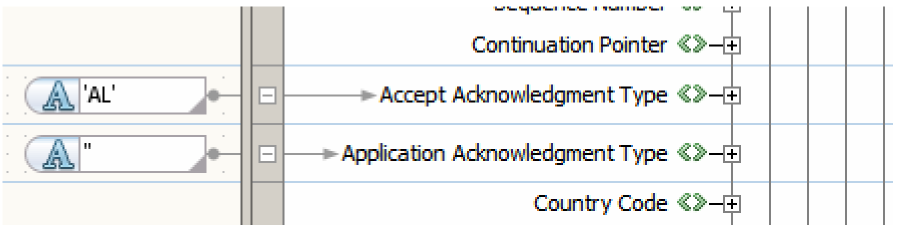
Name: Invoke1
Partner Link: HL7RR
Operation: opDoA03
Input Variable: vA03req
Output Variable: vACKRes

For Reply1 activity, create a Normal Response Output Variable vACKOut.

Open Assign1 activity in Mapper and map:

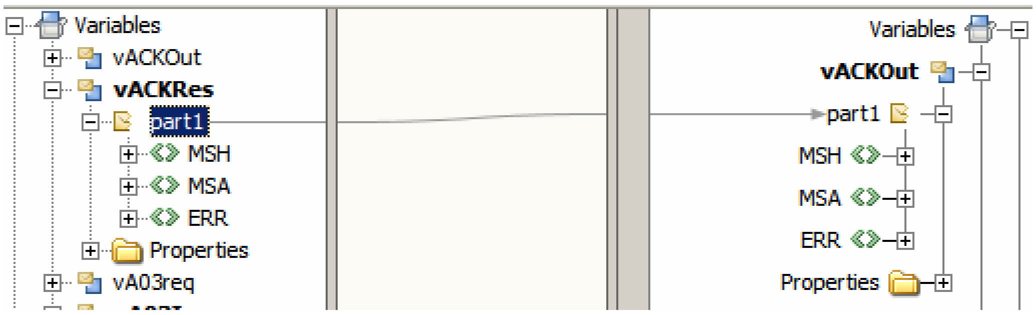
vA03In→part1 to vA03Req→part1
String literal “AL” to vA03Rep→part1→MSH→Accept Acknowledgement Type
Empty string to vA03Rep→part1→MSH→Application Acknowledgement Type





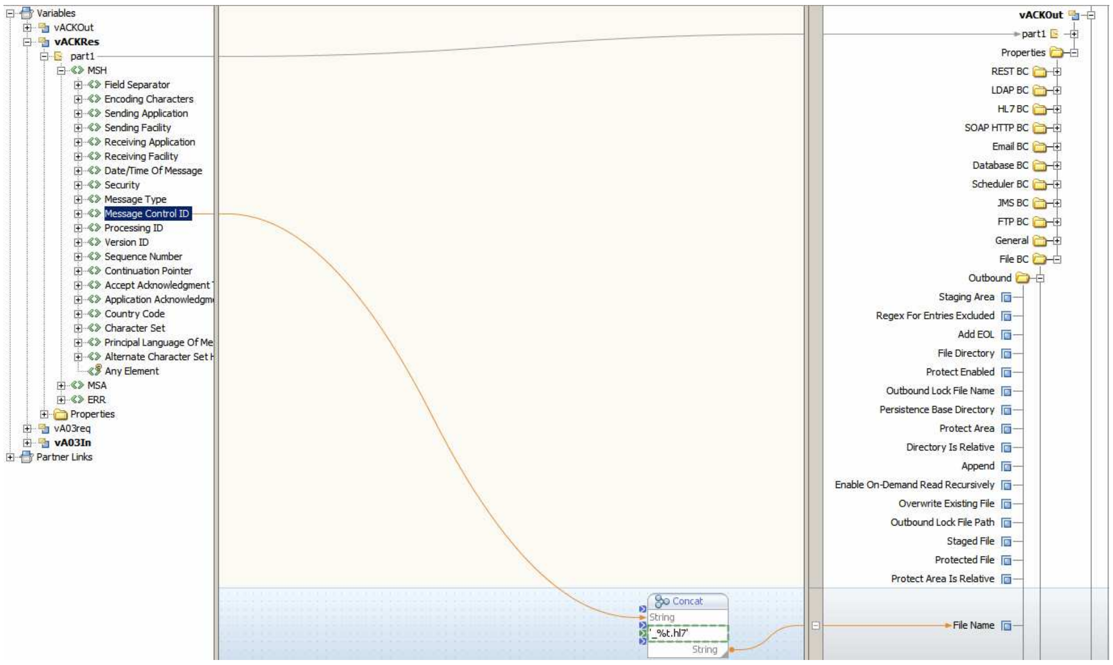
Switch to Design view and open Assign2 activity in Mapper.

Map vACKRes→part1 to vACKOut→part1



Add mapping for dynamically setting file name for the acknowledgement file, as follows:

Map concatenation of vA03In→part2→MSH→Message Control ID, and a String Literal “_02ACK_%t.hl7” to vACKOut→Properties→File BC→Outbound→File Name.

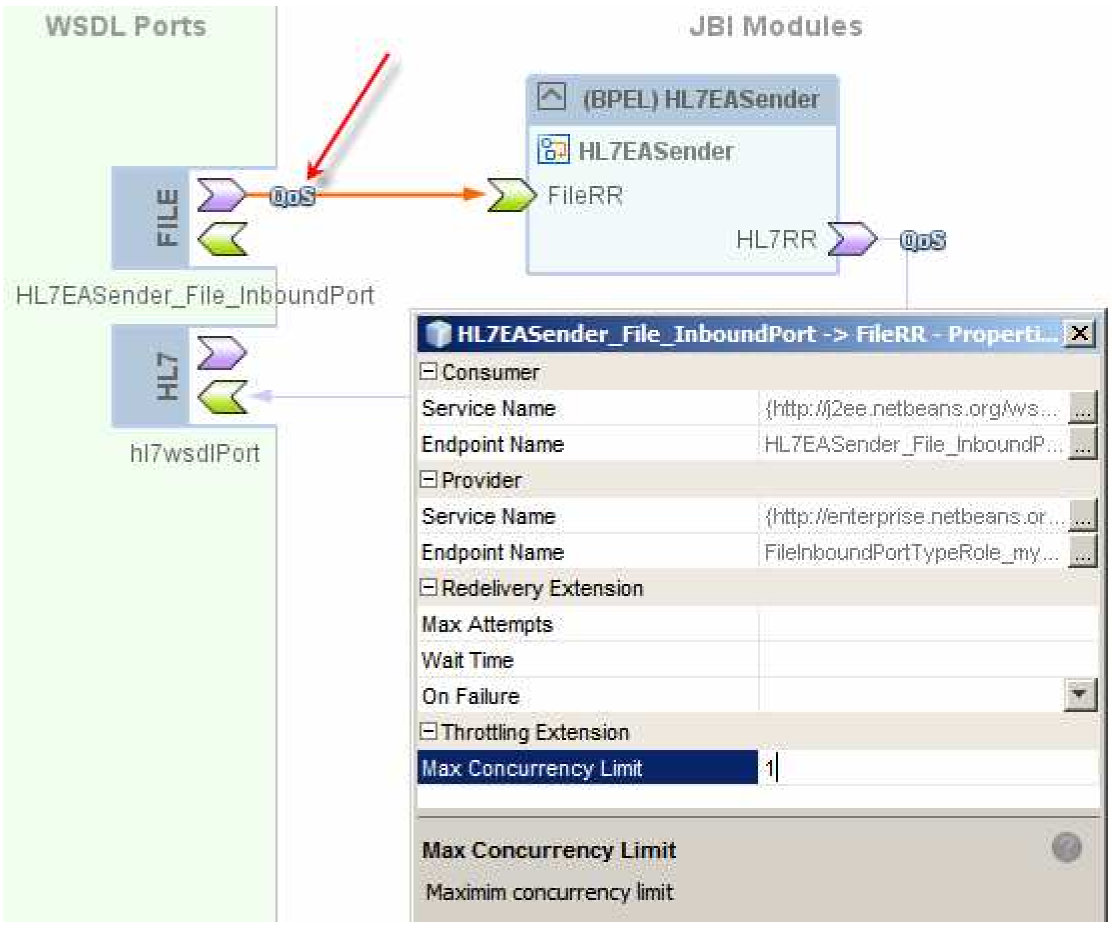


Build the project, to make sure it validates and builds.

Create “New Project”→”SOA”→”Composite Application”, named HL7EASender_CA.

Drag the BPEL Module project, HL7EASender, onto the “JBI Modules” part of the CASA Editor. Build the project.

Serialize message processing by setting the QoS Max Concurrency Limit to 1 on the File BC to BPEL Process link.



HL7EASender_File_InboundPort -> FileRR - Properti...

⊟ Consumer	
Service Name	{http://j2ee.netbeans.org/ws...
Endpoint Name	HL7EASender_File_InboundP...
⊟ Provider	
Service Name	{http://enterprise.netbeans.or...
Endpoint Name	FileInboundPortTypeRole_my...
⊟ Redelivery Extension	
Max Attempts	
Wait Time	
On Failure	
⊟ Throttling Extension	
Max Concurrency Limit	1

Max Concurrency Limit

Maximim concurrency limit

Deploy the project.

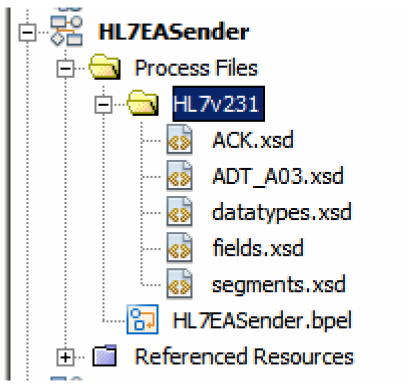
Construct HL7 Responder Project

The responder will receive a HL7 v2 message, write its content to the file system file, use its MSH segment to construct and acknowledgement, and send the acknowledgement back to the sender.

Create “New Project”→”SOA”→”BPEL Module Project”, named HL7EAResponder.

Create a sub-folder, HL7v231, in the “Process Files” folder. Right click the name of the subfolder and choose “New” → “External XML Schema Document(s)”. Locate the ACK.xsd and ADT_A03.xsd in the hl7 v2.3.1 folder hierarchy hl7v2xsd/2.3.1, and choose them.

Supplementary files, included in ACK.xsd and ADT_A03.xsd, were added as well.



We expect to receive HL7 version 2.3.1 delimited ADT A03 messages and send back HL7 v 2.3.1 delimited ACK messages. We also expect to be writing messages to a files.

Create a "New" → "WSDL Document…", named HL7EAResponder_HL7BC, as follows:

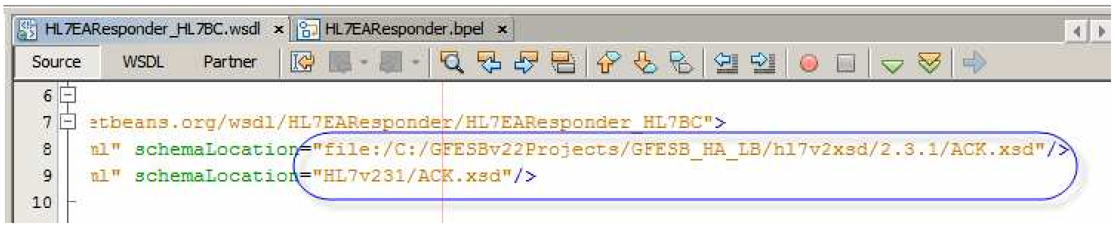
WSDL Type:	Concrete WSDL Document
Binding:	HL7
Type:	HL7 Version 2 – Inbound

Click the "Add two-way operation" button then enter and change, from default.

Operation Name:	opDoA03
Request Message:	ADT_A03
Request Message Type (MSH-9):	ADT^A03
Response message:	ACK

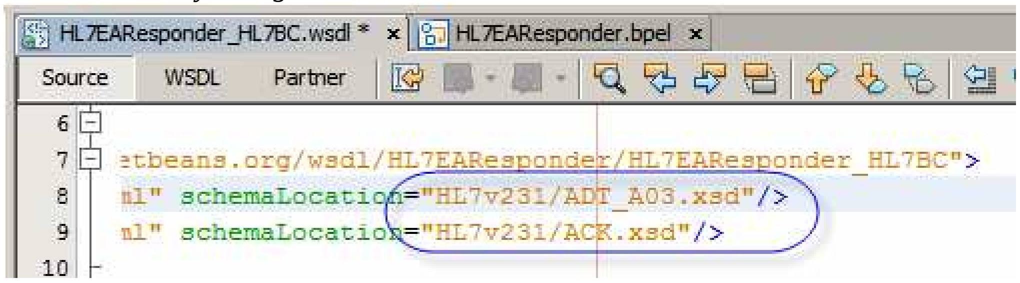
General Properties:
  Endpoint Properties:
    Location: hl7://locahost:34001 (or whatever host and port you use)
HL7 Version 2 Properties:
  HL7 Version 2 Properties:
    Acknowledgement Mode: enhanced
  MSH Properties:
    Validate MSH: enabled
    Sending Application: SystemA
    Sending Facility: HosA

Expect to hit trouble at this point. The NetBeans tooling in GlassFish ESB v2.2 apparently has a bug that is triggered when configuring a Request/Reply HL7 BC WSDL. Validate XML to see an error.
We asked for ADT_A03 as the request and ACK as the response. What was produced contained, in source mode:



```
etbeans.org/wsdl/HL7EAResponder/HL7EAResponder_HL7BC">
nl" schemaLocation="file:/C:/GFESBv22Projects/GFESB_HA_LB/hl7v2xsd/2.3.1/ACK.xsd"/>
nl" schemaLocation="HL7v231/ACK.xsd"/>
```

The ACK.xsd was referenced twice and ADT_A03.xsd not at all. You will need to manually change this to:



```
etbeans.org/wsdl/HL7EAResponder/HL7EAResponder_HL7BC">
nl" schemaLocation="HL7v231/ADT_A03.xsd"/>
nl" schemaLocation="HL7v231/ACK.xsd"/>
```

Once done, Validate XML again. It now should validate.

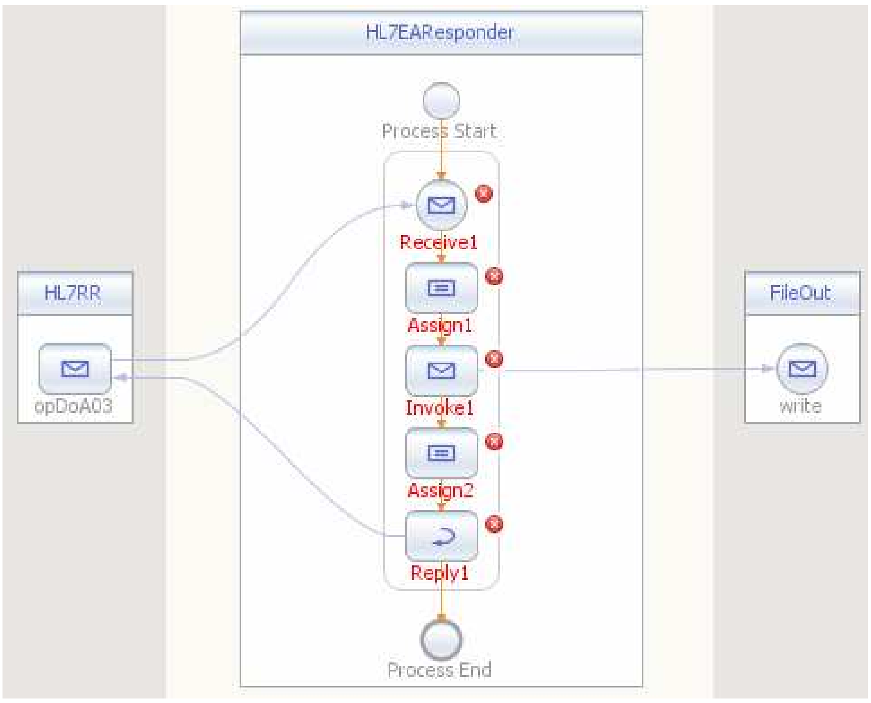
Create “New”→”WSDL Document…”, named HL7EAResponder_FileOut, as follows:

WSDL Type:	Concrete WSDL Document
Binding:	FILE
Type:	Write
Request Configuration:	
File Polling:	
File Name:	ADT_A03_processed_%d.hl7
Polling Directory:	/GFESBv22Projects/GFESB_HA_LB/data (or whatever directory you unzipped files into)
Payload Processing:	
Message Type:	encoded data
XSD Element/Type:	ADT_A03
Encoded Type:	hl7encoder-1.0

With both binding component configurations complete we can proceed to build the BPEL process to process HL7 messages.

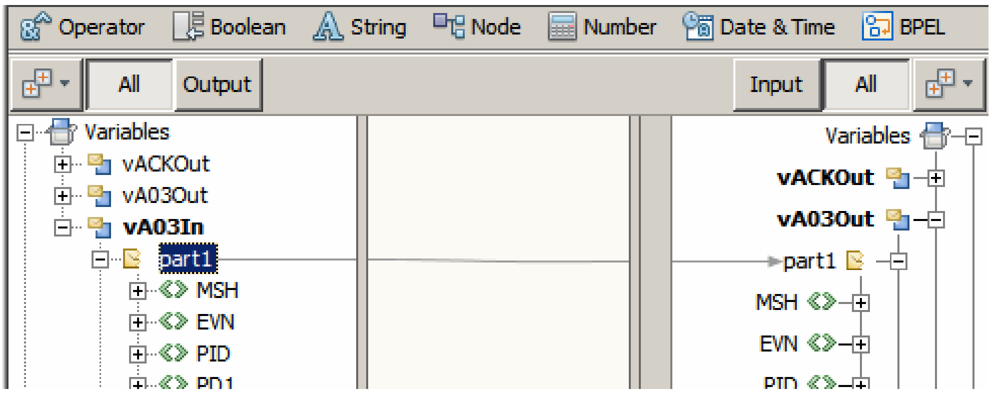
Open the HL7EAResponder.bpel process. Drag the HL7EAResponder_HL7BC WSDL onto the left hand swim line, renaming the partner link to HL7RR, and HL7EAResponder_FileOut to the right hand swim line, renaming the partner link to FileOut.

Add Receive, Assign, Invoke, Assign and Reply activities to the process model and connect as shown.



To Receive1 activity add Input Variable vA03In.
To Invoke activity add Input Variable vA03Out.
To Reply1 activity add Normal Response Output Variable vACKOut.

Select Assign1 and map vA03In→part1 to vA03Out→part1.



Add mapping for dynamically setting file name for the acknowledgement file, as follows:

Map concatenation of vA03In→part2→MSH→Message Control ID, and a String Literal “_01MSG_%t.hl7” to vA03Out→Properties→File BC→Outbound→File Name.

Switch back to Design mode and open Assign2 in Mapper.

Map MSH as follows:

vA03In→part1→MSH to vACKOut→part1→MSH

vA03In→part1→MSH→Sending Application to vACKOut→part1→MSH→Receiving Application

vA03In→part1→MSH→Sending Facility to to vACKOut→part1→MSH→Receiving facility

vA03In→part1→MSH→Receiving Application to vACKOut→part1→MSH→Sending Application

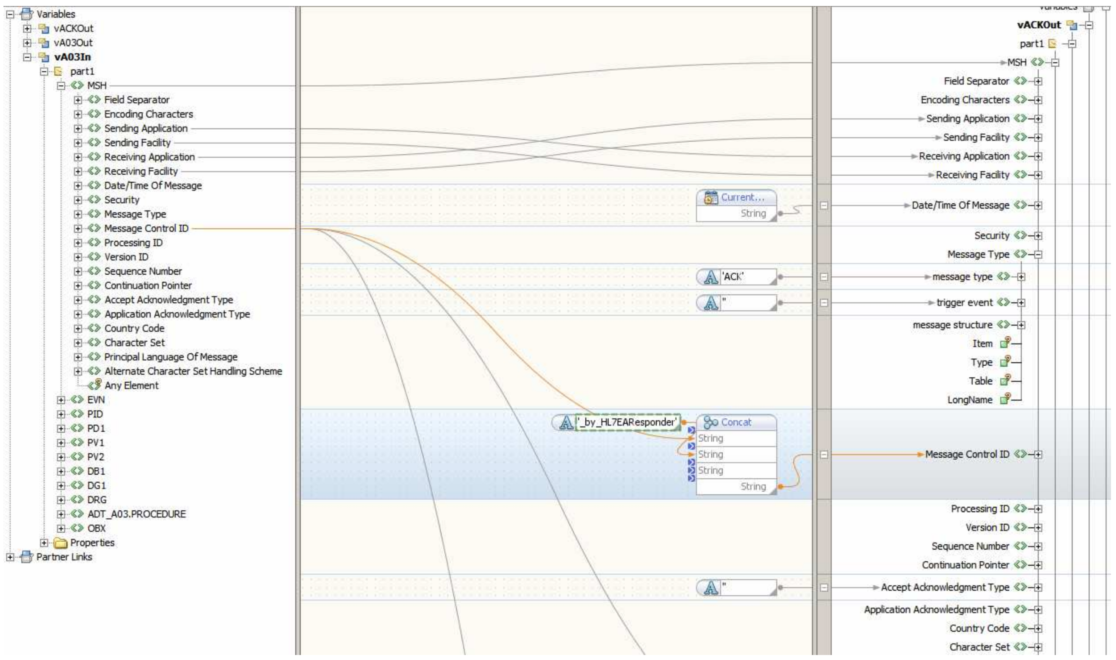
vA03In→part1→MSH→Receiving facility to vACKOut→part1→MSH→Sending Facility

BPEL→Current Date/Time literal to vACKOut→part1→MSH→Date/Time Of Message

String Literal “ACK” to vACKOut→part1→MSH→Message Type→message type
String Literal “” to vACKOut→part1→MSH→Message Type→trigger event

Concatenation of vA03In→part1→MSH→Message Control ID, and String Literal “_by_HL7EAResponder” to vACKOut→part1→MSH→Message Control ID

Literal String “” to vACKOut→part1→MSH→Accept Acknowledgement Type

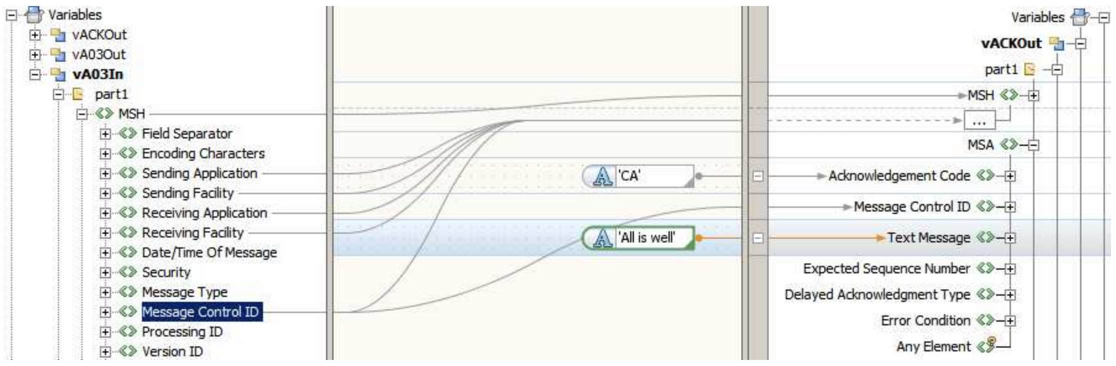


Map MSA as follows:

String Literal “CA” to vACKOut→part1→MSA→Acknowledgement Code

vA03In→part1→MSH→Message Control ID to vACKOut→part1→MSA→Message Control ID

String Literal “All is well” to vACKOut→part1→MSA→Text Message



Build the project.

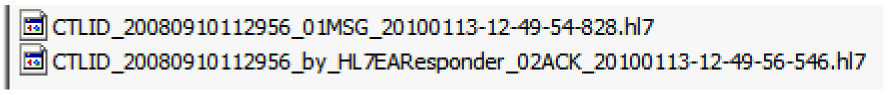
Create “New Project”→”SOA”→”Composite Application”, named HL7EAResponder_CA.

Drag the HL7EAResponder BPEL Module project onto the CASA Editor canvas, Build and Deploy.

Exercise the solution

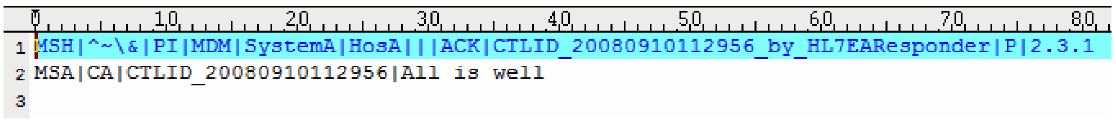
Submit the file, ADT_A03_raw_1.hl7, by copying to directory /GFESBv22Projects/GFESB_HA_LB/data (or whatever directory you have the File BC polling).

Once processing finishes you should see a couple of files in the data directory:



```
CTLID_20080910112956_01MSG_20100113-12-49-54-828.hl7
CTLID_20080910112956_by_HL7EAResponder_02ACK_20100113-12-49-56-546.hl7
```

Open the _02ACK_ file with a text editor and inspect the content.



```
MSH|^~\&|PI|MDM|SystemA|HosA|||ACK|CTLID_20080910112956_by_HL7EAResponder|P|2.3.1
MSA|CA|CTLID_20080910112956|All is well
```

Copy file ADT_A03_raw_1.hl7, renaming it to ADT_A03_raw_0.hl7. Use a Hex editor, or another editor that will not change carriage returns in the file to New Liens or some such garbage, and change the Processing ID “P” to “D” - MSH-11.

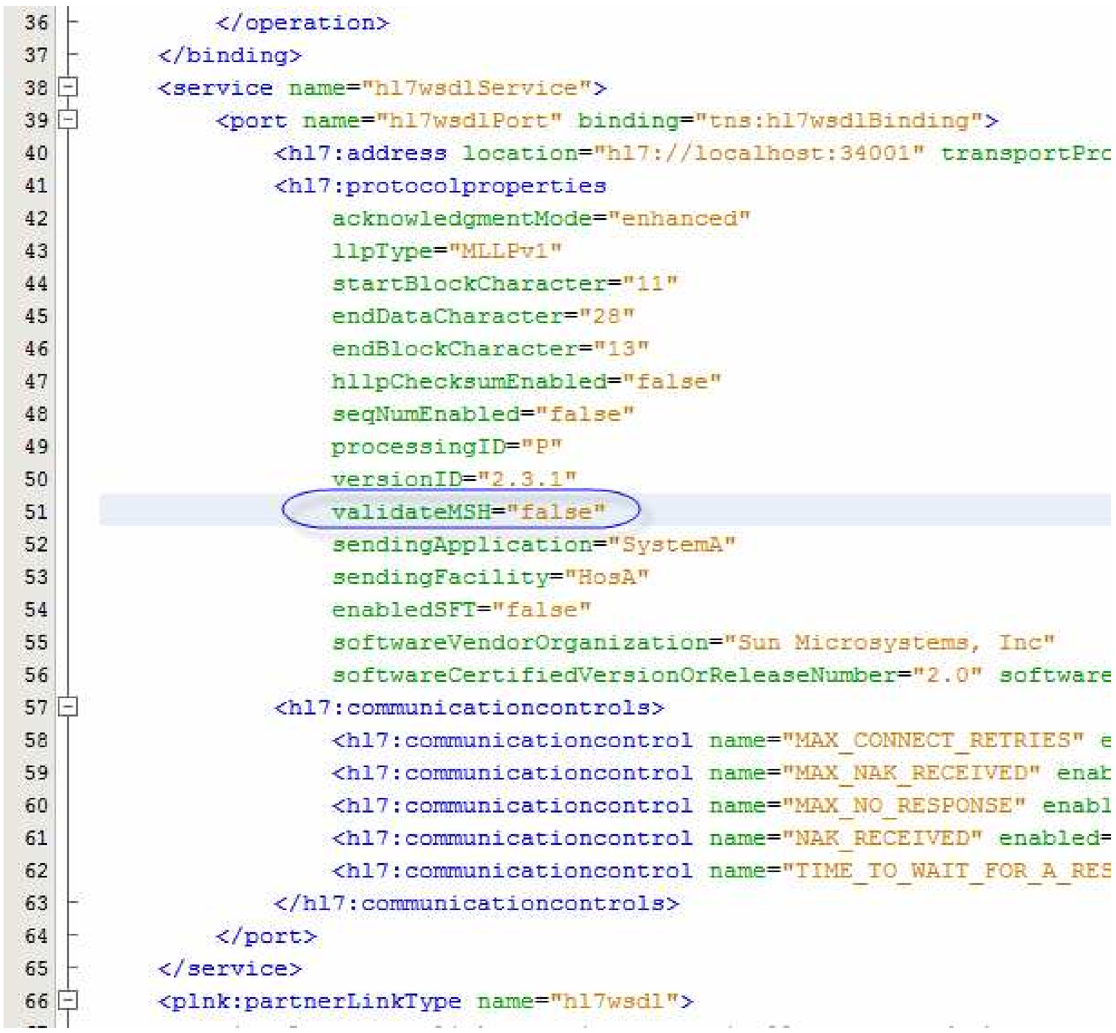
Submit this file to verify MSH validation Recall we configured this on the outbound and the inbound sides.

You should get a trace similar to this:

```
[#|2010-01-13T12:51:44.640+1100|INFO|sun-
appserver2.1|com.sun.jbi.hl7bc.extservice.HL7OutboundMessageValidationProcessor|_Threa
dID=86;_ThreadName=HL7BC-OutboundReceiver-3;|HL7BC-I0171: Validating the HL7 message
in Outbound|#]

[#|2010-01-13T12:51:44.640+1100|INFO|sun-
appserver2.1|com.sun.jbi.hl7bc.extservice.HL7OutboundMessageValidationProcessor|_Threa
dID=86;_ThreadName=HL7BC-OutboundReceiver-3;|HL7BC-I0164: MSH-1 field separator Value
is ||#]

[#|2010-01-13T12:51:44.640+1100|INFO|sun-
appserver2.1|com.sun.jbi.hl7bc.extservice.HL7OutboundMessageValidationProcessor|_Threa
dID=86;_ThreadName=HL7BC-OutboundReceiver-3;|HL7BC-I0165: MSH-2 encoding characters
value is ^~\&|#]

[#|2010-01-13T12:51:44.640+1100|INFO|sun-
appserver2.1|com.sun.jbi.hl7bc.extservice.HL7OutboundMessageValidationProcessor|_Threa
dID=86;_ThreadName=HL7BC-OutboundReceiver-3;|HL7BC-I0166: MSH-10 message control id is
CTLID_20080910112956|#]

[#|2010-01-13T12:51:44.640+1100|INFO|sun-
appserver2.1|com.sun.jbi.hl7bc.extservice.HL7OutboundMessageValidationProcessor|_Threa
dID=86;_ThreadName=HL7BC-OutboundReceiver-3;|HL7BC-I0167: MSH-11 processing id is D|#]

[#|2010-01-13T12:51:44.640+1100|SEVERE|sun-
appserver2.1|com.sun.jbi.hl7bc.extservice.HL7OutboundMessageValidationProcessor|_Threa
dID=86;_ThreadName=HL7BC-OutboundReceiver-3;_RequestID=579e40bc-3124-47a4-abcc-
ac8cf0f6718a;|HL7BC-E0289: MSH Validation failed; Processing IDs DO NOT match : D|#]

[#|2010-01-13T12:51:44.640+1100|SEVERE|sun-
appserver2.1|com.sun.jbi.hl7bc.extservice.HL7OutboundMessageValidationProcessor|_Threa
dID=86;_ThreadName=HL7BC-OutboundReceiver-3;_RequestID=579e40bc-3124-47a4-abcc-
ac8cf0f6718a;|HL7BC-E0285: MSH segment validation failed|#]

[#|2010-01-13T12:51:44.640+1100|SEVERE|sun-
appserver2.1|com.sun.jbi.hl7bc.OutboundMessageProcessor|_ThreadID=86;_ThreadName=HL7BC
-OutboundReceiver-3;_RequestID=579e40bc-3124-47a4-abcc-ac8cf0f6718a;|HL7BC-E0209:
Failed to process the message exchange with ID 246246245809757-33290-
134826403046250037 and exchange pattern http://www.w3.org/2004/08/wsdl/in-out due to
error: HL7BC-E0206: Unable to process message exchange 246246245809757-33290-
134826403046250037 as the normalized message has failed the MSH validation check. The
validation error code is: 202, and error message is: Unsupported Processing ID
java.lang.Exception: HL7BC-E0206: Unable to process message exchange 246246245809757-
33290-134826403046250037 as the normalized message has failed the MSH validation
check. The validation error code is: 202, and error message is: Unsupported Processing
ID

        at
com.sun.jbi.hl7bc.OutboundMessageProcessor.processRequestReplyOutbound(OutboundMessage
Processor.java:552)
        at
com.sun.jbi.hl7bc.OutboundMessageProcessor.processMessage(OutboundMessageProcessor.jav
a:318)
        at com.sun.jbi.hl7bc.OutboundAction.run(OutboundAction.java:66)
        at
java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.java:886)
        at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:908)
        at java.lang.Thread.run(Thread.java:619)
|#]
```

Message validation failed at the sending HL7 BC. Note that no acknowledgement was returned to the HL7EASender process. It got a Fault, which it could handle.

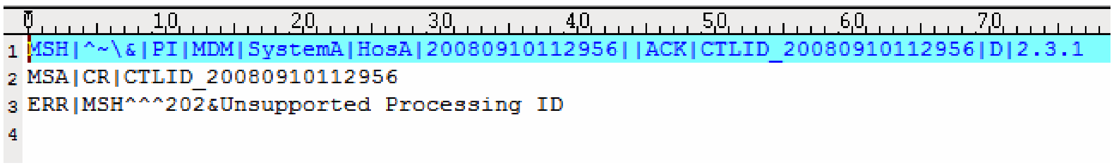
Modify the HL7EASender_HL7BC WSDL and disable MSH Validation.



```
            </operation>
        </binding>
        <service name="hl7wsdlService">
            <port name="hl7wsdlPort" binding="tns:hl7wsdlBinding">
                <hl7:address location="hl7://localhost:34001" transportPrc
                <hl7:protocolproperties
                    acknowledgmentMode="enhanced"
                    llpType="MLLPv1"
                    startBlockCharacter="11"
                    endDataCharacter="28"
                    endBlockCharacter="13"
                    hllpChecksumEnabled="false"
                    seqNumEnabled="false"
                    processingID="P"
                    versionID="2.3.1"
                    validateMSH="false"
                    sendingApplication="SystemA"
                    sendingFacility="HosA"
                    enabledSFT="false"
                    softwareVendorOrganization="Sun Microsystems, Inc"
                    softwareCertifiedVersionOrReleaseNumber="2.0" software
                <hl7:communicationcontrols>
                    <hl7:communicationcontrol name="MAX_CONNECT_RETRIES" e
                    <hl7:communicationcontrol name="MAX_NAK_RECEIVED" enab
                    <hl7:communicationcontrol name="MAX_NO_RESPONSE" enabl
                    <hl7:communicationcontrol name="NAK_RECEIVED" enabled=
                    <hl7:communicationcontrol name="TIME_TO_WAIT_FOR_A_RES
                </hl7:communicationcontrols>
            </port>
        </service>
        <plnk:partnerLinkType name="hl7wsdl">
```

Build and Deploy the HL7EASender, and submit the file with invalid Processing ID.

Note that an ACK was generated by the HL7 BC in the HL7EAReceiver project, without actually invoking the BPEL Module. The ACK is a negative ACK, complaining that the Processing ID in the incoming message is not valid.



```
MSH|^~\&|PI|MDM|SystemA|HosA|20080910112956||ACK|CTLID_20080910112956|D|2.3.1
MSA|CR|CTLID_20080910112956
ERR|MSH^^^202&Unsupported Processing ID
```

The message was not delivered to the BPEL process therefore it was not written out to the file.

The HL7 BC at the receiver side has the smarts to reject (CR) messages it does not like, as per HL7 v2.3.1 Enhanced Acknowledgment processing rules. While in this case the HL7 BC validates and rejects invalid messages (Validate MSH = true), the BPEL process could have done this explicitly as well. It is up to the designer to determine where acknowledgments are generated.

Summary

This Note walked through development of two BPEL Module-based solutions that cooperated in generating and processing Enhanced Accept Acknowledgements using HL7 v2.3.1 messages. In addition, the solutions were used to illustrate receiving HL7 BC ACK generation, when receiving an invalid HL7 message.