Oracle SOA Suite 11g R1 PS5

SOA Suite for healthcare integration Series

Implementing an A19 Query Processor

michael@czapski.id.au

January 2013

Table of Contents

Introduction

While most HL7 v2 messaging is of the store-and-forward kind, where a sender sends a message to a partner, receives a positive acknowledgement and forgets the transaction, there are circumstances where a sender wants to query the receiver and get and process a response. For ADT-related queries A19 transaction serves this function. A client application can send an A19 QRY message, with a patient identifier and assigning authority and request patient demographic information to be returned using the A19 ADR response message.

In this article we will develop and exercise an A19 Query Processor solution.

This article assumes that the reader has the SOA Suite for healthcare integration environment with all necessary components installed and ready to use. The Bill of Materials for such an environment and a discussion on where the components can be obtained is provided in the earlier article, "SOA Suite for healthcare integration Series - Overview of the Development Environment", to be found at http://blogs.czapski.id.au/2012/08/soa-suite-for-healthcare-integration-series-overview-of-the-development-environment.

Solution Overview

In this article the inbound HL7 v2 A19 QRY messages will be received by the inbound endpoint, will be routed to the SOA Composite to be processed and for a response A19 ADR

to be created. The A19 ADR response will be routed back to the inbound endpoint and will be forwarded by it to the original message sender as a response.

The inbound SOA Suite for healthcare integration adapter will perform the casting activity while translating the message from HL7 v2 delimited to the "equivalent" XML format.

The runtime components and their relationships are presented in Figure 1.

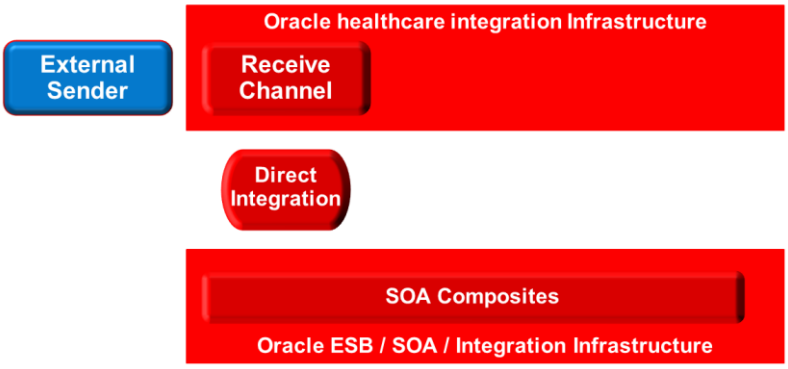


Figure 1 Runtime components

To summarise:

External Sender is the component which stands for a query client – the sender of HL7 A19 messages

The Oracle Healthcare Integration infrastructure is the part of the SOA Suite which deals with HL7 messages, acknowledgements, message tracking, message persistence, message processing KPI collection and so on, and the Receive Endpoint is the listening endpoint which receives messages

Direct Integration is the behind-the-scenes mechanism which hands messages over to an appropriate SOA Suite-based logic component for further processing or hands over messages produced by SOA Suite components to the healthcare integration infrastructure for processing (translation, sending, tracking).

ESB / SOA / Integration infrastructure hosts the SOA Composites and other logic components which process messages, whether HL7 v2 or not.

The solution components are depicted in Figure 2.

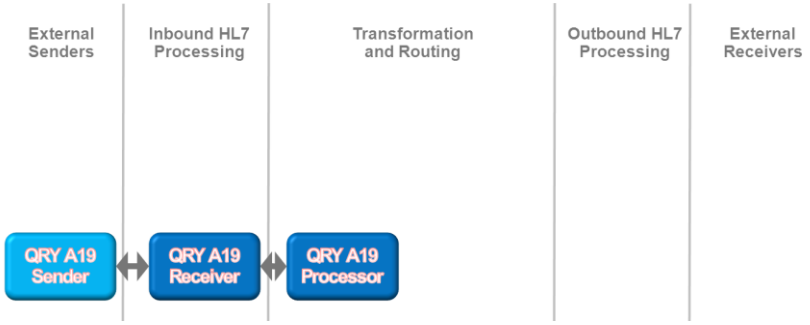


Figure 2 Solution Components

The diagram uses the convention which clearly separates the external systems, the SOA Suite for healthcare integration-specific components and generic SOA Suite components using the "swim-line" analogy.

QRY A19 Sender is the CMDHL7 sender tool, or another tool capable of sending HL7 v2 Delimited messages over TCP/IP using the MLLP protocol. It will send A19 Query messages and will receive and display A19 ADR responses.

QRY A19 Receiver is the SOA Suite for healthcare integration HL7 v2 listener.

QRY A19 Processor is a SOA Composite which receives the message from the HL7 listener, constructs the response message and passes it back to the SOA Suite for healthcare integration.

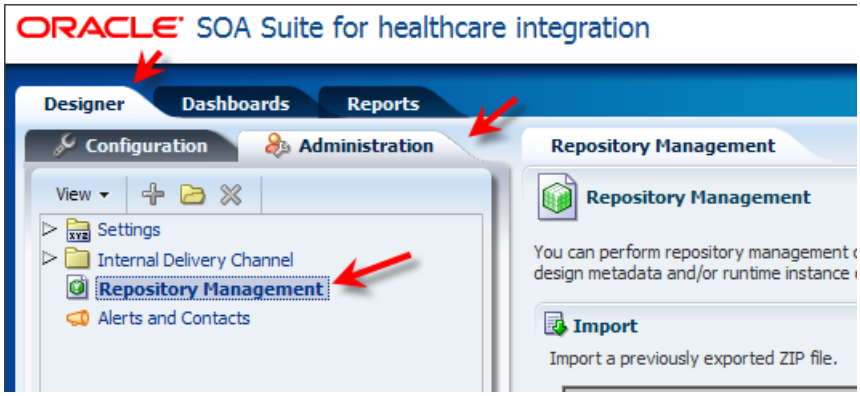
We need to construct and introduce A19 QEY and A19 ADR message strictures / documents, configure an A19 Query responder Endpoint and implement an A19 responder composite application. This we will do in the subsequent sections.

Clear Instances from the Repository

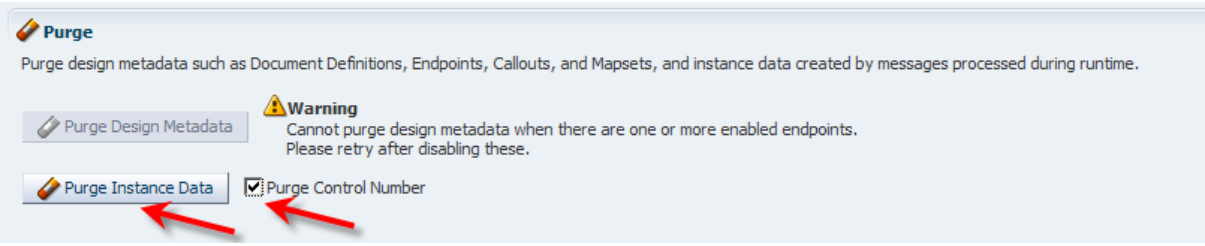
To start with a "clean slate" we will use the Healthcare Integration Console to clear out all old message tracking information. Bear in mind that this is irreversible and one should think carefully about clearing instance repository in production systems.

☐ Start the Healthcare Integration Console, http://localhost:7001/healthcare, and log in with your credentials, perhaps weblogic/welcome1

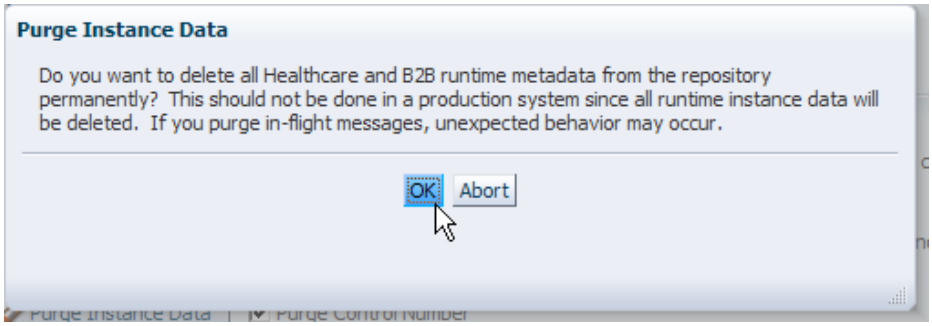
☐ Click "Designer" Tab, click "Administration" Tab and double-click on the "Repository Management" node



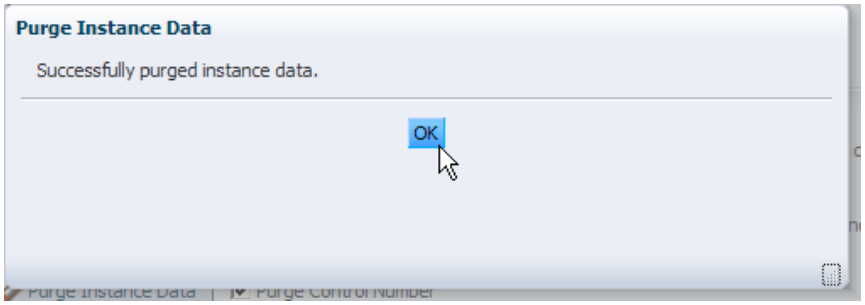
☐ Check the "Purge Control Number" checkbox, if appropriate, and click the "Purge Instance Data" button



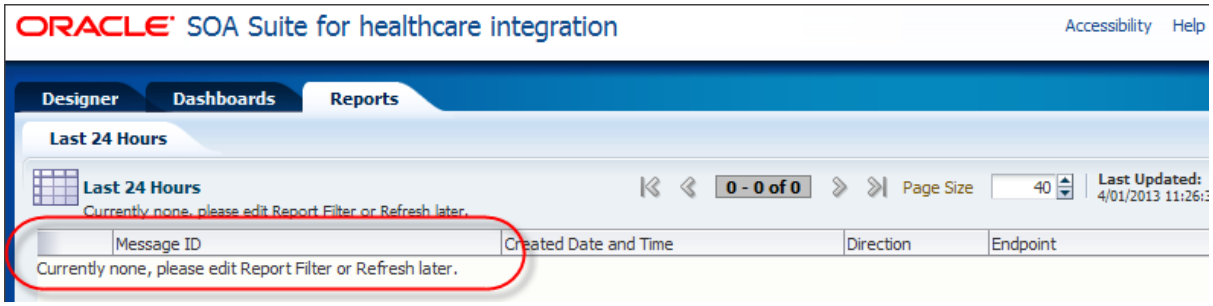
☐ Click "OK" to purge messages and message tracking information from the repository



☐ Once the process is completed, dismiss the wizard by clicking "OK"



☐ Click on the "Reports" Tab, look at the last 24-hours report and note that there are no messages in the repository

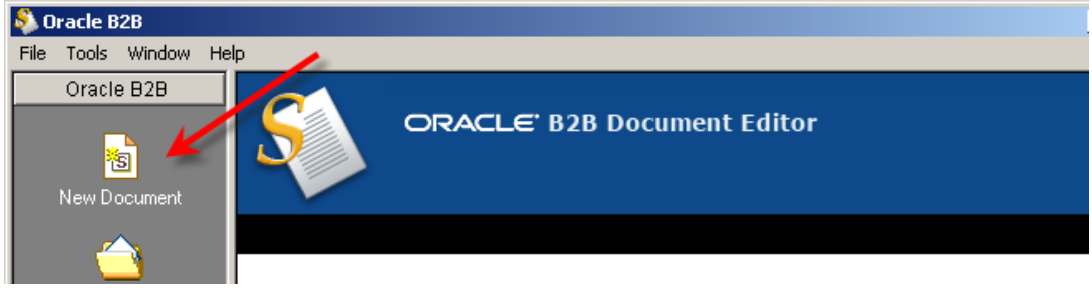


Add A19 QRY and A19 ADR Documents

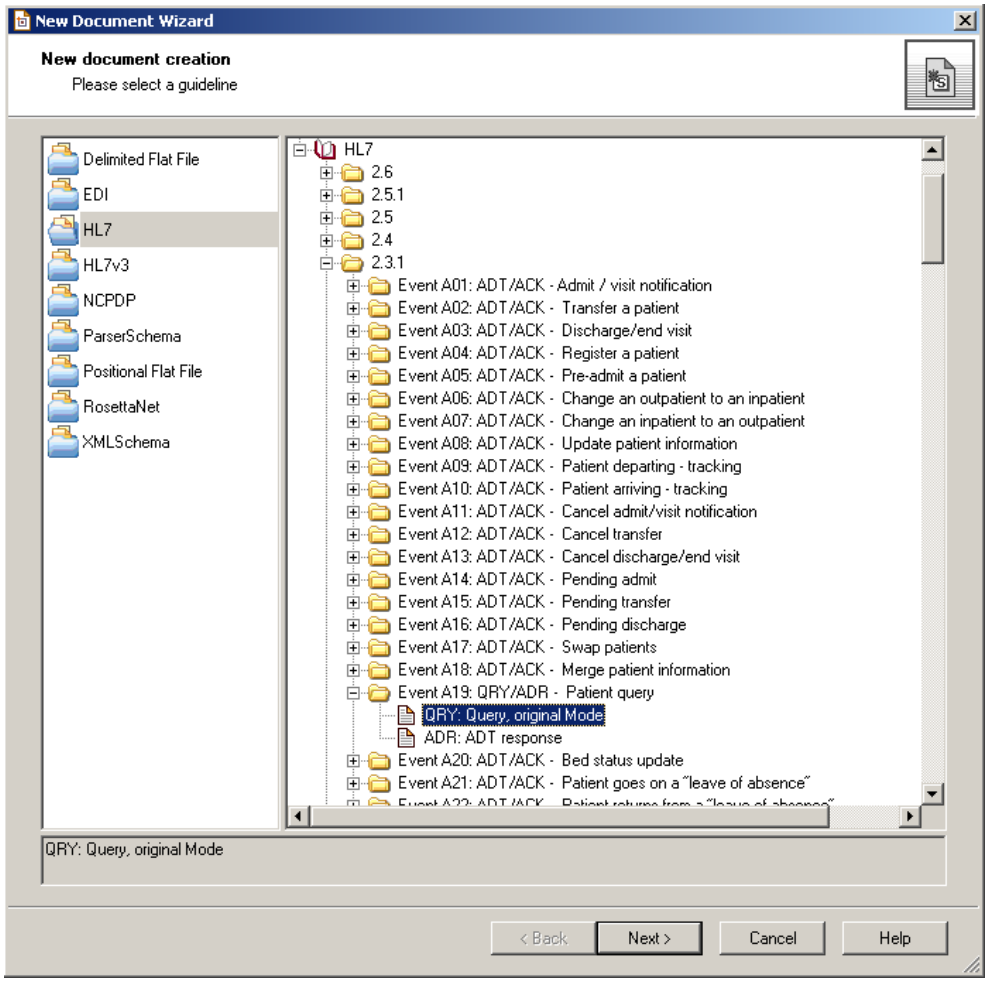
Since we have not dealt with A19 queries before we don't have the requisite message structured / documents, which to use. We will use the Oracle Document Editor to retrieve from the library and export for our use both of these documents.

☐ Start the "Oracle B2B Document Editor"

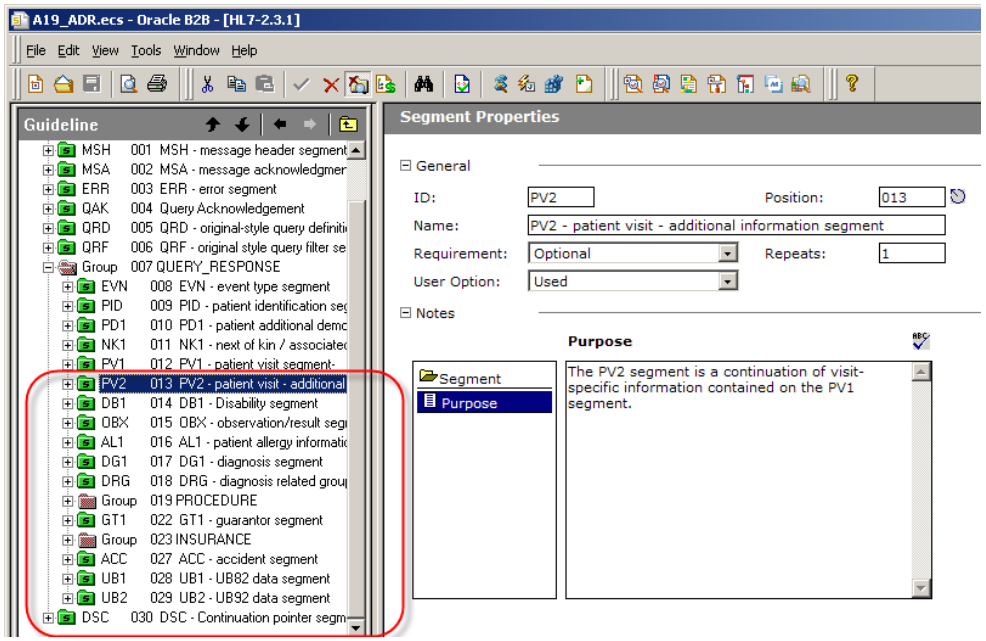
☐ Click the "New Document" icon



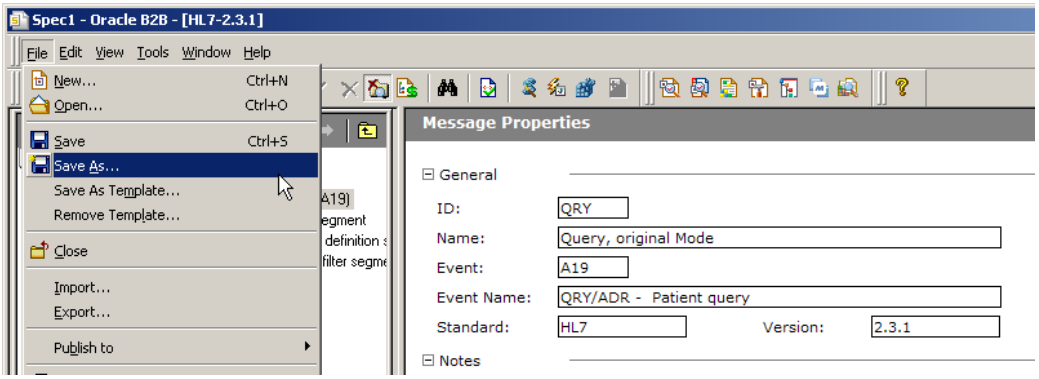
☐ Expand the "HL7" → "2.3.1" → "Event A19: QRY/ADR – Patient Query" node, double-click the "QRY: Query, original Mode" node then click "Next" and "Finish"



☐ Expand the QUERY_RESPONSE Group node and delete all segments starting with PV2 to the end of the group and the DSC segment



☐ Pull down the "File" menu and choose "Save As"



☐ Save the ecs file as A19_QRY.ecs

☐ Pull down the "File" menu and choose "Export…"

☐ Select "Oracle B2B 2.0" and click "Next", "Next" and "Finish" to save the XSD file, A19_QRY.xsd

☐ Repeat the process to create a new document A19_ADR.ecs and A19_ADR.xsd

☐ Close the "Oracle B2B Document Editor"

☐ Start the "Healthcare Integration Console", http://localhost:7001/healthcare

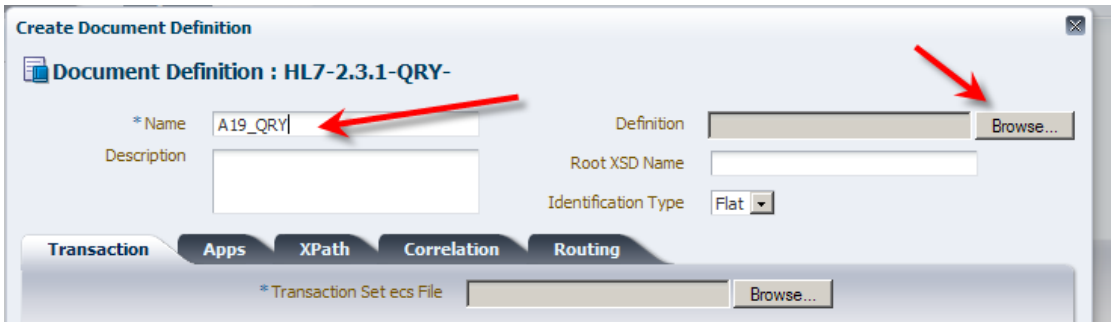
☐ Expand the "Design" → "Document Protocol" node tree through "HL7" → "2.3.1"

☐ Right-click on the node "2.3.1", choose "Create", enter "QRY" as document type name and click "OK"

☐ Right-click on the node "2.3.1", choose "Create", enter "ADR" as document type name and click "OK"

☐ Right-click the node "QRY" and choose "Create"

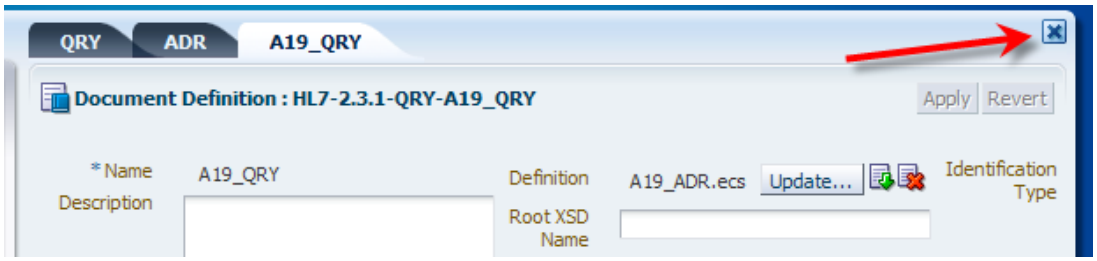
☐ Enter "A19_QRY" into the Name field of the "Create Document " dialog box, click the "Browse" button alongside the "Definition" label, locate the XML Schema Definition file, A19_QRY.xsd, and select it



☐ Click the "Browse" button alongside the "Transaction Set ecs File" label, locate the ECS file, A19_QRY.ecs, and select it

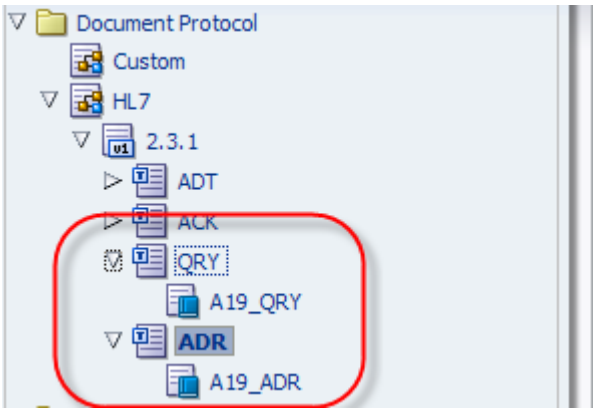
☐ Click "OK" to complete the dialogue

☐ Use the "Close" "Button" to close all open Tabs



☐ Repeat the process to add the A19_ADR document type under the "HL7" → "2.3.1" → "ADR" node, using A19_ADR.xsd and A19_ADR.ecs files

Our document hierarchy should now look like that shown in the illustration.

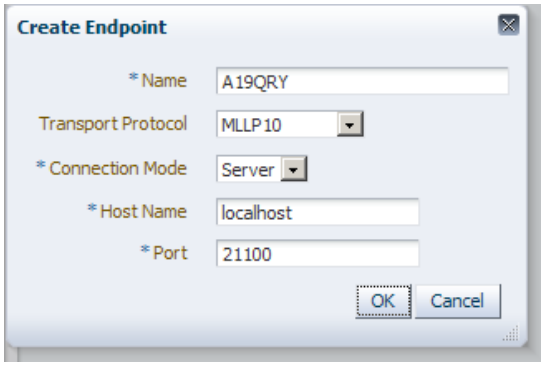


Configure A91Processor endpoint

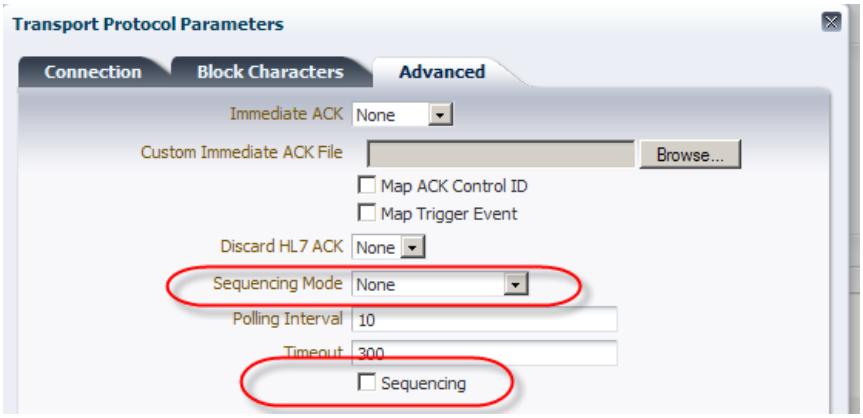
☐ Collapse the "Document Protocol" hierarchy and expand the "Endpoint" hierarchy

☐ Right-click on the "Endpoint" node and choose "Create"

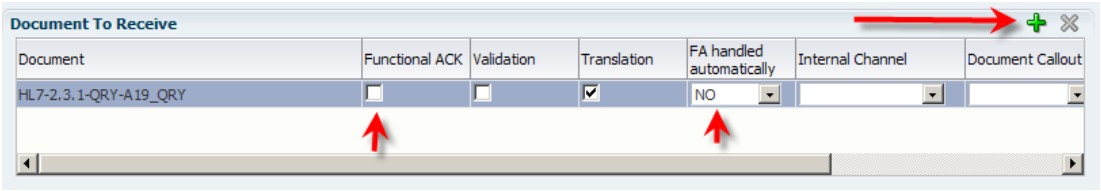
☐ Name the endpoint "A19QRY", configure it as a server which uses the MLLP 1.0 protocol and listens on localhost on port 21100



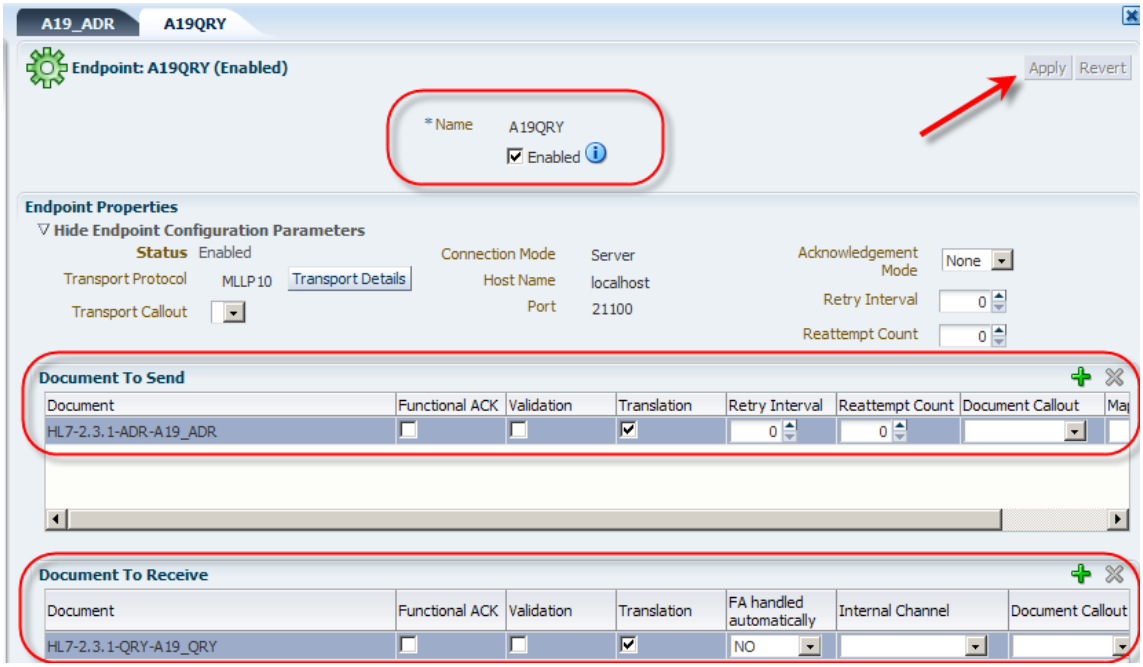
☐ Click the "Transport Details" button, click the "Advanced" Tab, uncheck the "Sequencing" checkbox, set the "Sequencing Mode" to "none" and click "OK"



- ☐ Check the "Enabled" checkbox
- ☐ Click the "Add" button (+) alongside and to the right of the "Document to Receive" label
- ☐ Select "A19_QRY" document type from the hierarchy and click "OK"
- ☐ Uncheck the "Functional ACK" button and choose "NO" for "FA handled automatically – we want the endpoint to require the SOA Composite to provide the functional ack whit=ch the endpoint will return to the message sender



- ☐ Click the "Add" button (+) alongside and to the right of the "Document to Send" label
- ☐ Select "A19_ADR" document type from the hierarchy and click "OK"
- ☐ Click "Apply" to deploy the configured endpoint



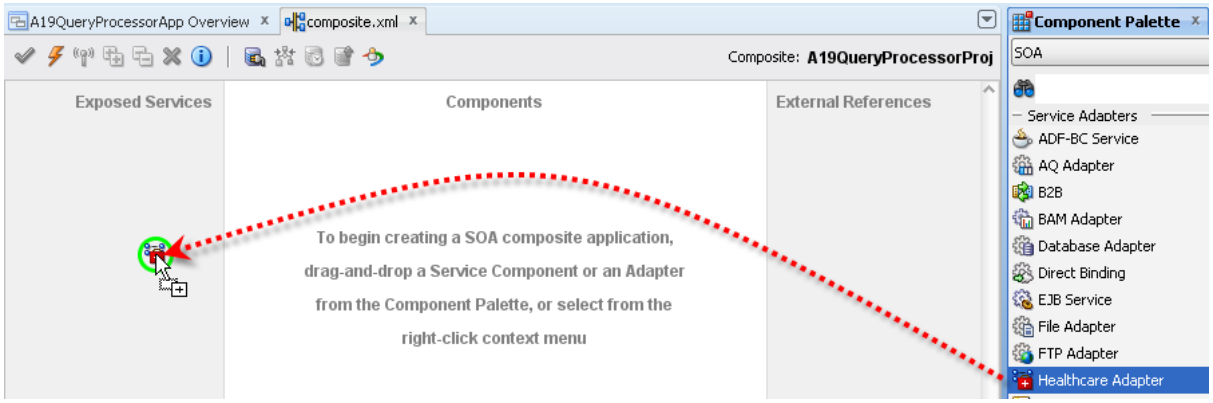
Implement A19 Processor Composite Application

With the endpoint configured and deployed we will now develop the SOA Composite to receive the query and produce the response

☐ Start JDeveloper Studio

☐ Create a new SOA application, "A19QueryProcessorApp" and a new project "A19QueryProcessorProj" with an empty composite – we did this often enough in this article series to not require detailed steps, right?

☐ Drag the "Healthcare Adapter" from the "Component Palette" → "SOA" → "Service Adapters" list to the "Exposed Services" swim line



☐ Name the service "HCI_QRY_In" and click "Next"

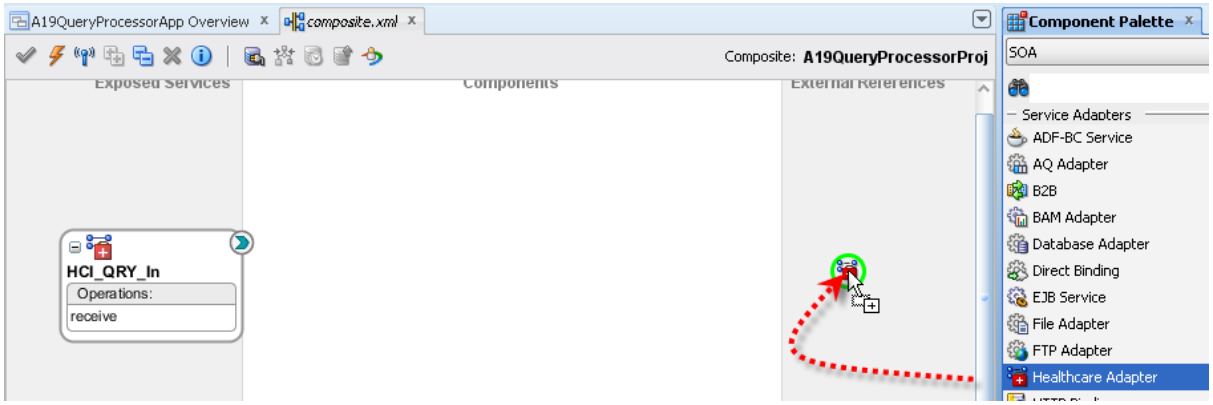
☐ Choose "AppServer Connection" and click "Next"

☐ Choose "receive" "Operation" and click "Next"

☐ Leave "Document Definition Handling" at default and click "Next"

☐ Choose "A19_QRY" document form the hierarchy, then click "Next" and "Finish" – the receiving side of the adapter is configured

☐ Drag the "Healthcare Adapter" from the "Component Palette" → "SOA" → "Service Adapters" list to the "External References" swim line

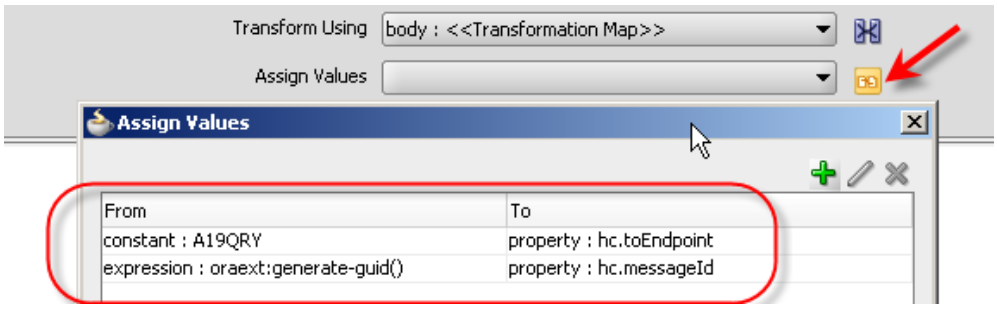


☐ Name the service "HCI_ADR_Out" and click "Next"

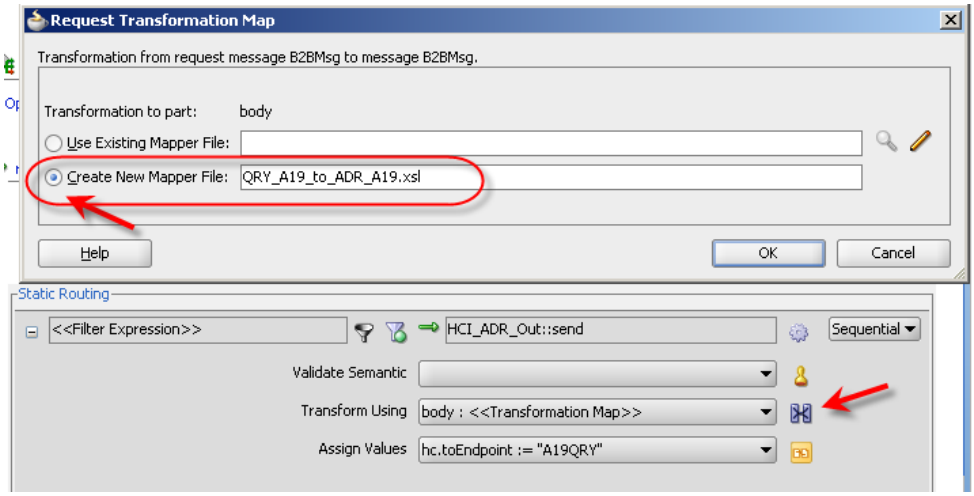
☐ Choose "AppServer Connection" and click "Next"

- ☐ Choose "send" "Operation" and click "Next"
- ☐ Leave "Document Definition Handling" at default and click "Next"
- ☐ Choose "A19_ADR" document form the hierarchy, then click "Next" and "Finish"
- ☐ Drag the "Mediator" component from the "Component Palette" → "SOA" → "Service Components" to the "Components" swim line
- ☐ Name the mediator "QRY2ADRMediator"
- ☐ Connect the adapters to the mediator component

- ☐ Double-click the mediator component to open the mplan and add the following value assignments
    - Constant "A19QRY" to property "hc.toEndpoint"
    - Expression, function "generate-guid" to property "hc.messageId"

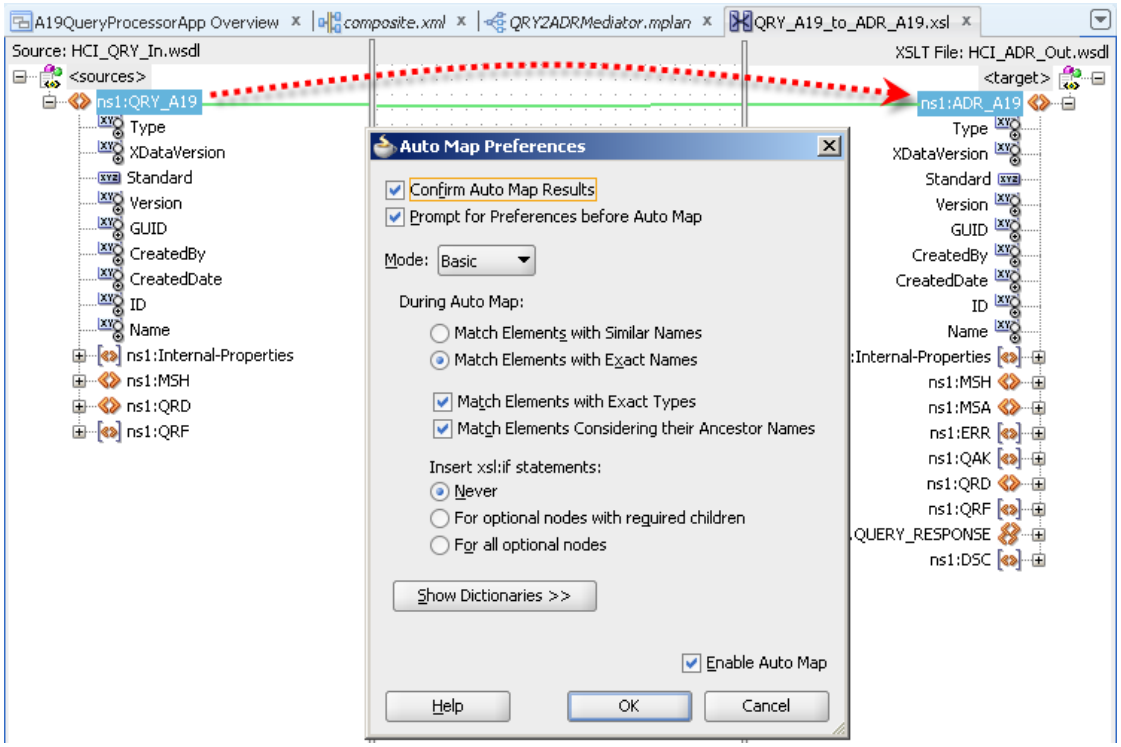


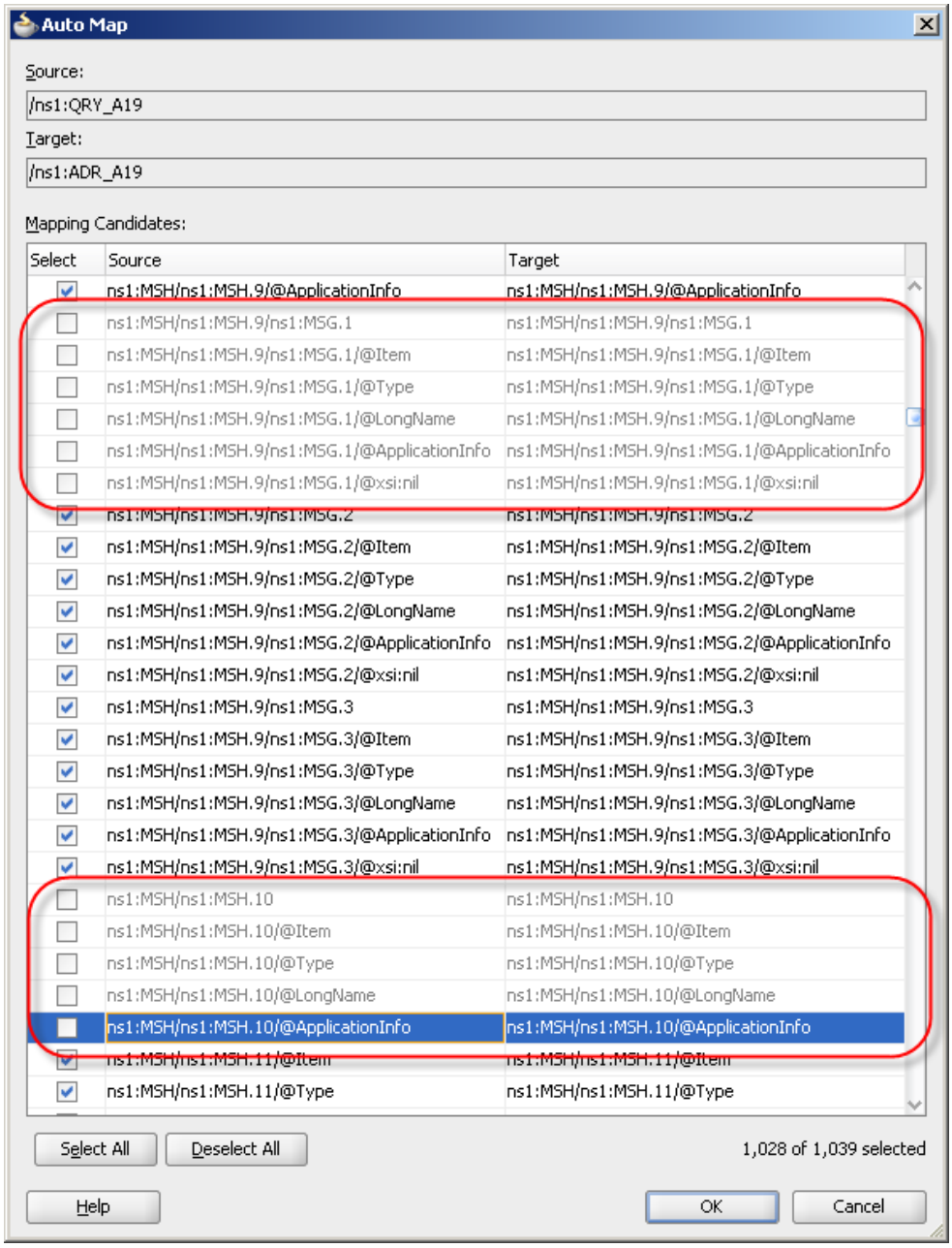
- ☐ Create a new mapping file, "QRY_A19_to_ADR_A19.xsl"



- ☐ Automap at the top level from QRY_A19 node to ADR_A19 node, except for the

MSH.9.MSG.1 and MSH.10, which will be mapped manually

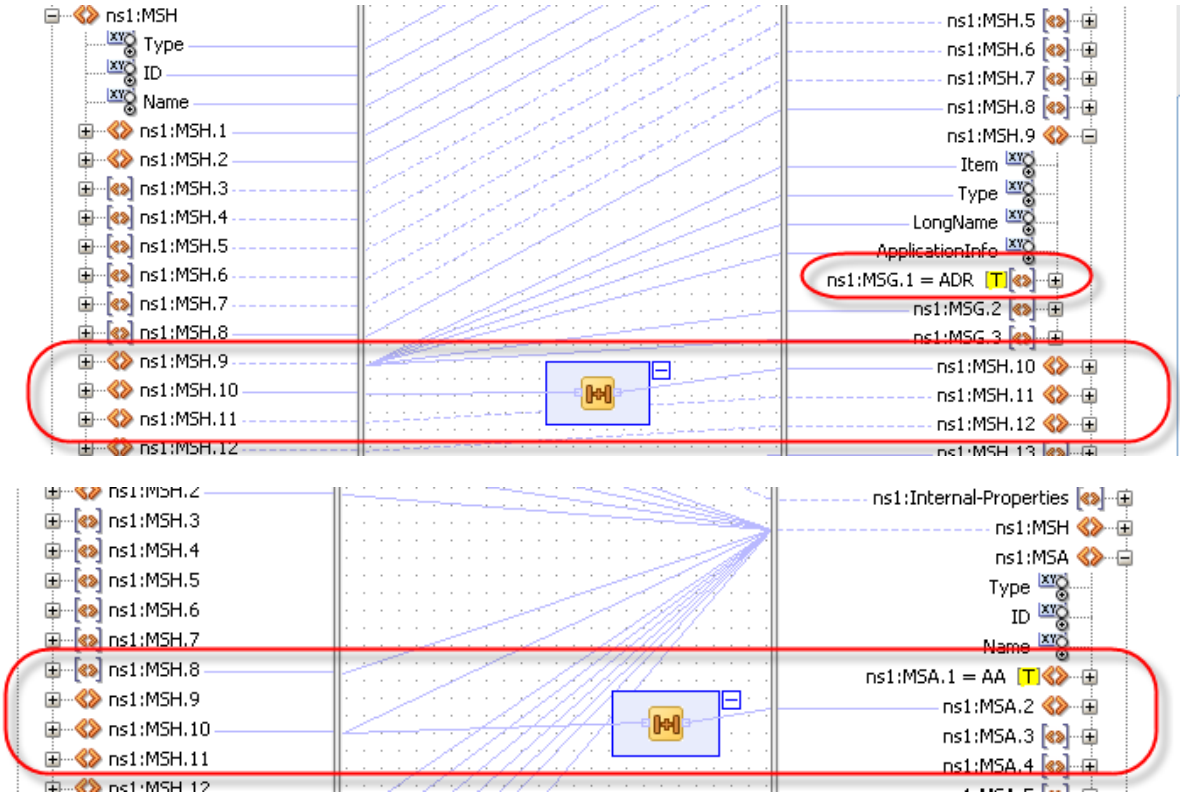




☐ Map from the QRY message to the ADR message as shown in the table below – note that concatenating the empty string, '', is necessary to coerce data type assignment to the string type and avoid spurious error notifications

From	To
Literal ADR	ADR_A19-->MSH-->MSH.9-->MSG.1
QRY_A19-->MSH-->MSH.10	concat('ACK_',ADR_A19-->MSH-->MSH.10)
Literal AA	ADR_A19-->MSA-->MSA.1
concat(QRY_A19-->MSH-->MSH.10,'')	ADR_A19-->MSA-->MSA.2
concat(QRY_A19-->QRD-->QRD.8-->XCN.1,'')	ADR_A19-->QUERY_RESPONSE-->PID-->PID.3-->CX.1
QRY_A19-->QRD-->QRD.8-->XCN.10→HD.1	ADR_A19-->QUERY_RESPONSE-->PID-->PID.3-->CX.4→HD.1
Smith	ADR_A19-->QUERY_RESPONSE-->PID-->PID.5-->XPN.1-->FN.1
John	ADR_A19-->QUERY_RESPONSE-->PID-->PID.5-->XPN.2
19610111	ADR_A19-->QUERY_RESPONSE-->PID-->PID.7-->TS.1
M	ADR_A19-->QUERY_RESPONSE-->PID-->PID.8
0908070605	ADR_A19-->QUERY_RESPONSE-->PID-->PID.19

Here are a couple of steps in the process



☐ Click "Save All", close the XSL editor and "mplan" editor

☐ Deploy the project

Submit A19 QRY and Review A19 ADR Response

We will use the CMDHL7Sender command line client to read a file containing a single HL7 A19 QRY message and submit it to the ADT Receiver endpoint. We will then look at the response returned to the client. The sample A19 Query message is shown below. Note

```
MSH|^~\&|CLI1|A19QRY|GWYQ|A19ADR|200607031745||QRY^A19|A_0000000|P|2.3.1
QRD|200607031745|R|I|Q1004|||1^RD|3716691^^^^^^^^D224|DEM
```

Query definition, QRD segment, annotated:

QRD	Segment name
200607031745	Query Date/Time
R	Query format code – R = Record-oriented
I	Query priority – I = Immediate
Q1004	Query ID
-	Deferred response type (not applicable)
-	Deferred response date/time

1^RD	Quantity limited request – how many items to return = 1 Record
3716691^^^^^^^^D224	Who subject filter – Patient ID 3716691 issued by Assigning Authority D224
DEM	What subject filter – DEM = Demographics

In short, we are asking for a single record containing demographic information for patient D224/3716691.

The response will include a PID segment (Patient Identification) with the demographic information, if the patient is found.

The high-level structures of the QRY and ADR messages, reproduced from the HL7 2.3.1 standard document, are below. Elements enclosed in "[]" are optional.

```
QRY                    Patient Query                              Chapter

MSH                    Message Header                               2
QRD                    Query Definition                             2
[ QRF ]                Query Filter                                 2


ADR                    ADT Response                               Chapter

MSH                    Message Header                               2
MSA                    Message Acknowledgment                       2
[ERR]                  Error                                        2
QRD                    Query Definition                             2
     [ QRF ]           Query Filter                                 2
{
     [ EVN ]           Event Type                                   3
      PID              Patient Identification                       3
     [PD1]             Additional Demographics                      3
     [ {NK1} ]         Next of Kin /Associated Parties              3
PV1                    Patient Visit                                3
     [ PV2 ]           Patient Visit - Additional Info.             3
     [ { DB1 } ]       Disability Information                       3
     [ {OBX} ]         Observation/Result                           7
     [ {AL1} ]         Allergy Information                          3
     [ {DG1} ]         Diagnosis Information                        6
     [DRG]             Diagnosis Related Group                      6
     [ {PR1       Procedures                                       6
         [{ROL}] Role12
     }]
     [ {GT1} ]         Guarantor                                    6
     [
       {
        IN1            Insurance                                    6
        [ IN2 ]        Insurance Additional Info.                   6
        [ {IN3} ] Insurance Add'l Info - Cert.                      6
       }
     ]
     [ ACC ]           Accident Information                         6
     [ UB1 ]           Universal Bill Information                   6
     [ UB2 ]           Universal Bill Information                   6

}
[ DSC ]                Continuation Pointer                         2
```

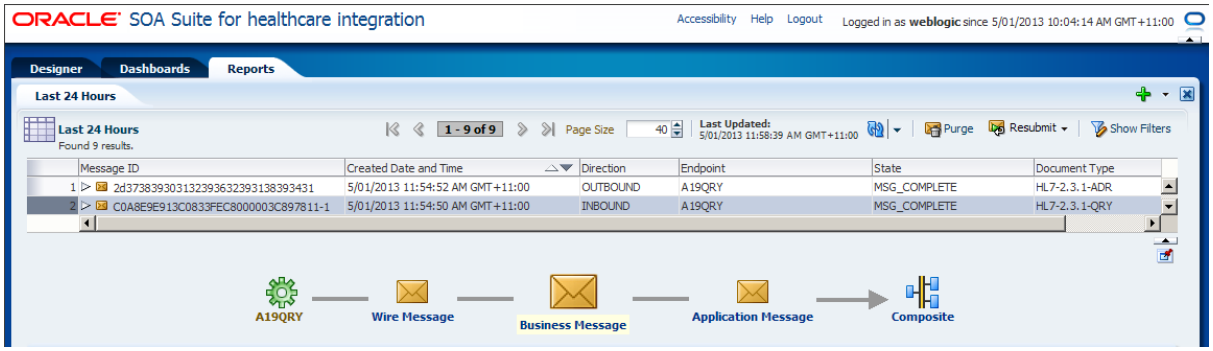
☐ In a command / terminal window execute the following command

```
java -jar c:\tools\CMDHL7\CMDHL7Sender_v0.7.jar -a SystemA -b HosA -c ID_
-n 1 -d \r\r\n -p 21100 -h localhost -t 30000 -f
c:\hl7\adt\sources\QRY_A19.hl7
```

☐ Inspect the response which the command line tool displays in the console window

```
MSH|^~\&|SystemA|HosA|GWYQ|A19ADR|200607031745||ADR^A19|ADR_ID__0000000|P|2.3.1
MSA|AA|ID__0000000
QRD|200607031745|R|I|Q1004|||1^RD|3716691^^^^^^^^D224|DEM
PID|||3716691^^^D224||Smith^John||19611011|M||||||||||0908070605
```

☐ In the "Healthcare Integration Console" switch to "Reports" Tab and inspect the messages



Summary

In this article we developed and exercised an A19 Query Processor solution.