



IBM Integration Bus

Embedded
Global Cache
with
WebSphere
eXtreme Scale

June, 2013

Hands-on lab built at product
code level Version 9.0.0.0

1. INTRODUCTION TO GLOBAL CACHE.....	3
1.1 SCENARIO	4
1.2 LAB PREPARATION.....	6
2. SET THE DEFAULT GLOBAL CACHE POLICY FOR IB9NODE	7
3. IMPORT THE APPLICATION.....	8
3.1 REVIEW THE WXS_REQUEST APPLICATION.....	9
3.2 REVIEW THE WXS_CUSTOMER APPLICATION	13
3.3 REVIEW THE WXS_RESPONSE APPLICATION.....	14
4. DEPLOY AND TEST IN AN INTEGRATION NODE	16
4.1 SHOW THE EMBEDDED CACHE RESOURCE STATISTICS IN INTEGRATION EXPLORER.....	19
4.2 SHOW THE GLOBAL CACHE STATISTICS FOR THE INTEGRATION NODE.....	21
5. DEPLOY AND TEST IN DIFFERENT INTEGRATION NODES.....	23
5.1 CONFIGURE TWO INTEGRATION NODES FOR "SHARED" GLOBAL CACHE.....	23
5.2 DEPLOY THE WXS_RESPONSE APPLICATION IN IB9NODE2	25
5.3 MODIFY THE UDP VALUE IN WXS_REQUEST	27
5.4 TEST AND REVIEW THE GLOBAL CACHE STATISTICS	31
<i>This concludes the Global Cache with WebSphere eXtreme Scale lab.</i>	<i>32</i>

1. Introduction to Global Cache

WebSphere Message Broker V8.0.0.1 (fix pack 1), provided a new feature called Global Cache. IBM WebSphere Message Broker V8.0.0.2 (fix pack 2) further extended this feature to provide an automatic data revocation feature. IBM Integration Bus V9.0.0.0 provides these features without the need to activate them using the `mqsichangebroker` command.

Global Cache allows a number of Integration Nodes to be integrated for workload balancing. Global Cache provides the ability to have a cache storing information about the requester which would later be used to correlate the replies correctly. Using Global cache gives the flexibility for different Integration Nodes to handle the request and reply parts of an application. Using a global cache allows you to scale up the number of clients, while maintaining predictable response time for each client.

Global Cache uses an embedded version of WebSphere eXtreme Scale, and provides the following functions:

- 'Elastic' In-Memory Data Grid – managing itself to scale out, scale in, failover, failure etc.
- Virtualizes free memory within a grid of Java Virtual machines (JVMs) into a single logical space which is accessible as a partitioned, key addressable space for use by applications
- Provides fault tolerance through replication with self-monitoring and healing
- The space can be scaled out by adding more JVMs while its running without restarting
- Provides a predictable scaling option
- Access to external WebSphere Xtreme Scale data grids

IBM Integration Bus Version 9.0.0.0 contains support for:

- 1) An embedded WebSphere eXtreme Scale (WXS) grid which can be used as a global cache from within message flows.

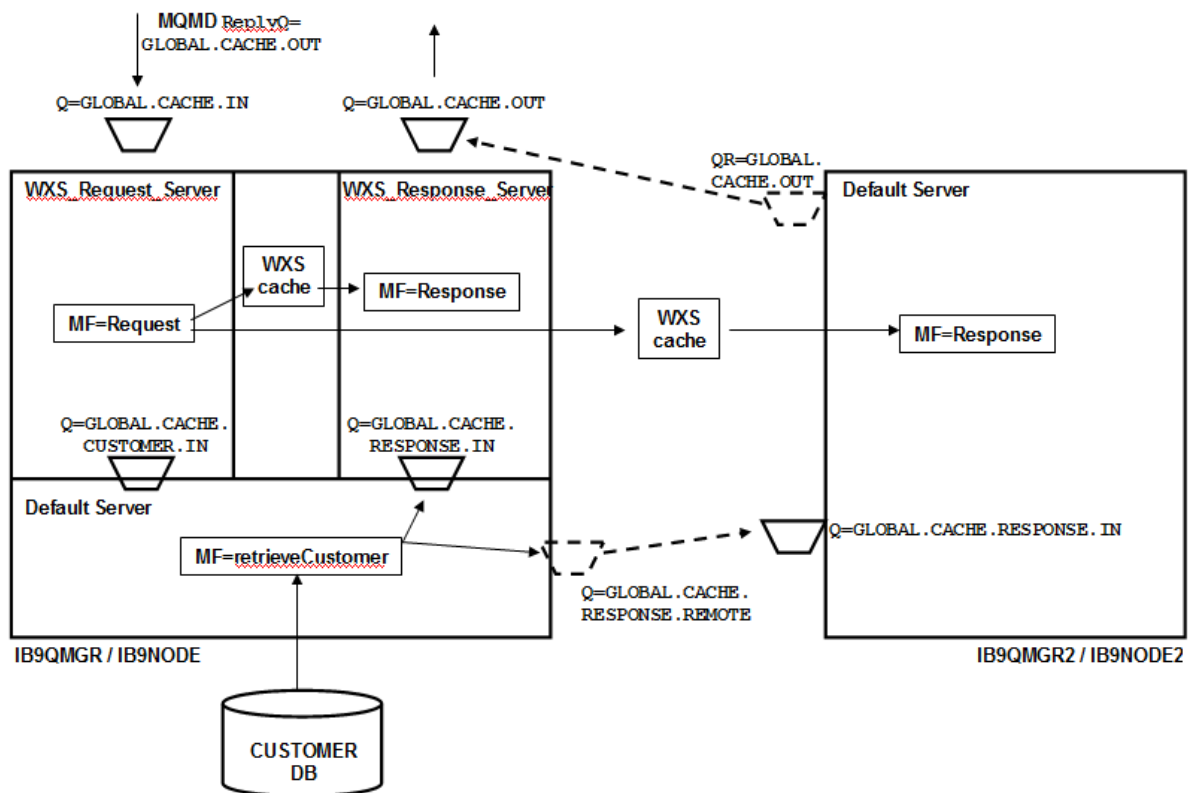
WXS components are hosted within Integration Server processes and operate with no requirement for additional configuration. The default scope of one cache is across one Integration node (i.e. multiple Integration Servers) but it can be extended to be across multiple Integration nodes.

- 2) Access to WebSphere Xtreme Scale Grids that are running outside of the Integration Node for example of a DataPower XC10 machine.

IBM Integration node support of external WebSphere Extreme Scale grids is covered under the Lab Guide “**Global Cache Using DataPower XC10 Appliance**” from the same series of Lab Guides.

1.1 Scenario

The following schematic shows the system context of the Global Cache Lab.



There are three message flows in this scenario:

1. **Request** – this flow receives messages on the `GLOBAL.CACHE.IN` queue. The application that puts messages on this queue also specifies the ultimate reply queue, which will be set to `GLOBAL.CACHE.OUT` in this example. Note that there are two instances of this reply queue, one on each of the queue managers. (This has been done for simplicity of demonstration, rather than creating a full MQ clustering scenario).

The Request message flow takes the MessageID and ReplyQ from the incoming message, and stores them in the embedded Global Cache. This is so that the Response message flow will be able to use this data to send the reply back to the specified reply queue, even though Response may be running in a different Integration Server, or Integration Node.

The Request flow has a user-defined property, which specifies the name of the MQ queue that the retrieveCustomer flow will send the resulting message to. This value is used to populate the ReplyQ that will be used by retrieveCustomer. This has been done for ease of illustration of the Global Caching component, when switching the demo between multiple Integration Servers and multiple Integration Nodes.

2. **retrieveCustomer** – this flow takes the input message (passed by Request) and uses the data to retrieve a customer record from the CUSTOMER database. This is done with a simple Compute node. It then sends the output message to the specified Reply queue. Depending on the user-defined property in Request, this can be a local queue or a remote queue (where the message will be sent to IB9QMGR2).

3. **Response** – this flow runs in both Integration Nodes, and reads the `GLOBAL.CACHE.RESPONSE.IN` queue. It uses the MessageID to retrieve the required final reply queue name from the global cache, and sends the message to the final output queue.

1.2 Lab preparation

In the pre-built vmware, all this section has been done for you.

To run this lab, unzip the supplied file `global_cache.zip` into the directory `c:\student` directory. This will create a subdirectory called `\global_cache` with several further subdirectories.

Create the MQ resources

In the `c:\student\global_cache\install` folder, run the command `createCacheQueues.bat`.

This batch script also creates a second queue manager `IB9QMGR2` and second Integration Node, `IB9NODE2`, as well as the resources used for the default IIB queue manager to communicate with the second queue manager.

Create the databases

In an Integration Command Console, go the folder `c:\student\global_cache\install`.

Issue the commands:

```
CreateRegularDB
CreateRegularTables

SetBrokerSecurityForCaching

mqsistop IB9NODE
mqsistop IB9NODE2

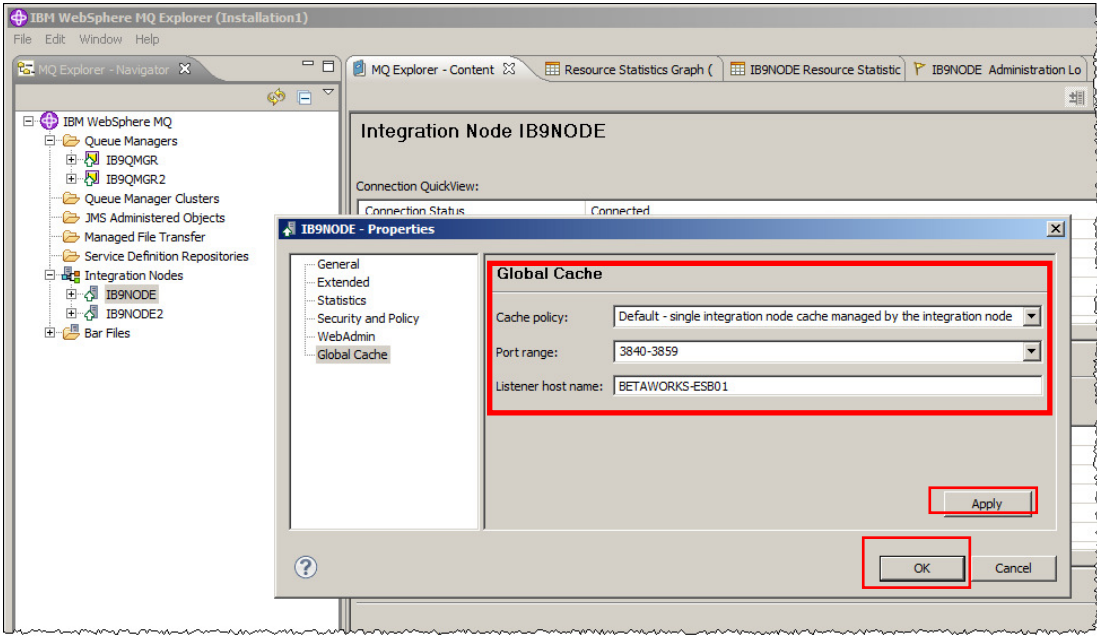
mqsistart IB9NODE
mqsistart IB9NODE2
```

2. Set the default global cache policy for IB9NODE

By default the embedded global cache component is disabled. To enable it, a “Cache Policy” needs to be set. The default cache policy creates a default topology of cache components in a single Integration node.

The default port range used by the default Integration Node for the embedded eXtreme Scale component is 2800-2819. In the event that this port range conflicts with any other applications, you can change this port range using the IB9QMGR.

In the case of the pre-built system which includes WAS and Information Server, this is necessary to avoid a port conflict.

1.	<p>In Integration Explorer, select IB9NODE.</p> <p>Right-click IB9NODE, and select Properties. Select Global Cache.</p>
2.	<p>Set Cache policy to “Default - single integration node cache managed by the integration node”.</p> <p>Change the port range to “3840-3859”.</p> <p>Set the Listener host name to “BETAWORKS-ESB01”.</p> <p>Click Apply, then OK.</p> 
3.	<p>Stop and restart the Integration Node.</p>

3. Import the application

The application that you will use to investigate the Global Cache is provided for you. In this section, you will import the application, and investigate certain aspects of the flow logic.

1. In the Integration Toolkit, import the Project Interchange file
c:\student\global_cache\resources\WXS_Start.zip

2. Explore the items have been defined in the Application Development navigator.

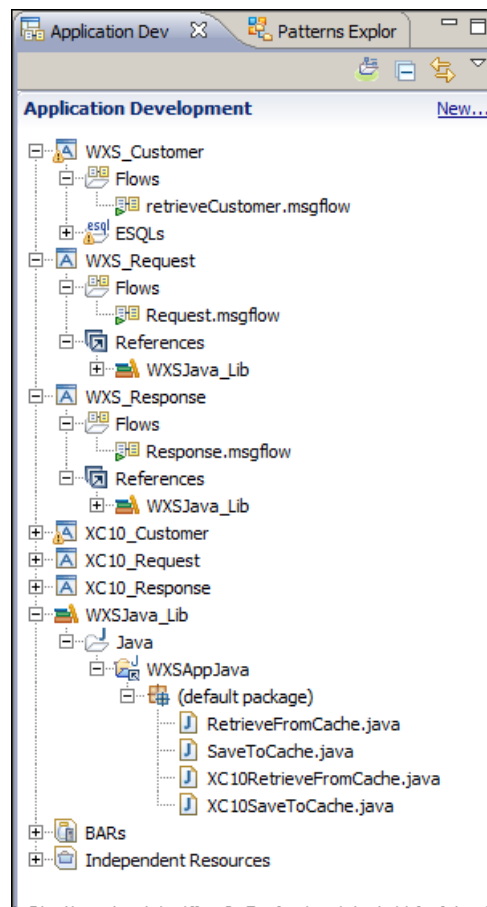
Things to note:

There are three applications, each containing one message flow. The applications have been organized in this way in order that each application (and therefore message flow) can be deployed independently to separate Integration Servers, or to separate Integration Nodes. This is important for this particular global cache scenario.

The WXS_Request and WXS_Response applications have a reference to the WXSJava_Lib library.

The WXSJava_Lib library contains two java compute nodes, which are responsible for writing information to the cache, and for reading from it.

The WXS_Customer application has no reference. The retrieveCustomer message flow is independent, and uses a simple Compute node to read a DB2 database.

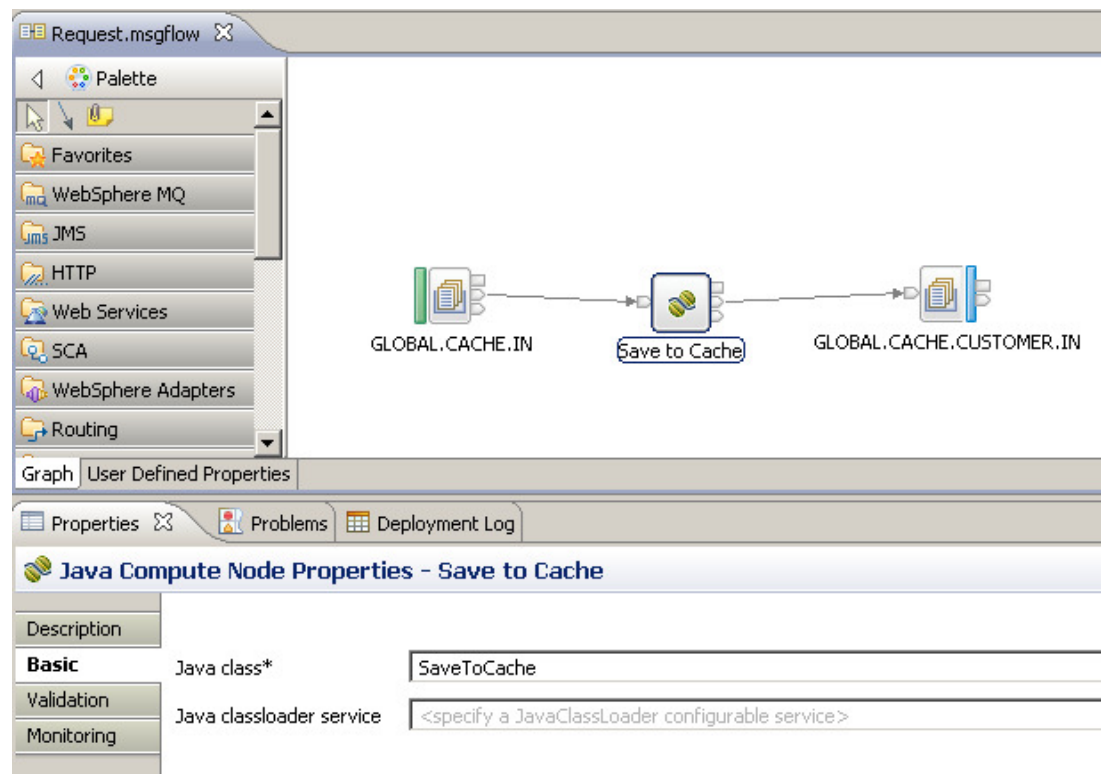


3.1 Review the WXS_Request application

3. Open the **Request** message flow:

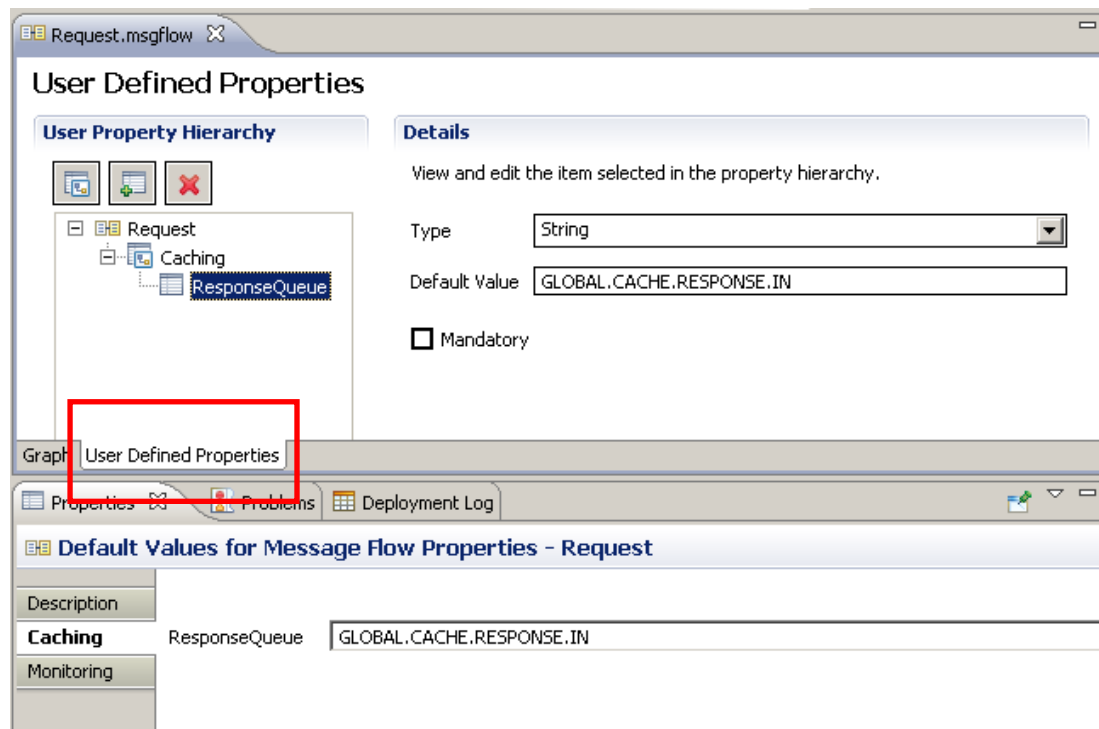
This message flow reads an MQ message from queue `GLOBAL.CACHE.IN`, and uses a java compute node to store the MQ MessageID and Reply Queue Name (required to correlate the final MQ response message) into the cache.

The message is then sent to the `retrieveCustomer` message flow by writing it to the `GLOBAL.CACHE.CUSTOMER.IN` queue.



4. In the flow editor, click “User defined properties”.

You will see that this flow has a user-defined property called `ResponseQueue`. This UDP is used to determine whether the output from the `retrieveCustomer` flow is sent to a local queue, or to a remote queue (for the purpose of demonstrating caching across multiple Integration Nodes). The default value is `GLOBAL.CACHE.RESPONSE.IN`, which is a local MQ queue.



In the multiple Integration node scenario, this value will be overridden by changing the value in a barfile, avoiding the need to make any changes to the java code. It will be changed to a remote queue definition, so the reply message will be sent to a different queue manager (IB9QMGR2), with the Response message flow running in a different Integration Node.

When finished, switch back to the Graph view.

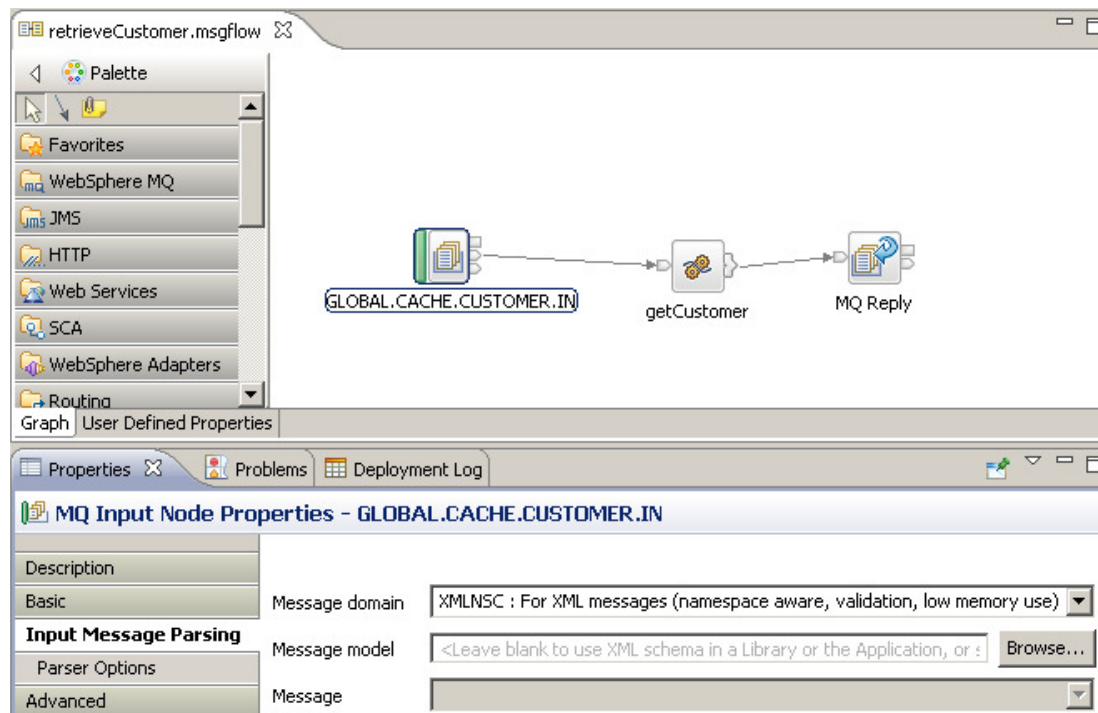
5.	Double-click on the SaveToCache Compute node to see how the Java logic writes information to the cache. There are three key sections of java code:
6.	<p>This section of code retrieves the MQ MsgId and ReplyQ from the MQMD:</p> <pre>// Get the original MsgId and ReplyToQ of the incoming request MbElement rootEl = outAssembly.getMessage().getRootElement(); MbElement replyToQE1 = rootEl.getFirstElementByPath("/MQMD/ReplyToQ"); String replyToQ = replyToQE1.getValueAsString(); MbElement msgIdEl= rootEl.getFirstElementByPath("/MQMD/MsgId"); String msgId = msgIdEl.getValueAsString();</pre>
7.	<p>The following sections of code save the message and replyToQ information in the (embedded) global cache to a range of different maps.</p> <p>Note:</p> <ol style="list-style-type: none"> When writing data to the cache a <key><value> pair is written, in this example message ID is used for the key – a “duplicate Key” condition is raised if you attempt to write a key that already exists. MbGlobalMapSessionPolicy can be used before writing data to a map to define how long entries will exist in the map before they are automatically deleted:
8.	<pre>/* * We're about to overwrite the original ReplyToQ of this message. * So, write it to the global cache it for safe keeping. This will * be used subsequently by the Response message flow to send the * response back to the original reply queue. */ /* * Map 1 - "aliveUntilRestart" * write data to a map that will preserve the data until the IB node * restarts ie no explicit data eviction policy - default is to keep * the data until restart of IB node. */ MbGlobalMap globalMap=MbGlobalMap.getGlobalMap("aliveUntilRestart"); globalMap.put(msgId, replyToQ);</pre>
9.	<pre>/* * Map 2 "aliveFor60Seconds" * Write data to a map that will automatically expire after * 60 seconds. The retrieveFromCache code will use this map to * restore the replyToQ */ MbGlobalMap LiveFor60=MbGlobalMap.getGlobalMap("aliveFor60Seconds", new MbGlobalMapSessionPolicy(60)); LiveFor60.put(msgId, replyToQ);</pre>

10.	<pre>/* * Map 3 "aliveFor120Seconds" * Write data to a map that will automatically expire after 120 * seconds */ MbGlobalMap LiveFor120= MbGlobalMap.getGlobalMap("aliveFor120Seconds", new MbGlobalMapSessionPolicy(120)); LiveFor120.put(msgId, replyToQ);</pre>
11.	<p>Finally, this section of code reads the value of a user-defined property "ResponseQueue". The value of this user-defined property determines whether the reply queue of the retrieveCustomer flow will be a local or remote queue.</p> <pre>responseQueue = getUserDefinedAttribute("ResponseQueue").toString(); outMessage.getRootElement().getFirstElementByPath("/MQMD/ReplyToQ"). setValue(responseQueue + instance);</pre>
12.	<p>Close the java editor when finished (without saving any changes), and close the Request message flow.</p>

3.2 Review the WXS_Customer application

13. Now open the retrieveCustomer message flow.

On the MQ Input node, the Message Parsing domain has been set to XMLNSC. This is because the flow needs access to the input data (CustomerID) to retrieve a record from the CUSTOMER table.



14. Open the getCustomer Compute node. The ESQL code will retrieve a row from the CUSTOMER table, putting it temporarily in Environment.Variables.

The FirstName and LastName fields are then stored in the output message.

```
BEGIN
-- CALL CopyMessageHeaders();
SET OutputRoot = InputRoot;
SET OutputRoot.XMLNSC.CUSTOMER = NULL;
-- populate the environment with passenger info from the database
SET Environment.Variables =
    THE (SELECT T.* FROM Database.CUSTOMER AS T
        WHERE T.CUSTOMERID = InputRoot.XMLNSC.CUSTOMER.CUSTOMERID);

-- populate the output message with info from the database query
CREATE FIELD OutputRoot.XMLNSC.CUSTOMER;
DECLARE outpass REFERENCE TO OutputRoot.XMLNSC.CUSTOMER;
SET outpass.FirstName = Environment.Variables.FIRSTNAME;
SET outpass.LastName = Environment.Variables.LASTNAME;
RETURN TRUE;
END;
```

Close the ESQL editor.

3.3 Review the WXS_Response application

15. Now open the Response message flow.

First, note that the MQ input node reads the queue `GLOBAL.CACHE.RESPONSE.IN`. This queue exists in both `IB9QMGR` and `IB9QMGR2`, so the flow can execute in either of the integration nodes.

The screenshot displays the IBM Integration Bus Studio interface. The top pane shows a message flow diagram for 'Response.msgflow'. It consists of three nodes connected in sequence: an input node labeled 'GLOBAL.CACHE.RESPONSE.IN', a central node labeled 'Retrieve from Cache', and an output node labeled 'MQ Reply'. The 'Retrieve from Cache' node is highlighted with a blue border. Below the diagram, the 'Properties' tab is active, showing the 'Java Compute Node Properties - Retrieve from Cache'. The 'Basic' section is expanded, showing the 'Java class*' field set to 'RetrieveFromCache' and the 'Java classloader service' field set to '<specify a JavaClassLoader configurable service>'. Other tabs like 'Description', 'Validation', and 'Monitoring' are visible but not expanded.

16. Open the **RetrieveFromCache** java node.

The following are the important parts of the java code.

This section retrieves the MessageID from the MQMD.

```
String msgId = inAssembly
    .getMessage()
    .getRootElement()
    .getFirstElementByPath("/MQMD/CorrelId")
    .getValueAsString();
```

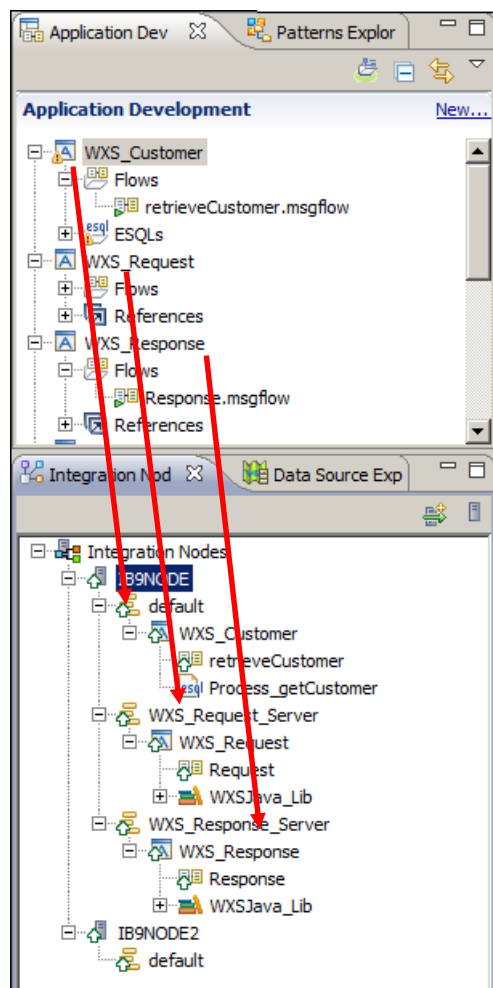
17.	<p>This section uses the MessageID to retrieve the name of the reply queue from the Global Cache. The key value pair retrieved from the map will automatically be deleted 60 seconds after the last access attempt.</p> <pre>/* * Now we can restore the original ReplyToQ by looking it up * in the cache. Get data from aliveFor60Seconds Map : */ MbGlobalMapglobalMap=MbGlobalMap.getGlobalMap("aliveFor60Seconds"); String replyToQ = (String)globalMap.get(msgId);</pre>
18.	<p>This section sets the ReplyQ field in the output message, and resets the value of the MessageID to the original value.</p> <pre>// Set the ReplyToQ field outMessage.getRootElement().getFirstElementByPath("/MQMD/ReplyToQ"). setValue(replyToQ); // Set the MsgId back to what it was before outMessage.getRootElement().getFirstElementByPath("/MQMD/MsgId").set Value(getBytes(msgId));</pre> <p>Close the RetrieveFromCache java code.</p>

4. Deploy and test in an Integration Node

This part of the lab will test the caching applications, with all applications deployed to a single Integration Node. To demonstrate the global cache, each of the three applications will be deployed into separate Integration Servers, as follows:

1. Default Integration Server – WXS_Customer
2. WXS_Request_Server – WXS_Request
3. WXS_Response_Server – WXS_Response

1. First, create two new Integration Servers, WXS_Request_Server and WXS_Response_Server. (Right-click the IB9NODE, and select “New Integration Server”).
2. Deploy the applications as shown above. Drag and drop the application into the required Integration Server.
3. When all three applications are deployed, the IB9NODE should look something like this:

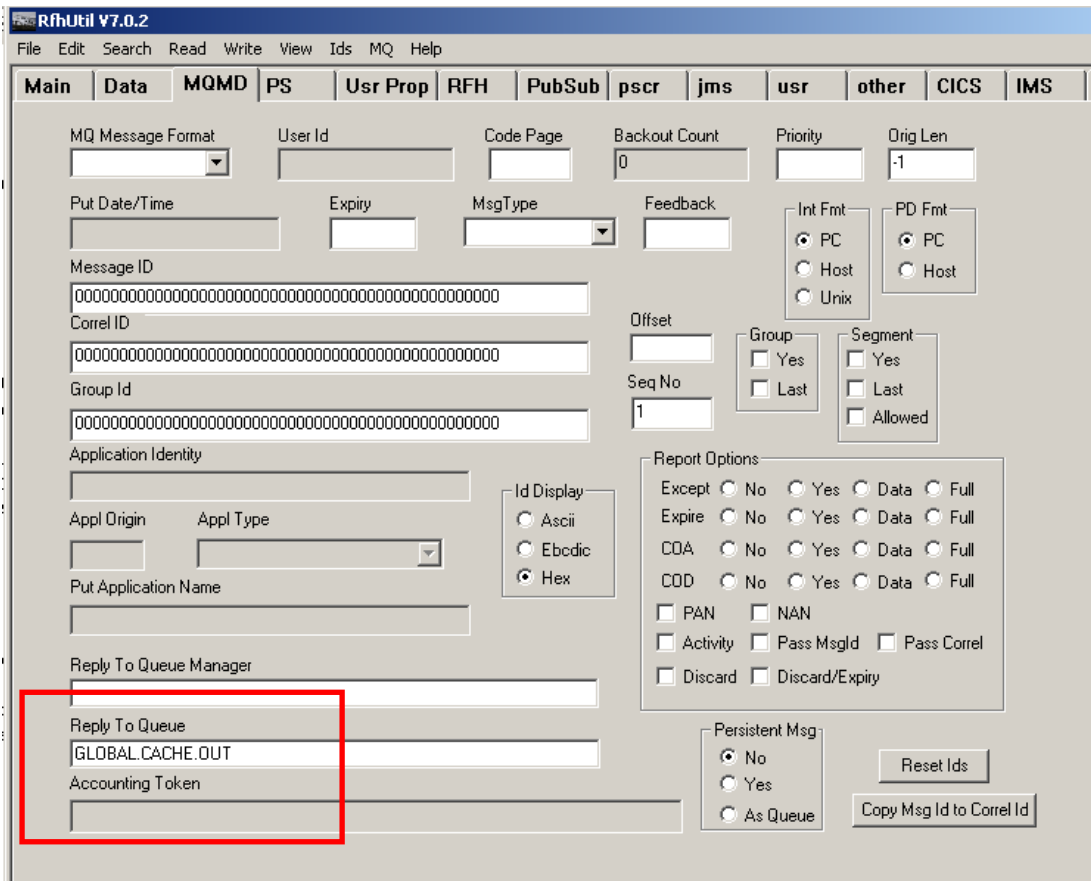


4. Open RFHUtil, and open the file named c:\student \ global_cache \ data \ Customer0001.xml.

This data contains a CustomerID with a value of 0001. The CustomerID will be used to retrieve customer data from the SUBREG1 database, by the WXS_Customer application.

To ensure the request/reply scenario works, you must specify the name of the reply queue, which is the name of the queue where the final output will appear (ie. the Response application will write to this queue).

5. On the MQMD tab, set the “Reply to Queue” to GLOBAL . CACHE . OUT.



On the Main tab, write the message to the input queue GLOBAL . CACHE . IN on IB9QMGR

6. In Integration Explorer, you should see that the message count for GLOBAL.CACHE.OUT has been incremented by 1.

This demonstrates that the Request and Response applications both have access to the same shared data (the name of the reply queue), even though they are running in separate Integration Servers. This is provided by the embedded global cache.

The screenshot shows the 'Queues' view in MQ Explorer. A table lists various queues with their types and message counts. The row for 'GLOBAL.CACHE.OUT' is highlighted with a red rectangle, showing an open output count of 1 and a current queue depth of 1.

Queue name	Queue type	Open input count	Open output count	Current queue depth	Put messages	Get messages
DEST.FAILURE	Local	0	0	0	Allowed	Allowed
DEST.IN	Local	0	0	0	Allowed	Allowed
DESTLIST.FAILURE	Local	0	0	0	Allowed	Allowed
DESTLIST.IN	Local	0	0	0	Allowed	Allowed
FAILURE	Local	0	0	0	Allowed	Allowed
GLOBAL.CACHE.CUSTOMER.IN	Local	1	1	0	Allowed	Allowed
GLOBAL.CACHE.IN	Local	1	1	0	Allowed	Allowed
GLOBAL.CACHE.OUT	Local	0	1	1	Allowed	Allowed
GLOBAL.CACHE.RESPONSE.IN	Local	1	1	0	Allowed	Allowed
GLOBAL.CACHE.RESPONSE.REMOTE	Remote				Allowed	

7. In a second RFHUtil instance, read the queue GLOBAL.CACHE.OUT.

This should show a record from the CUSTOMER table.

The screenshot shows the 'GLOBAL.CACHE.OUT' window in RFHUtil. The 'Data' tab is selected, displaying XML message data. The data format is set to XML, and the message content is as follows:

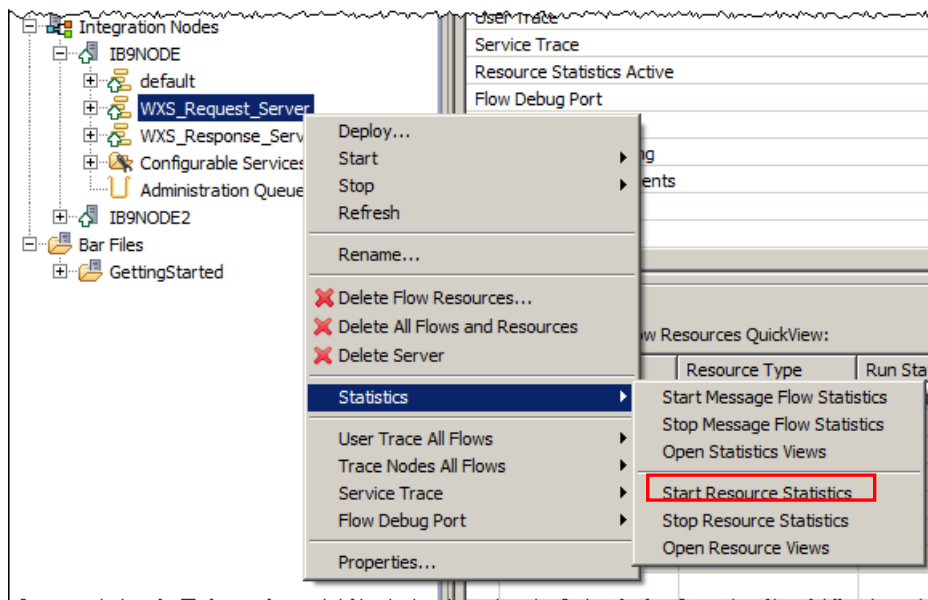
```

<CUSTOMER>
  <FirstName>Paul</FirstName>
  <LastName>Tergeist (Regular) </LastName>
</CUSTOMER>
    
```

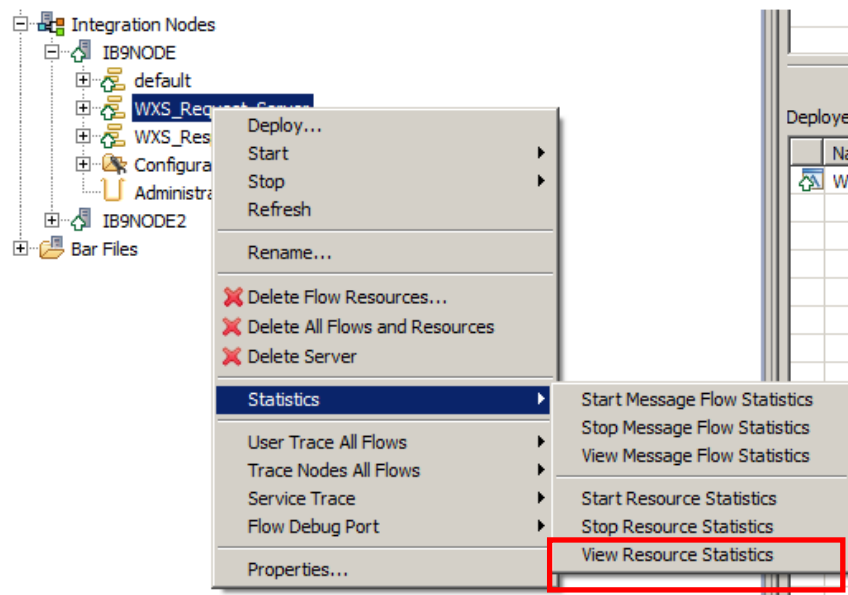
4.1 Show the embedded Cache Resource Statistics in Integration Explorer

1. Now check the Resource Statistics for the three Integration Servers.

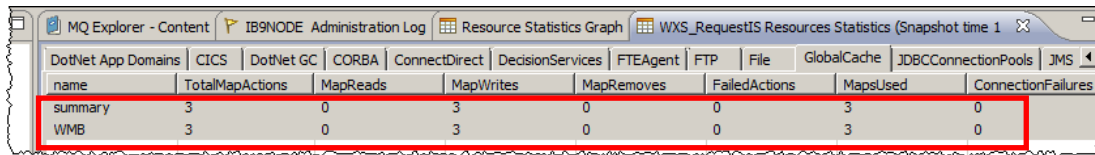
In Integration Explorer, right-click the WXS_Request_Server, and select Statistics, Start Resources Statistics.



2. Open the Resource Views, by right-click the WXS_Request_Server again, and select Statistics, Open Resource Views



3. Both the graphical and tabular views will open. On the tabular view, click the GlobalCache tab. You will see that the MapWrites count will be incremented by 3 since the WXS_Request application has written to 3 maps: "aliveFor60Seconds"; "alivefor120Seconds"; "aliveUntilRestart".



name	TotalMapActions	MapReads	MapWrites	MapRemoves	FailedActions	MapsUsed	ConnectionFailures
summary	3	0	3	0	0	3	0
WMB	3	0	3	0	0	3	0

4. Performing the same tasks on the WXS_Response_Server will show the GlobalCache "MapReads" count being incremented by one. The WXS_Response application reads from the map "aliveFor60Seconds". Leaving the data in the other two maps untouched.

4.2 Show the Global Cache statistics for the Integration Node

In addition to the statistics provided by the Integration Node for each Integration server, the Integration node also provides statistics about the overall performance of the Extreme Scale cache at the Integration Node level.

This is provided with the `mqsicacheadmin` command, which has several parameters.

1.	<p><i>If more than 120 seconds has past since you wrote a message to "GLOBAL.CACHE.IN" using RFHUTIL write another message and make sure the queue depth for "GLOBAL.CACHE.OUT" is updated by (indicating the cache has been used).</i></p>
2.	<p>To display the map entries in the Integration Node, open an Integration Console.</p> <p>Issue the command</p> <pre>mqsicacheadmin IB9NODE -c showMapSizes</pre> <p>From the listing of the MapsSizes you will see that the three maps <code>aliveFor60Seconds</code>, <code>aliveFor120seconds</code>, <code>aliveUntilRestart</code> all contain (at least) one entry (more if you have written additional messages to "GLOBAL.CACHE.IN"</p> <pre>C:\IBM\MQSI\9.0.0.0>mqsicacheadmin IB9NODE -c showMapSizes BIP7187I: Output from the mqsicacheadmin command. The output from the WebSphere eXtreme Scale xscmd utility is ' Starting at: 2013-04-25 15:20:08.088 CWXSI0068I: Executing command: showMapSizes *** Displaying results for WMB data grid and mapSet map set. *** Listing maps for IB9NODE_BETAWORKS-ESB01_3840 *** Map Name Partition Map Entries Used Bytes Shard Type Container ----- aliveFor120Seconds 8 1 488 B Primary IB9NODE_BETAWORKS- ESB01_3840_C-0 aliveFor60Seconds 8 1 488 B Primary IB9NODE_BETAWORKS- ESB01_3840_C-0 aliveUntilRestart 8 1 488 B Primary IB9NODE_BETAWORKS- ESB01_3840_C-0 Server total: 3 (1 KB) *** Listing maps for IB9NODE_BETAWORKS-ESB01_3847 *** Map Name Partition Map Entries Used Bytes Shard Type Container ----- aliveFor120Seconds 8 1 488 B SynchronousReplica IB9NODE_B ETAWORKS-ESB01_3847_C-1 aliveFor60Seconds 8 1 488 B SynchronousReplica IB9NODE_B ETAWORKS-ESB01_3847_C-1 aliveUntilRestart 8 1 488 B SynchronousReplica IB9NODE_B ETAWORKS-ESB01_3847_C-1 Server total: 3 (1 KB) Total catalog service domain count: 6 (2 KB) (The used bytes statistics are accurate only when you are using simple objects o r the COPY_TO_BYTES copy mode.) CWXSI0040I: The showMapSizes command completed successfully. Ending at: 2013-04-25 15:20:10.322 ' BIP8071I: Successful command completion.</pre>

3. Wait 60 seconds and repeat the command again. This time you will see that data in `aliveFor60Seconds` has been automatically deleted from the map (the data eviction policy on the map was specified to delete data after 60 seconds from the last update):
- ```
C:\IBM\MQSI\9.0.0.0>mqsicacheadmin IB9NODE -c showMapSizes
BIP7187I: Output from the mqsicacheadmin command. The output from the WebSphere
eXtreme Scale xscmd utility is '
Starting at: 2013-04-25 15:21:21.650

CWXSI0068I: Executing command: showMapSizes

*** Displaying results for WMB data grid and mapSet map set.

*** Listing maps for IB9NODE_BETAWORKS-ESB01_3840 ***
Map Name Partition Map Entries Used Bytes Shard Type Container

aliveFor120Seconds 8 1 488 B Primary IB9NODE_BETAWORKS-
ESB01_3840_C-0
aliveFor60Seconds 8 0 0 Primary IB9NODE_BETAWORKS-
ESB01_3840_C-0
aliveUntilRestart 8 1 488 B Primary IB9NODE_BETAWORKS-
ESB01_3840_C-0
Server total: 2 (976 B)

*** Listing maps for IB9NODE_BETAWORKS-ESB01_3847 ***
Map Name Partition Map Entries Used Bytes Shard Type Container

aliveFor120Seconds 8 1 488 B SynchronousReplica IB9NODE_
BETAWORKS-ESB01_3847_C-1
aliveFor60Seconds 8 0 0 SynchronousReplica IB9NODE_
BETAWORKS-ESB01_3847_C-1
aliveUntilRestart 8 1 488 B SynchronousReplica IB9NODE_
BETAWORKS-ESB01_3847_C-1
Server total: 2 (976 B)
Results were not returned for map name (not provided) and partition (not provide
d). Verify that the provided map name and partition are correct or try the comma
nd again with fewer filters.

Total catalog service domain count: 4 (1 KB)
(The used bytes statistics are accurate only when you are using simple objects o
r the COPY_TO_BYTES copy mode.)

CWXSI0040I: The showMapSizes command completed successfully.

Ending at: 2013-04-25 15:21:23.572
'
BIP8071I: Successful command completion.
```
4. Wait a further 60 seconds so that the data in the `aliveFor120Seconds` map also expires. Repeat the `mqsicacheadmin` command again to see that the map `aliveUntilRestart` is the only map left which contains map entries. Data in this map will not be automatically deleted whilst the IB9NODE cache catalog servers are running. When we wrote data to this map we did not specify `MbMapSessionPolicy`
- Data in a map with no data eviction policy can be cleared from the cache:
- when the IB9NODE is stopped or restarted
  - using `"mqsicacheadmin <IBNODE> -c clearGrid -m <MAPNAME>"`

## 5. Deploy and test in different Integration Nodes

This section will deploy the same applications, but this time into separate Integration Nodes. The WXS\_Response application will be deployed into IB9NODE2.

There are three primary tasks to configure the global cache for inter-Integration Node use:

- Create and edit a global cache configuration file
- Configure the Integration Node IB9NODE to use the cache configuration file
- Configure the second Integration Node, IB9NODE2, to use the cache configuration file

### 5.1 Configure two Integration Nodes for “shared” global cache

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1. | <p>A sample version of the cache configuration file is provided in the IBM Integration Bus installation directory, under <code>\MQSI\9.0.0.0\sample\globalcache</code>. Several samples are provided. This scenario is based on the “two Integration Node” sample.</p> <p>The cache configuration file has been provided for you. Open the file <code>c:\student\global_cache\resources\IB9NODE_IB9NODE2_CacheConfig.xml</code>.</p> <p>Note the follow key lines. The listenerHost and port ranges much match the values specified for the Global Cache config using the Integration Explorer.</p> <pre>&lt;?xml version="1.0" encoding="UTF-8"?&gt; &lt;cachePolicy xmlns="http://www.ibm.com/xmlns/prod/websphere/messagebroker/globalcache/policy-1.0"&gt;    &lt;!--The Integration node "IB9NODE" runs on "BETAWORKS-ESB01".--&gt;   &lt;broker name="IB9NODE" listenerHost="BETAWORKS-ESB01"&gt;      &lt;!-- This broker hosts one catalog server. --&gt;     &lt;catalogs&gt;1&lt;/catalogs&gt;      &lt;!-- This broker uses ports between 3840-3859.--&gt;     &lt;portRange&gt;       &lt;startPort&gt;3840&lt;/startPort&gt;       &lt;endPort&gt;3859&lt;/endPort&gt;     &lt;/portRange&gt;   &lt;/broker&gt;    &lt;!-- The Integration node "IB9NODE2" runs on "BETAWORKS-ESB01". --&gt;   &lt;broker name="IB9NODE2" listenerHost="BETAWORKS-ESB01"&gt;      &lt;!-- This broker hosts no catalog servers. --&gt;     &lt;catalogs&gt;0&lt;/catalogs&gt;      &lt;!-- This broker uses ports between 4820-4839 --&gt;     &lt;portRange&gt;       &lt;startPort&gt;4820&lt;/startPort&gt;       &lt;endPort&gt;4839&lt;/endPort&gt;     &lt;/portRange&gt;   &lt;/broker&gt; &lt;/cachePolicy&gt;</pre> |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

2. In the Integration Explorer, (under Integration Nodes) right-click IB9NODE, and select Properties, Global Cache.

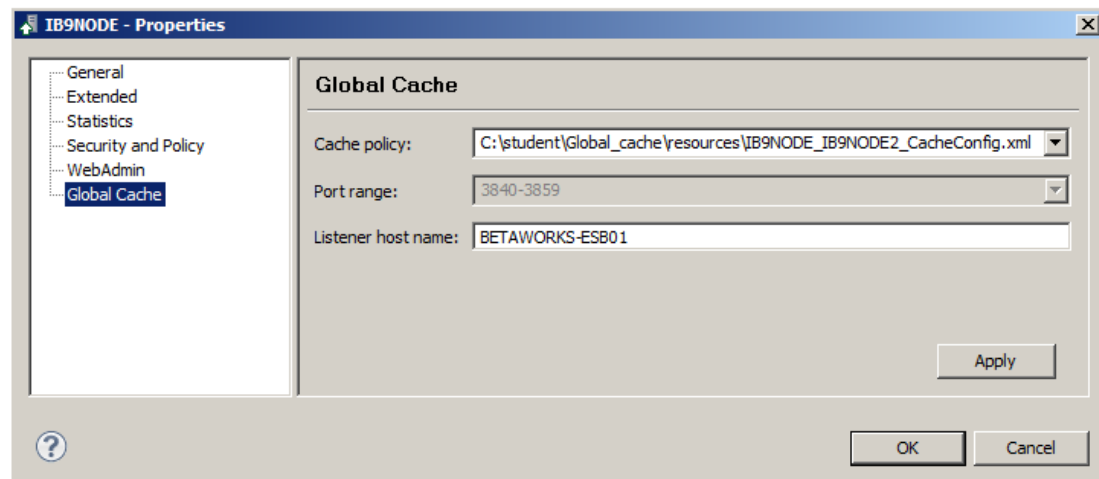
Overtyping the value in cache Policy with the fully qualified name of the global cache configuration file name:

C:\student\Global\_cache\resources\IB9NODE\_IB9NODE2\_CacheConfig.xml

Leave the port range unchanged (the config file will override this value).

Leave the Listener host name as "BETAWORKS-ESB01".

Click **Apply**, then OK.



3. In the Integration Explorer, right-click IB9NODE2, and select Properties, Global Cache.

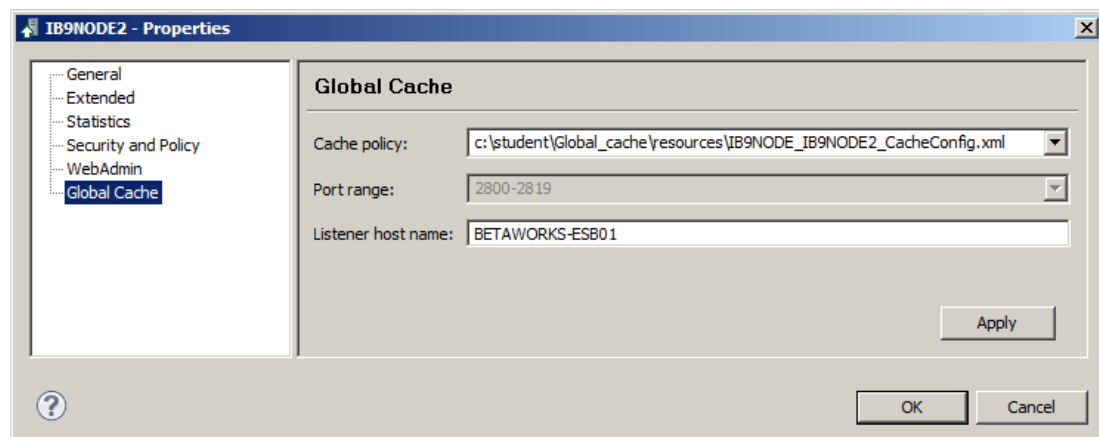
Overtyping the value in cache Policy with the global cache configuration file name,

c:\student\Global\_cache\resources\IB9NODE\_IB9NODE2\_CacheConfig.xml

Leave the port range unchanged (the config file will override this value).

If not already set, set the Listener host name to "BETAWORKS-ESB01".

Click Apply, then OK.



4. Stop and restart both Integration Nodes.

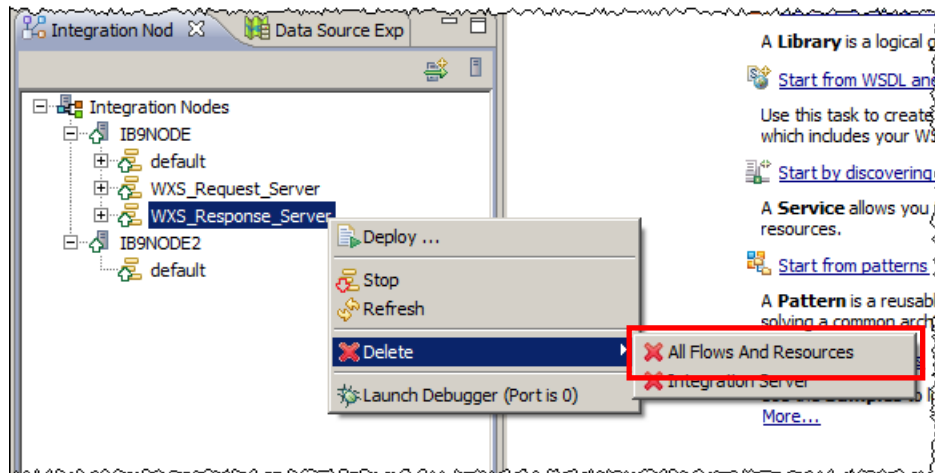
Examine the windows Event Log to ensure that both nodes have started.



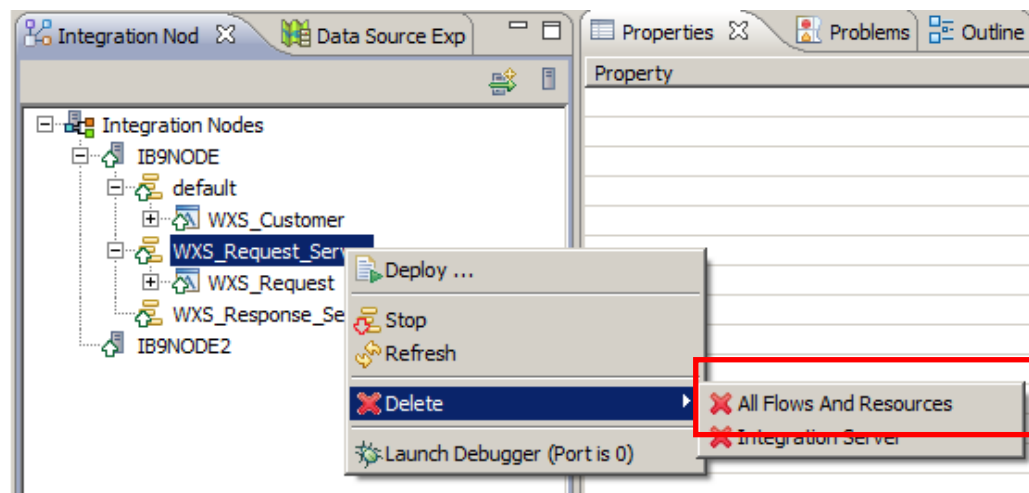
## 5.2 Deploy the WXS\_Response application in IB9NODE2

5. To ensure that the application does not have duplicate copies, in Integration Toolkit delete the WXS\_Response application from IB9NODE.

Right-click WXS\_Response\_Server and select Delete, All Flows And Resources:

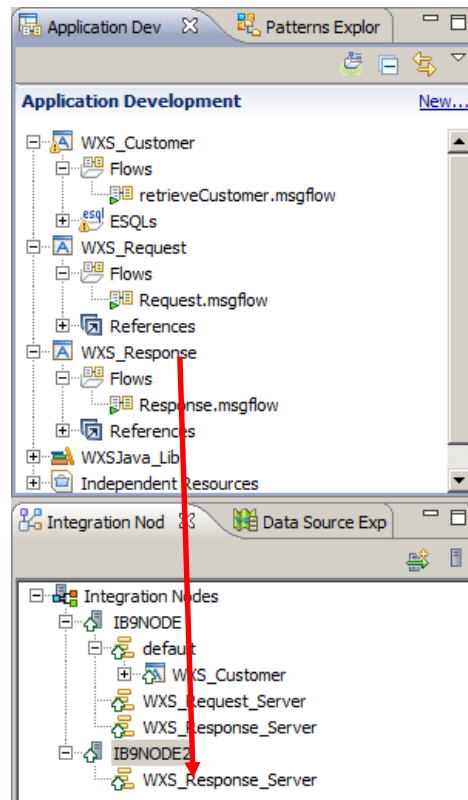


6. In Integration Toolkit, delete the WXS\_Request application from WXS\_Request\_Server (right click on WXS\_Request\_Server and select "delete", "All flows and resources"):



(Select OK when asked are you sure?)

7. Create a new Integration Server called `WXS_Response_Server` in `IB9NODE2`.  
Deploy the `WXS_Response` application to this Integration Server.



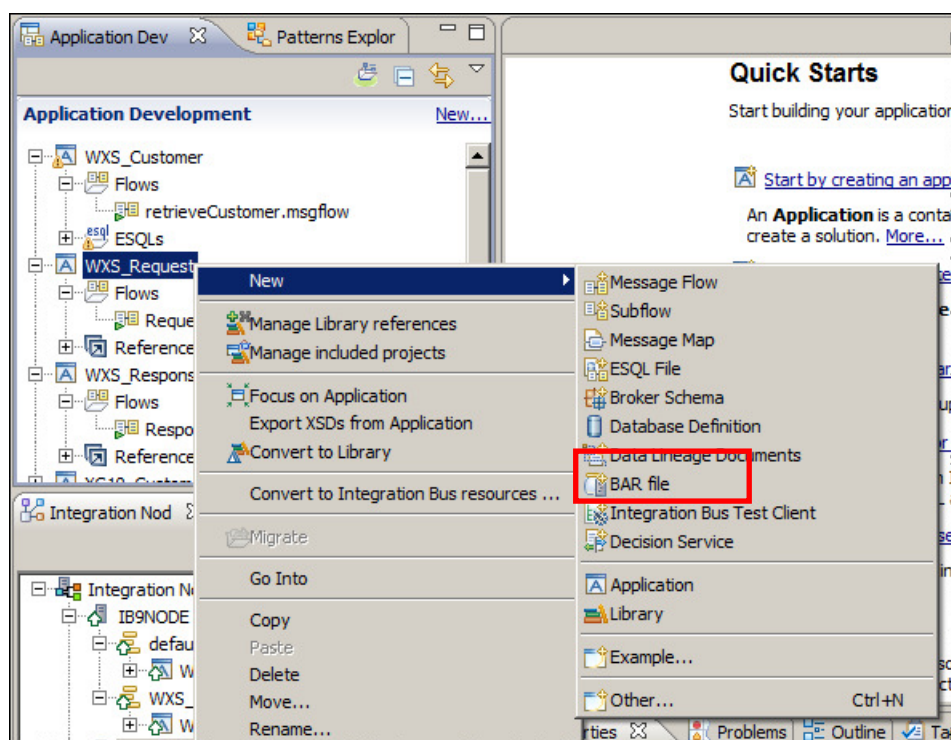
### 5.3 Modify the UDP Value in WXS\_Request

We want to make sure that the WXS\_Customer application sends its customer data back to the second integration node IB9NODE2.

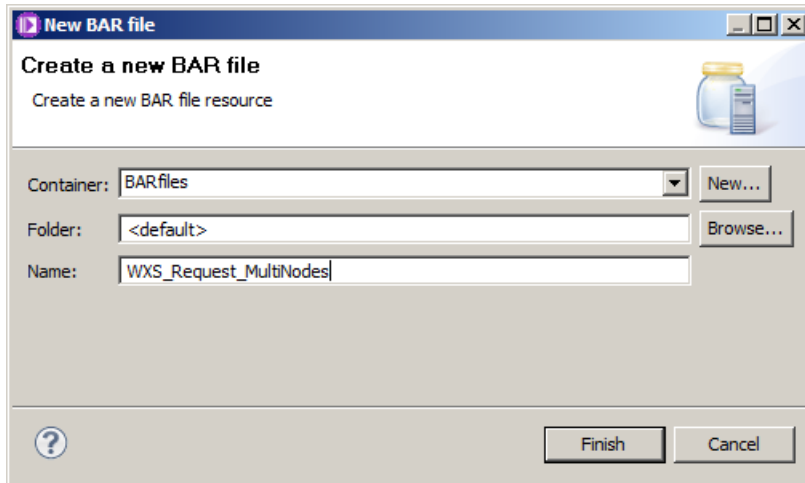
We will do this by changing the WXS\_Customer application to send its output to a remote MQ queue definition on IB9QMGR (GLOBAL.CACHE.RESPONSE.REMOTE) which points to a local queue (GLOBAL.CACHE.RESPONSE.IN) defined on IB9QMGR2. The WXS\_Customer application gets this information from a user-defined property "ResponseQueue" which is set in the WXS\_Request application.

We will now change this value by editing the property in the bar file.

8. In the Application Development window (In the Integration Toolkit), right click on WXS\_Request and choose "New", "Bar file" :

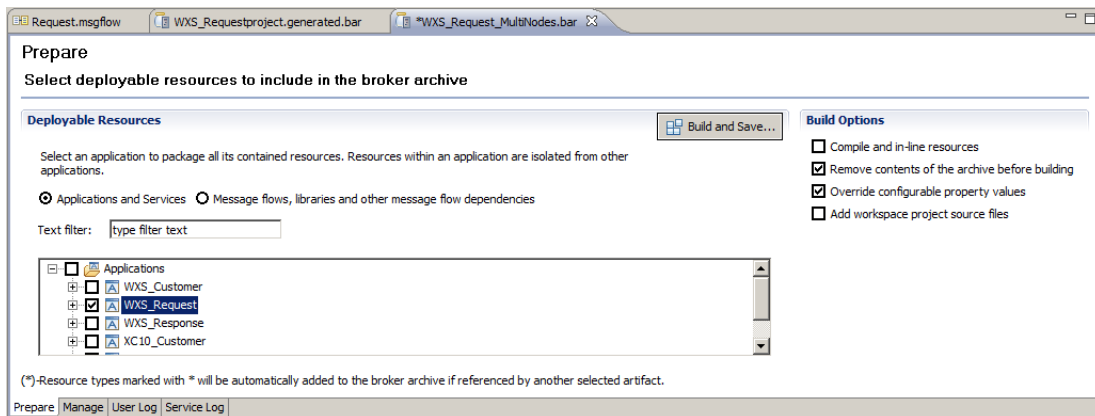


9. In the Create new bar file wizard, call the new bar file "WXS\_Request\_MultiNodes":

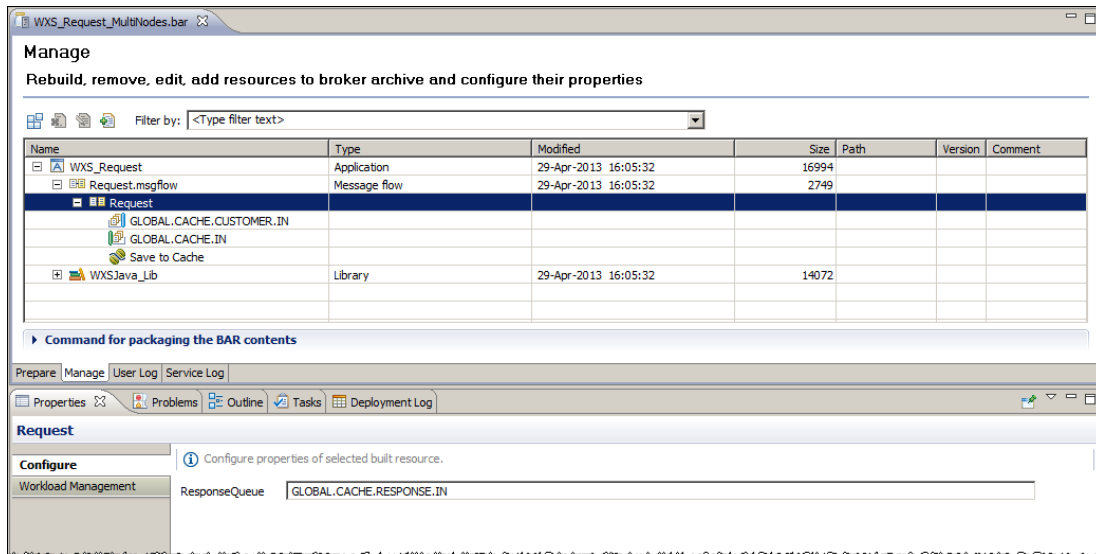


(Click Finish to create the bar file).

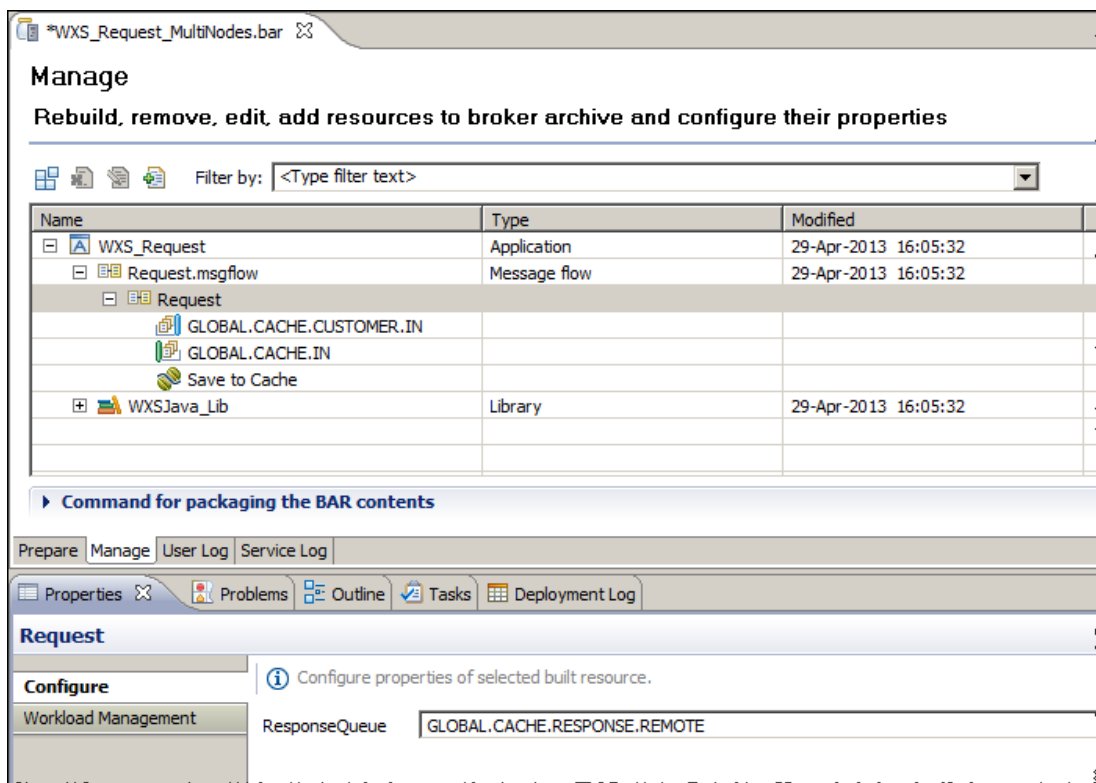
10. When the "Prepare" tab appears, select the tick box WXS\_Request, Click Build and Save button (click OK when the bar file has been successfully created):



- On the Manage tab, expand the WXS\_Request application, and select the Request flow, the properties tab will show the User Defined Property “ResponseQueue” (used to define the response queue that the WXS\_Response application will use as input:

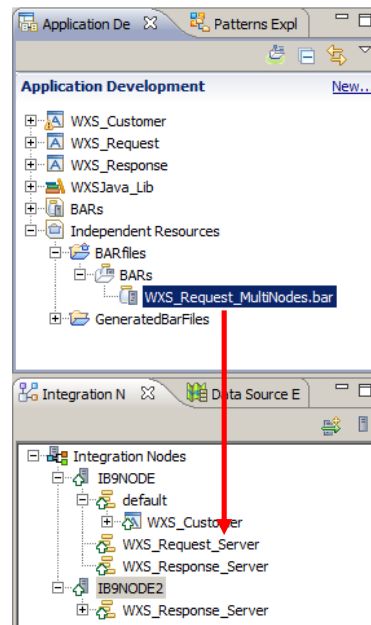


- Change the value for ResponseQueue to “GLOBAL.CACHE.RESPONSE.REMOTE”

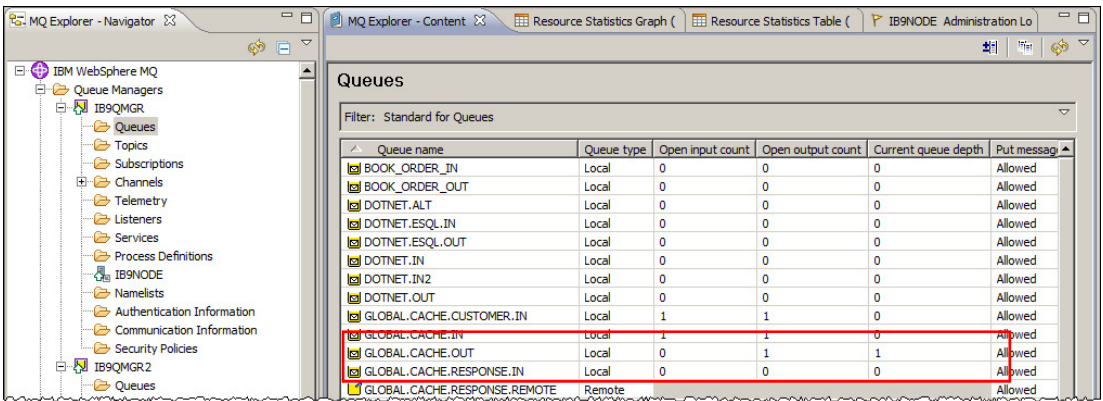
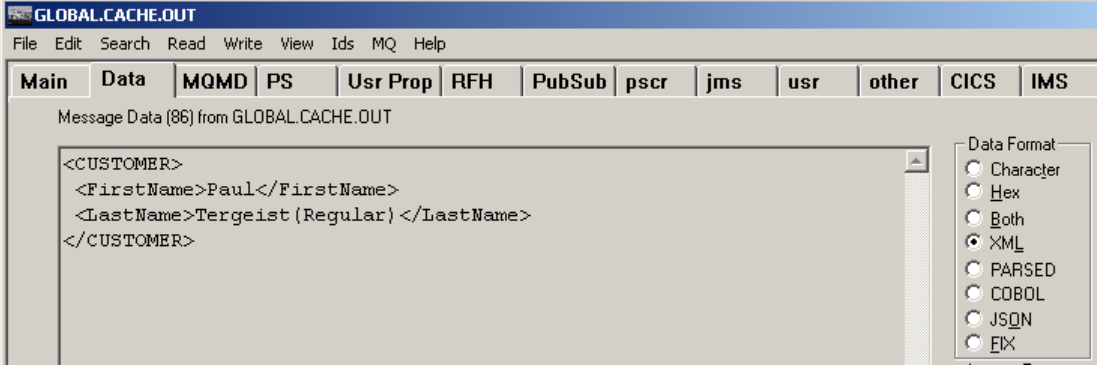


Save the Bar file (<ctrl> s)

13. Deploy the saved bar file to the (empty) WXS\_Request\_Server in IB9NODE (ie. the same place as before, although this time you are deploying the updated barfile, rather than the application):



## 5.4 Test and Review the Global Cache Statistics

14. The two queue managers, IB9QMGR and IB9QMGR2 have already been defined with the necessary MQ channels and transmission queues. The channels are set to automatically start when a message appears on the transmission queue.
15. Now retest the applications with RFHUtil.
- Using the same input data, and the same input queue, write the message to GLOBAL.CACHE.IN. (Make sure the MQMD Reply Queue is still set to GLOBAL.CACHE.OUT).
16. In MQ Explorer, again select the queues defined for the IB9QMGR queue manager.
- Note that the queue depth for the GLOBAL.CACHE.OUT on IB9QMGR has been incremented by 1.
- 
- | Queue name                   | Queue type | Open input count | Open output count | Current queue depth | Put message |
|------------------------------|------------|------------------|-------------------|---------------------|-------------|
| BOOK_ORDER_IN                | Local      | 0                | 0                 | 0                   | Allowed     |
| BOOK_ORDER_OUT               | Local      | 0                | 0                 | 0                   | Allowed     |
| DOTNET.ALT                   | Local      | 0                | 0                 | 0                   | Allowed     |
| DOTNET.ESQL.IN               | Local      | 0                | 0                 | 0                   | Allowed     |
| DOTNET.ESQL.OUT              | Local      | 0                | 0                 | 0                   | Allowed     |
| DOTNET.IN                    | Local      | 0                | 0                 | 0                   | Allowed     |
| DOTNET.IN2                   | Local      | 0                | 0                 | 0                   | Allowed     |
| DOTNET.OUT                   | Local      | 0                | 0                 | 0                   | Allowed     |
| GLOBAL.CACHE.CUSTOMER.IN     | Local      | 1                | 1                 | 0                   | Allowed     |
| GLOBAL.CACHE.IN              | Local      | 1                | 1                 | 0                   | Allowed     |
| GLOBAL.CACHE.OUT             | Local      | 0                | 1                 | 1                   | Allowed     |
| GLOBAL.CACHE.RESPONSE.IN     | Local      | 0                | 0                 | 0                   | Allowed     |
| GLOBAL.CACHE.RESPONSE.REMOTE | Remote     |                  |                   |                     | Allowed     |
17. Use the second RFHUtil to read the queue GLOBAL.CACHE.OUT.
- 
- ```

<CUSTOMER>
  <FirstName>Paul</FirstName>
  <LastName>Tergeist (Regular) </LastName>
</CUSTOMER>

```
18. In an Integration Console, run the following command:
- ```
mqsicacheadmin IB9NODE -c showMapSizes
```
- to see the map entries written to the three maps in the WXS\_Request application:

```

19. C:\IBM\MQSI\9.0.0.0>mqsicacheadmin IB9NODE -c showMapSizes
BIP7187I: Output from the mqsicacheadmin command. The output from the WebSphere
eXtreme Scale xscmd utility is '
Starting at: 2013-04-30 11:15:02.853

CWXSI0068I: Executing command: showMapSizes

*** Displaying results for WMB data grid and mapSet map set.

*** Listing maps for IB9NODE_BETAWORKS-ESB01_3840 ***
Map Name Partition Map Entries Used Bytes Shard Type Container

aliveFor120Seconds 8 1 488 B Primary IB9NODE_BETAWORKS-
ESB01_3840_C-0
aliveFor60Seconds 8 1 488 B Primary IB9NODE_BETAWORKS-
ESB01_3840_C-0
aliveUntilRestart 8 1 488 B Primary IB9NODE_BETAWORKS-
ESB01_3840_C-0
Server total: 3 (1 KB)

*** Listing maps for IB9NODE_BETAWORKS-ESB01_3844 ***
Map Name Partition Map Entries Used Bytes Shard Type Container

aliveFor120Seconds 8 1 488 B SynchronousReplica IB9NODE_B
ETAWORKS-ESB01_3844_C-1
aliveFor60Seconds 8 1 488 B SynchronousReplica IB9NODE_B
ETAWORKS-ESB01_3844_C-1
aliveUntilRestart 8 1 488 B SynchronousReplica IB9NODE_B
ETAWORKS-ESB01_3844_C-1
Server total: 3 (1 KB)
Results were not returned for map name (not provided) and partition (not provide
d). Verify that the provided map name and partition are correct or try the comma
nd again with fewer filters.

Total catalog service domain count: 6 (2 KB)
(The used bytes statistics are accurate only when you are using simple objects o
r the COPY_TO_BYTES copy mode.)

CWXSI0040I: The showMapSizes command completed successfully.

Ending at: 2013-04-30 11:15:07.744
'
BIP8071I: Successful command completion.

C:\IBM\MQSI\9.0.0.0>

The above shows the output from the mqsicacheadmin command within 60 seconds of the
application running as all three maps have map entries.

If you wait longer to run the command maps "AliveFor60Seconds" and
"aliveFor120Seconds" will have had their data removed automatically.

```

This concludes the Global Cache with WebSphere eXtreme Scale lab.