IBM

βetaWorks

IBM Integration Bus

Publishing to Kafka and IBM MessageHub using the Kafka nodes

Featuring:

Configuring local Kafka servers on Windows
Using the IIB KafkaProducer node
Using the IBM Bluemix MessageHub service

January 2017
Hands-on lab built at product
Version 10.0.0.7

The KafkaProducer node
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks

Version 10.0.0.7

1. Introduction and Preparation

1.1 Introduction

In this lab you will use the Kafka Producer node. A complementary lab, 16L18_10007, will use the Kafka Consumer nodes to receive data produced by the REST API developed in this lab.

The Kafka Producer and Consumer nodes were introduced in IIB v10.0.0.7.

1.2 Scenario

The scenario described in this lab is based on a data replication use case. A primary data source (HRDB, relational database) will be updated by an application that is the primary owner of the database. Any updates that are made to HRDB will also be propagated to the Kafka server, where they will be used to drive the corresponding data updates into any number of target data stores. For example, this might include one or more NoSQL databases, or a CRM system such as Salesforce.com.
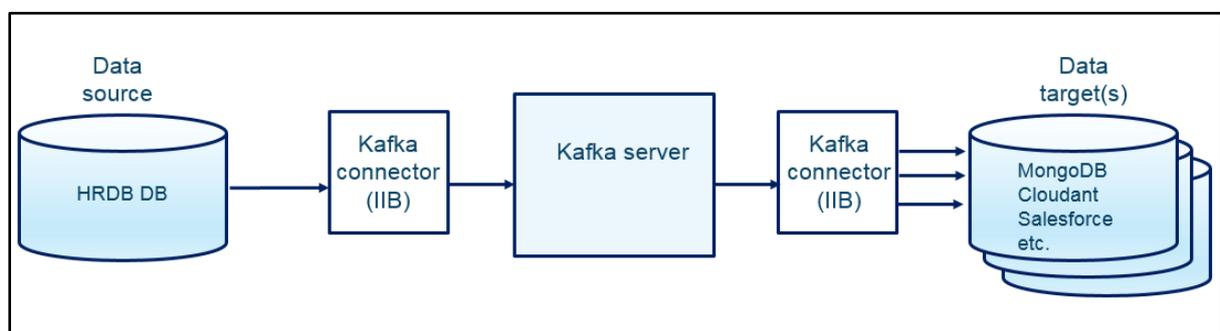
This lab will extend a partially-built solution of the HR_Service labs developed earlier. In this lab (built with IIB 10.0.0.7), the HR_Service scenario has been redesigned to accommodate additional data elements. These new elements are contained in a new database table called EMPLOYEE_SUPPLEMENTARY, and are described in the Model Definitions section below.

1.3 Kafka servers

This lab will investigate two Kafka servers:

- Chapters 2-4 will investigate the use of locally-installed Kafka servers. Development and testing of this lab was done with three Kafka servers installed locally on the IIB workshop Windows system. These three servers have configured Kafka replication (replication factor three can be used for topics)

- Chapter 5 will reproduce the same scenario, but using the IBM Bluemix MessageHub service instead of the local Kafka servers. MessageHub is IBMs implementation of Kafka on Bluemix.

Page 3 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks

1.3.1 Model Definitions
The following message models are used by this updated version of the HR_Service REST API.

- DBRESP – contains database response information

- EMPLOYEE – defines columns in the EMPLOYEE table

- EMPLOYEE_SUPPLEMENTARY – defines columns in the EMPLOYEE_SUPPLEMENTARY table

- DEPARTMENT – defines columns in the DEPARTMENT table

- EmployeeResponse
    - DBResp (type = DBRESP)
    - Employee   (Array, type = EMPLOYEE)

- DepartmentResponse
    - DBResp (type = DBRESP)
    - Department   (Array, type = DEPARTMENT)

- CompleteResponse
    - DBResp_employee (type = DBRESP)
    - Employee   (type = EMPLOYEE, single object, not array)
    - DBResp_department (type = DBRESP)
    - Department   (type = DEPARTMENT, single object, not array)
    - DBResp_employee_supplementary (type = DBRESP)
    - Employee_supplementary (type = EMPLOYEE_SUPPLEMENTARY)

- EmployeeSupplementaryResponse used when only accessing EMPLOYEE_SUPPLEMENTARY)
    - DBResp
    - Employee_supplementary

- EmployeeAddUpdateCompleteRequest (input message, used when adding a complete new employee)
    - Employee
    - EmployeeSupplementary

Page 4 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks

1.3.2 The HR_Service REST API

The partially-built solution version of the HR_Service REST API has implemented six operations:

- getEmployee
- getDepartment
- getSupplementary (retrieves data from the EMPLOYEE_SUPPLEMENTARY table)
- getComplete (invokes the getEmployee, getDepartment and getSupplementary operations)
- createEmployeeFromMultipart (adds new employee using a MIME request)
- createEmployeeFromEmbeddedImage (adds a new employee with a plain JSON request)

The four "get" operations are not used in this lab scenario. The two "create" operations invoke a common IIB subflow that performs the database inserts into the EMPLOYEE and EMPLOYEE_SUPPLEMENTARY tables. In this lab, you will extend this subflow to publish a notification message to the Kafka server when a new employee is added to the EMPLOYEE tables. A subsequent IIB application (described in the lab 16L18_10007) will consume this message, retrieve the new employee data from HRDB, and use this to propagate the update to another database, such as a NoSQL database (eg. MongoDB, or Cloudant).

Page 5 of 37   The KafkaProducer node   Version 10.0.0.7
                               Kafka servers and IBM MessageHub
                                   Provided by IBM BetaWorks
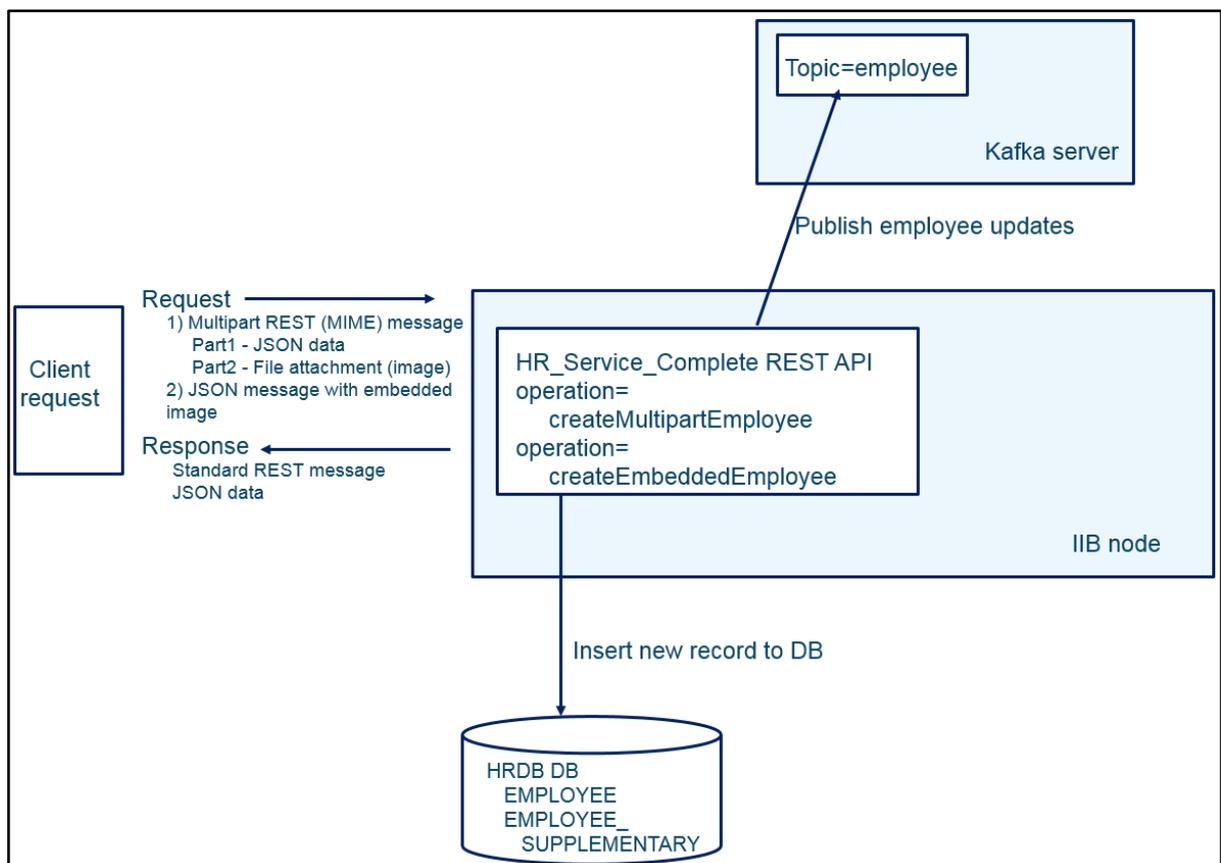
1.4 Configure TESTNODE_iibuser for REST APIs

The instructions in this lab guide are based on a Windows implementation, with a user named "iibuser".
The IIB workshop Windows VMWare image on which this lab is based is not available outside IBM, so you will need to provide your own software product installations where necessary.

If using the workshop VMWare system, login to Windows as the user "iibuser", password = "passw0rd". (You may already be logged in).

Start the IIB Toolkit from the Start menu.

The IIB support for the REST API requires some special configuration for the IIB node and server. Cross-Origin Resource Scripting (CORS) must be enabled for the IIB node to execute REST applications. This is also required when testing with the SwaggerUI test tool. See http://www.w3.org/TR/cors/?cm_mc_uid=09173639950214518562833&cm_mc_sid_50200000=1452177651 for further information.

1. Ensure that TESTNODE_iibuser is started.

2. Check that CORS has been enabled on the IIB node by running the following command in an Integration Console:

```
mqsireportproperties TESTNODE_iibuser
      -e default
      -o HTTPConnector
      -r
```

3. If CORS is enabled, you will see the following lines (amongst others):

```
corsEnabled='true'
  corsAllowOrigins='*'
  corsAllowCredentials='false'
  corsExposeHeaders='Content-Type'
  corsMaxAge='-1'
  corsAllowMethods='GET,HEAD,POST,PUT,PATCH,DELETE,OPTIONS'
  corsAllowHeaders='Accept,Accept-Language,Content-Language,Content-
    Type'
```

4. If CORS has not been enabled, run the following commands:

```
mqsichangeproperties TESTNODE_iibuser
      -e default
      -o HTTPConnector
      -n corsEnabled -v true

mqsistop TESTNODE_iibuser

mqsistart TESTNODE_iibuser
```

Page 6 of 37   The KafkaProducer node   Version 10.0.0.7
                                 Kafka servers and IBM MessageHub
                                   Provided by IBM BetaWorks

1.5 Configure Integration Bus node to work with DB2

A new database table has been introduced for labs written for IIB 10.0.0.7.

To run this lab, you should recreate any instances of the HRDB database. Follow the instructions below. You may need to alter the database schemas and authorities in the DDL to reflect your own environment.

1.5.1 Create database and tables

The HRDB database contains three tables: EMPLOYEE, EMPLOYEE_SUPPLEMENTARY and DEPARTMENT. These tables have already been created on the supplied workshop VMWare image. If you wish to create your own instance of this database (or to recreate on the workshop VM), do the following tasks:

1. Login with a user that has authority to create a new database and tables (on the workshop VM, use the user **iibadmin** (password=passw0rd).

2. Open an IIB Command Console (from the Start menu), and navigate to

    c:\student10\Create_HR_database

3. Run the commands

    1_Create_HRDB_database

    2_Create_HRDB_Tables

4. Logout user iibadmin (or your own user).

Appropriate database permissions are included in the scripts to GRANT access to the user iibuser. You may need to adjust these to match your own user definitions.

1.5.2 Create JDBC and security configurable services

To run this lab, the Integration Bus node must be enabled to allow a JDBC connection to the HRDB database.

1. Login with your standard IIB developer login (**iibuser** on the workshop VM).

2. Open an IIB Integration Console (from the Start menu), and navigate to

    c:\student10\Create_HR_database

3. Run the command

    3_Create_JDBC_for_HRDB

    Accept the defaults presented in the script. This will create the required JDBC configurable service for the HRDB database.

4. Run the command

    4_Create_HRDB_SecurityID

5. Stop and restart the node to enable the above definitions to be activated. As an example, on the workshop VM:

    mqsistop TESTNODE_iibuser

    mqsistart TESTNODE_iibuser

Page 7 of 37   The KafkaProducer node   Version 10.0.0.7
                                    Kafka servers and IBM MessageHub
                                         Provided by IBM BetaWorks

1.6 Open the Windows Log Monitor for IIB

A useful tool for IIB development on Windows is the IIB Log Viewer. This tool continuously monitors the Windows Event Log, and all messages from the log are displayed immediately.

From the Start menu, click IIB Event Log Monitor. The Monitor will open; it is useful to have this always open in the background.

This tool is not shipped as part of the IIB product; please contact us directly if you would like a copy.

Page 8 of 37   The KafkaProducer node   Version 10.0.0.7
                              Kafka servers and IBM MessageHub
                                 Provided by IBM BetaWorks

2. Explore and start the Kafka servers

The supplied VM system that is provided for the V10 workshop labs is supplied with a local installation of the Apache Kafka system. For the VM version containing IIB 10.0.0.7, the version of Kafka is 0.10.1.0

If you are running the lab scenarios on your own system, you will need to provision your own installation of Kafka. Alternatively, the sections of the lab document that reference the IBM Bluemix MessageHub Kafka implementation will work as described.

2.1 Kafka configuration for IIB workshop

On the workshop VM, Kafka is installed in **c:\tools\kafka_2.11-0.10.1.0**. In the **\bin\windows** folder, there are a number of ".bat" files that control various aspects of the Kafka system. For ease of use, some of these have been copied into the folder **c:\student10\kafka\commands**.

On the supplied workshop VM, Kafka has been configured to use a single Zookeeper server and three Kafka servers. This enables a topic Replication Factor of three. If you are doing this lab on your own system, follow the instructions below to reproduce the same configuration. You will need three Kafka servers, and a topic which has a replication factor of three.

The Kafka servers are shown schematically below. Note that all the servers are defined locally, so all have a unique listener port.
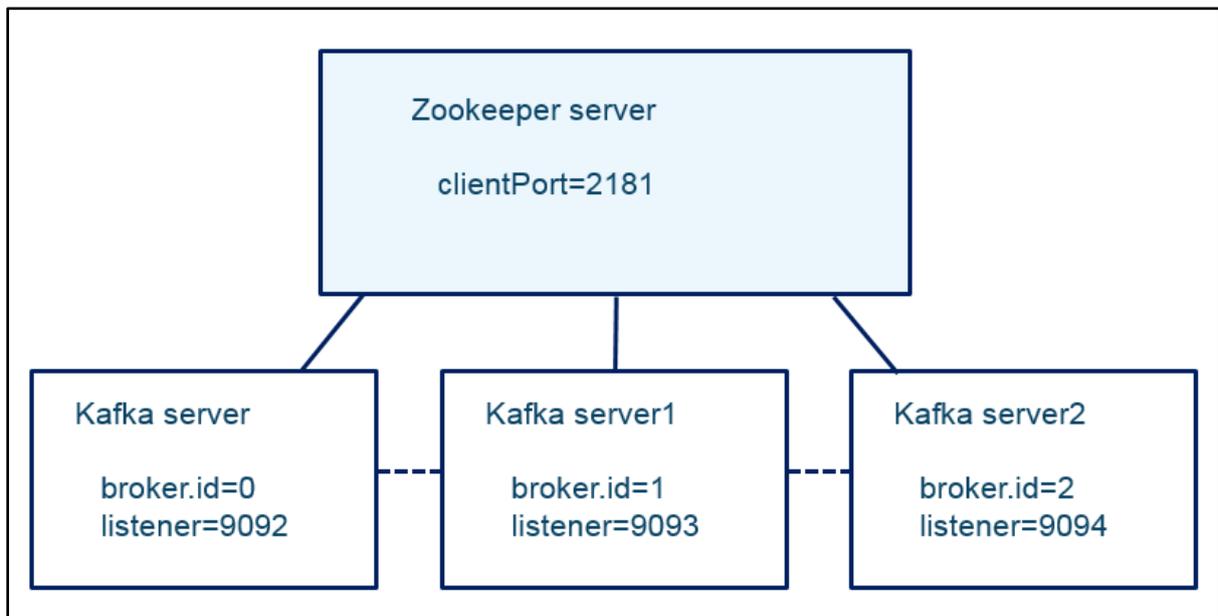
2.1.1 Kafka installation notes

The command files in c:\student10\kafka\commands have been tailored specifically for this lab and the workshop VMWare image. These make the various configuration items easier to achieve in this limited test environment.
If you are running this on your own system, make appropriate changes to these command files. Note that each command file sets the java CLASSPATH – make changes as appropriate for your system.

The Kafka logs have been placed in c:\kafka. This folder is referenced in the various "properties" files, described below.

Set the KAFKA_HOME variable to the install folder of kafka. Append the KAFKA_HOME variable to the PATH system environment variable.

Page 9 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks

2.2 Explore the Kafka Configuration

1. In Windows Explorer, navigate to the folder **c:\student10\kafka\config**.

2. Open the file **zookeeper.properties**.

   The data directory folder has been changed for use with the workshop VM system. If you are using your own installation, make appropriate changes here.

   > #dataDir=/tmp/zookeeper
   > dataDir=c://kafka/zookeeper
   > # the port at which the clients will connect
   > clientPort=2181
   > # disable the per-ip limit on the number of connections since this is a non-production config
   > maxClientCnxns=0

   Close the file when complete.

3. Open the file server.properties.

   Most properties have been left at the default values. The following properties have been set as follows:
   - Delete.topic.enable=true   (allows topics to be removed at server restart)
   - Broker.id=0
   - Listeners=PLAINTEXT://:9092
   - Log.dirs=c:/kafka/kafka-logs

   If appropriate, make similar changes for your own installation, and save the file.

   > # Topic deletion properties
   > delete.topic.enable=true
   >
   > ################## Server Basics ####################
   > # The id of the broker. This must be set to a unique integer for each broker.
   > broker.id=0
   >
   > ############### Socket Server Settings ##############
   > listeners=PLAINTEXT://:9092
   >
   > ################# Log Basics ###############
   > # A comma separated list of directories under which to store log files
   > log.dirs=c:/kafka/kafka-logs

4. Make similar changes to server-1.properties and server-2.properties as necessary, as follows:

   server-1.properties
   - Delete.topic.enable=true
   - Broker.id=1
   - Listeners=PLAINTEXT://:9093
   - Log.dirs=c:/kafka/kafka-logs-1

   server-2.properties
   - Delete.topic.enable=true
   - Broker.id=2
   - Listeners=PLAINTEXT://:9094
   - Log.dirs=c:/kafka/kafka-logs-2

Page 10 of 37   The KafkaProducer node   Version 10.0.0.7
                              Kafka servers and IBM MessageHub
   Provided by IBM BetaWorks

2.3 Start the Kafka servers

On the workshop VM system, Windows shortcuts have been provided for the Kafka commands that are required to start the various servers. These commands can be run as "iibuser".

1. From the Windows Start menu (or from the desktop), open the folder "Kafka commands)
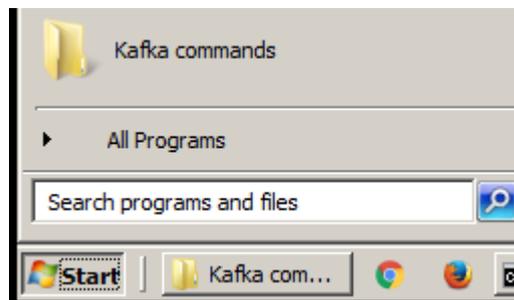
The following shortcuts will be available:
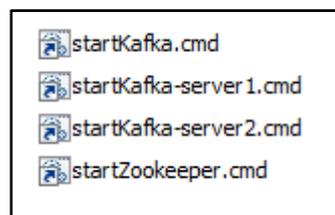
2. Open (run) **startZookeeper.cmd**. A Windows DOS command window will open and the zookeeper server will be started. A significant amount of log output will be produced.

   When started this way, the "startZookeeper.cmd" name will be shown in the title line of the DOS window.

3. Open (run) startKafka.cmd.

   As above, the server will produce some log output.

4. Repeat with **startKafka-server1.cmd** and **startKafka-server2.cmd**.

Page 11 of 37   The KafkaProducer node   Version 10.0.0.7
                              Kafka servers and IBM MessageHub
                                 Provided by IBM BetaWorks

5. At this point, all Kafka servers are running, so now create a new topic.

    Open a new DOS window, and change directory to `c:\student10\kafka\commands`

6. Run the **createTopic.cmd** file.

    Provide the following values:
    - Topic: employee
    - Replication factor: 3
    - Partitions 2

    ```
    C:\student10\Kafka\commands>createTopic.cmd

    C:\student10\Kafka\commands>echo off
    Enter topic to create (default is employee): employee
    Enter Replication Factor for employee (default is 1): 3
    Enter number of partitions to create (default is 1): 2
    Created topic "employee".
    ```

7. Run the command file `listTopics.cmd`.

    The command will return "employee".

    ```
    C:\student10\Kafka\commands>cmd /c "kafka-topics.bat --list --zookeeper localhost:2181"
    employee
    ```

8. Run the command file `describeTopic.cmd`.

    Provide "**employee**" as the topic name. The command will return information about the replication factor and partitions of the "employee" topic. If you have followed the instructions above, you will see output similar to that below.

    ```
    C:\student10\Kafka\commands>describeTopic.cmd

    C:\student10\Kafka\commands>echo off
    Enter topic to describe (default is employee): employee
    Topic:employee   PartitionCount:2        ReplicationFactor:3     Configs:
            Topic: employee Partition: 0     Leader: 2       Replicas: 2,0,1 Isr: 2,0,1
            Topic: employee Partition: 1     Leader: 0       Replicas: 0,1,2 Isr: 0,1,2
    ```

9. Run the command `consumeMessages.cmd`.

    Specify the "employee" topic, and connect to the Kafka server with port 9092.

    ```
    C:\student10\Kafka\commands>consumeMessages.cmd

    C:\student10\Kafka\commands>echo off
    Enter topic that you want to consume from (default is employee): employee
    Enter port that you want to connect to (server=9092, server1=9093, server2=9094, default is 9092): 9
    092
    ```

Page 12 of 37   The KafkaProducer node   Version 10.0.0.7
                                    Kafka servers and IBM MessageHub
                                        Provided by IBM BetaWorks

10. Open a further DOS windows, and navigate to c:\student10\kafka\commands.

Run the command **produceMessage.cmd**.

Specify the "employee" topic, and connect to Kafka with port 9092.

Type some text message input, as shown below. Each message is terminated with the Return key.

```
C:\student10\Kafka\commands>produceMessage.cmd

C:\student10\Kafka\commands>echo off
Enter topic you want to produce messages to (default is employee): employee
Enter port that you want to connect to (server=9092, server1=9093, server2=9094, default is 9092): 9
092
test message 1
test message 2
final message
```

11. Back in the consumeMessages window, observe that the text messages you just produced have been consumed by the consumeMessages client application.

```
C:\student10\Kafka\commands>consumeMessages.cmd

C:\student10\Kafka\commands>echo off
Enter topic that you want to consume from (default is employee): em
Enter port that you want to connect to (server=9092, server1=9093,
092
test message 1
test message 2
final message
```

You have now verified that the local Kafka system is configured correctly, and can be used by the IIB applications.

Page 13 of 37   The KafkaProducer node   Version 10.0.0.7
                                Kafka servers and IBM MessageHub
                                   Provided by IBM BetaWorks

3. Import and extend the HR_Service REST API

In this section you will extend the provided REST API, HR_Service. The provided version of HR_Service has already implemented the two createEmployee* operations. Both these operations use the createEmployeeMain.subflow. This subflow is the component that will be extended in this section.

1.   If you already have a workspace open, click File, Switch Workspace. Give the new workspace the name

   c:\users\iibuser\IBM\IIB 10\workspace_kafka

2.   In the new workspace, import the Project Interchange file:

   **C:\student10\HR_complete\HR_Service\solution\
   HR_Service_Kafka_StartingPoint.10.0.0.7.zip**

   Select all three projects from this PI file, and click Finish.

   HR_Service uses the EMPLOYEE, EMPLOYEE_SUPPLEMENTARY and DEPARTMENT tables from the HRDB database. This requires the HRDB Database Definition project, which represents the tables schemas. This is used by the Mapping nodes that access these tables. The HRDB Shared Library and HRDB_project items contain the database definitions for the DB2 database HRDB.

   Note – because the EMPLOYEE_SUPPLEMENTARY table has been added in the 10.0.0.7 version of this lab series, the HRDB.dbm file has been updated in this version of the HRDB shared library. However, it is backward-compatible with earlier versions, and can be used with earlier lab scenarios from this series.


The KafkaProducer node   Version 10.0.0.7
                      Kafka servers and IBM MessageHub
                         Provided by IBM BetaWorks
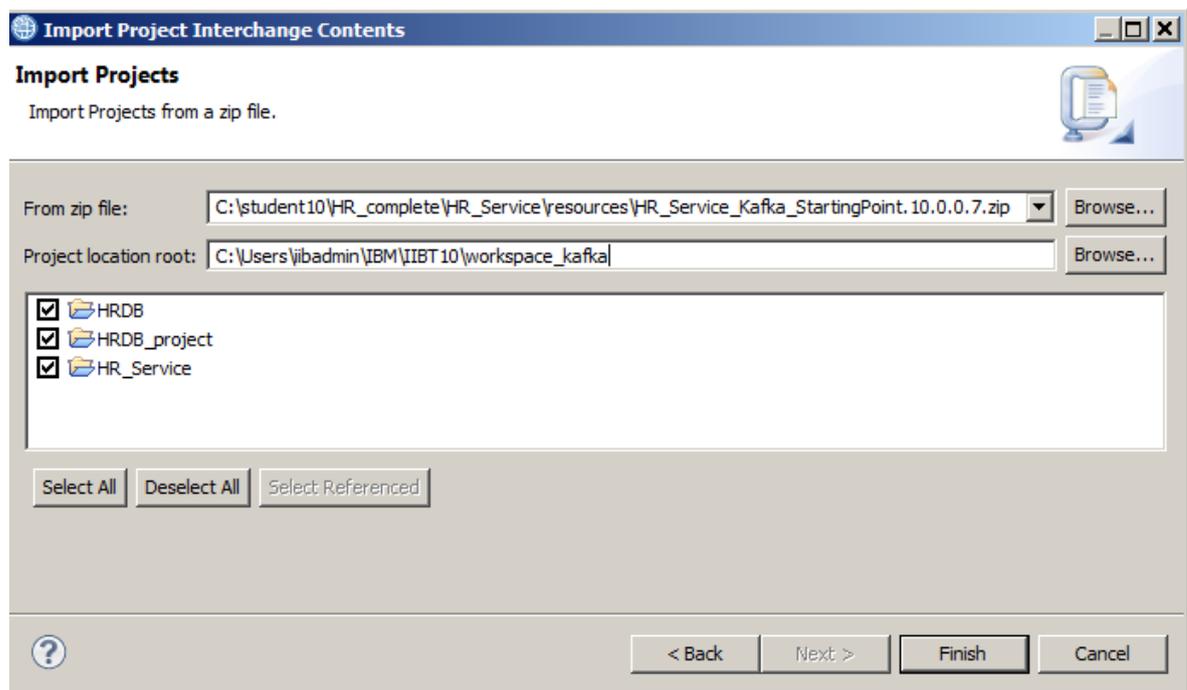
3.   When imported, you will see the HR_Service REST API project, and the HRDB shared library. The shared library has a library reference from the HR_Service REST API.

You will see several subflows are present in HR_Service. This indicates that several operations have already been implemented in this REST API. In addition, there are other subflows that are for more general use.

Expanding the REST API Catalog will show the entire list of operations that are defined in this REST API. As mentioned above, not all of these operations have been implemented.

Note that the "HR Employee and Department Services.json" document has been updated to Version 4.0.0. This is because new Model Definitions have been provided for the new database table, and all "Get" operations have been changed to use the "query parameter" form of a REST request with a "get" method.

Page 15 of 37   The KafkaProducer node   Version 10.0.0.7
                              Kafka servers and IBM MessageHub
                                  Provided by IBM BetaWorks
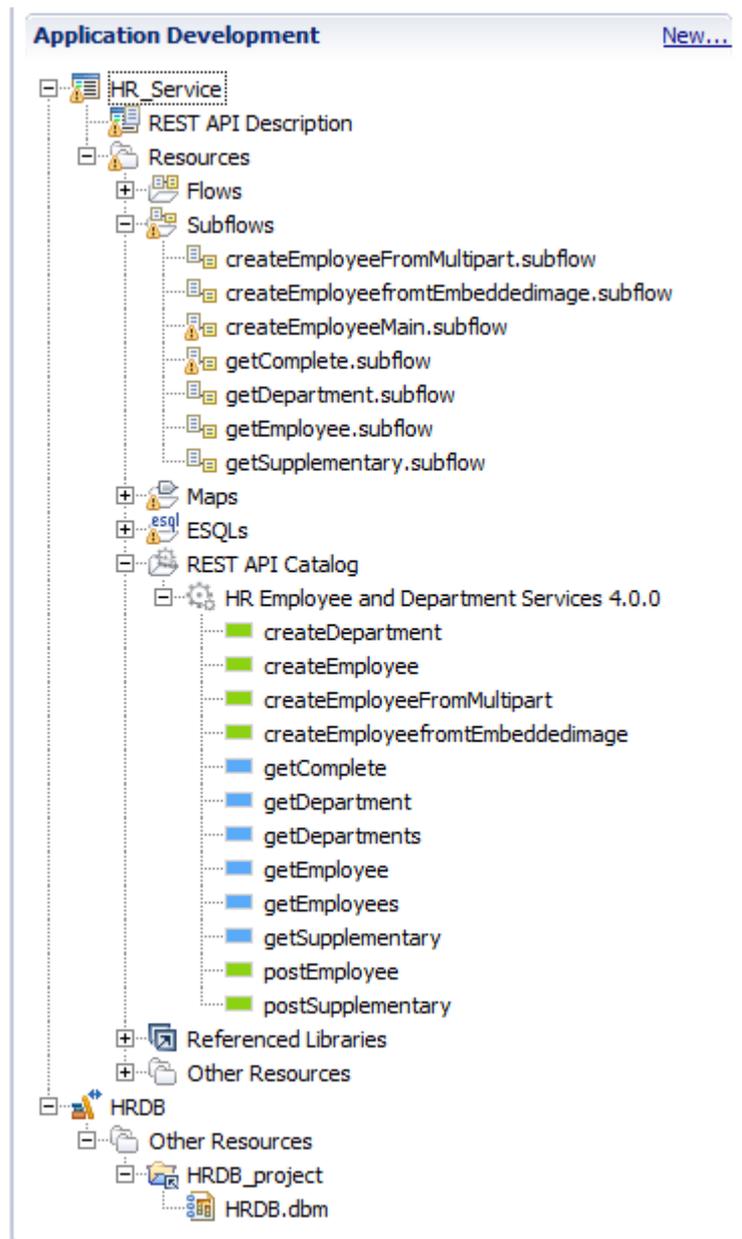
4. Open (double-click) the REST API Description. This will show the Header and Resources definitions in the REST API.

The "get" operations are not described here. If you need information on how to construct these operations, please refer to various lab documents earlier in this series, for example 10006_16L01/02.

Page 16 of 37   The KafkaProducer node   Version 10.0.0.7
                              Kafka servers and IBM MessageHub
                                   Provided by IBM BetaWorks

5. Further down the HR_Service editor, you will see the supplied Model Definitions.

Expand the supplied models. These will be used later in the lab.

6. Expand the **/employees/complete/embeddedImage** resource.

You will see a POST operation has been created in this resource.

Page 17 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks

7. Use the slide bar to move to the right, and open the subflow that implements this operation.
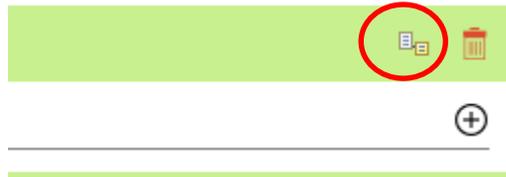
8. The subflow invokes a further subflow, createEmployeeMain, where the main logic for this operation is provided.
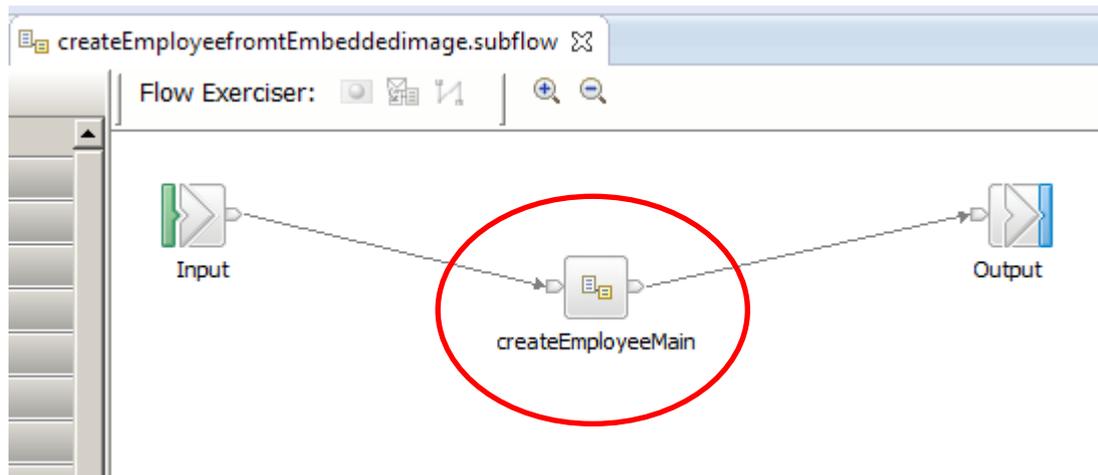
9. Similarly, in the /**employees/complete/multipart** resource, the **createEmployeeFromMultipart** operation also invokes the same subflow.

Page 18 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks
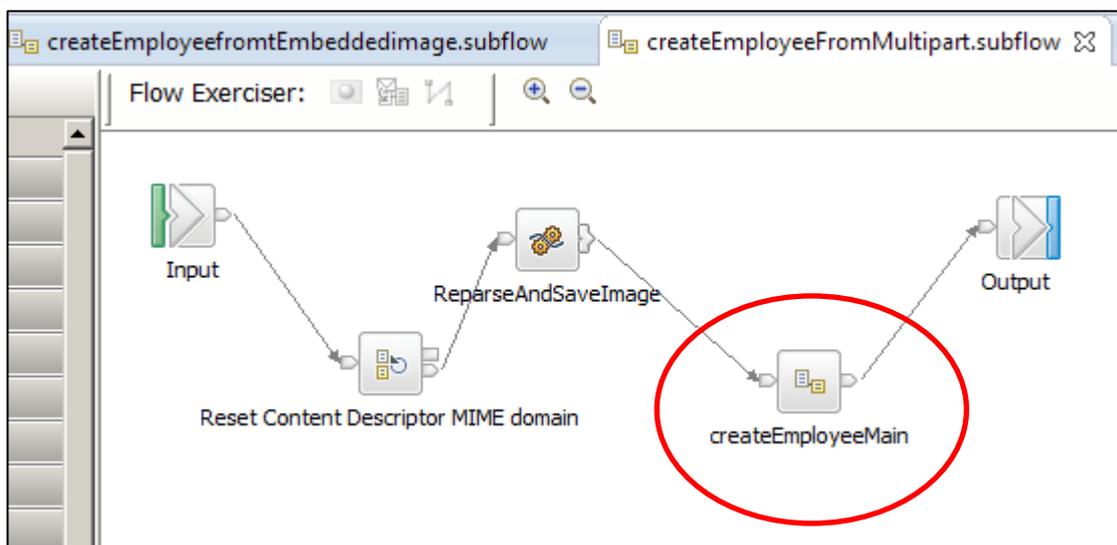
3.1 Investigate the createEmployeeMain subflow

1. From either of the two operation subflows mentioned above, open (double-click) the createEmployeeMain subflow.

    This subflow is complete, with the exception of a space for a KafkaProducer node.

    However, there are some additional features of some other nodes which will be explored, although no further development work is required for these.
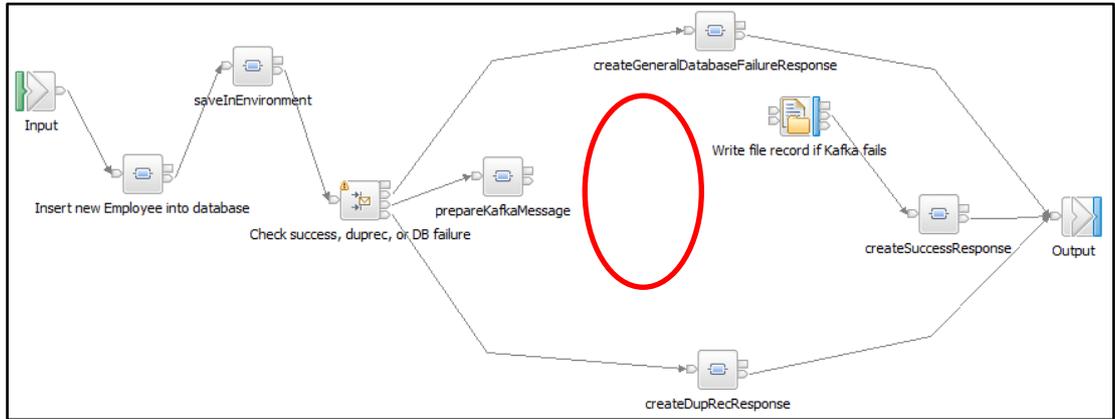

2. Open the "Insert new Employee into database" mapping node. This map accesses three database tables, all in the HRDB database, as follows:

    - Select from HRDB(DEPARTMENT)
    - Insert into HRDB(EMPLOYEE)
    - Insert into HRDB(EMPLOYEE_SUPPLEMENTARY)

    If all database operations are successful, all data is placed in the output message assembly, with the exception of the binary image element of the input message (see step 6 below).

Page 19 of 37   The KafkaProducer node   Version 10.0.0.7
                                Kafka servers and IBM MessageHub
                                   Provided by IBM BetaWorks
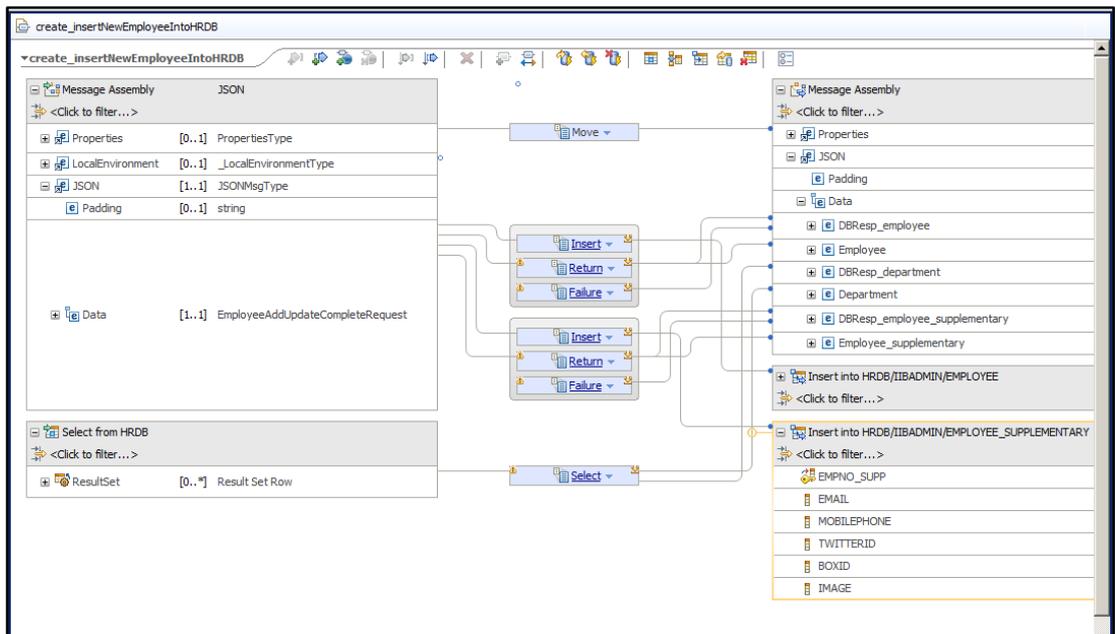
3.   Highlight the first Insert transform. The input Data element and the output "Insert into HRDB/IIBADMIN/EMPLOYEE" elements are highlighted.

If you want to see the precise elemenrts mappings, click the "Insert" text of the transform.

Return to the highest level of the map when finished.
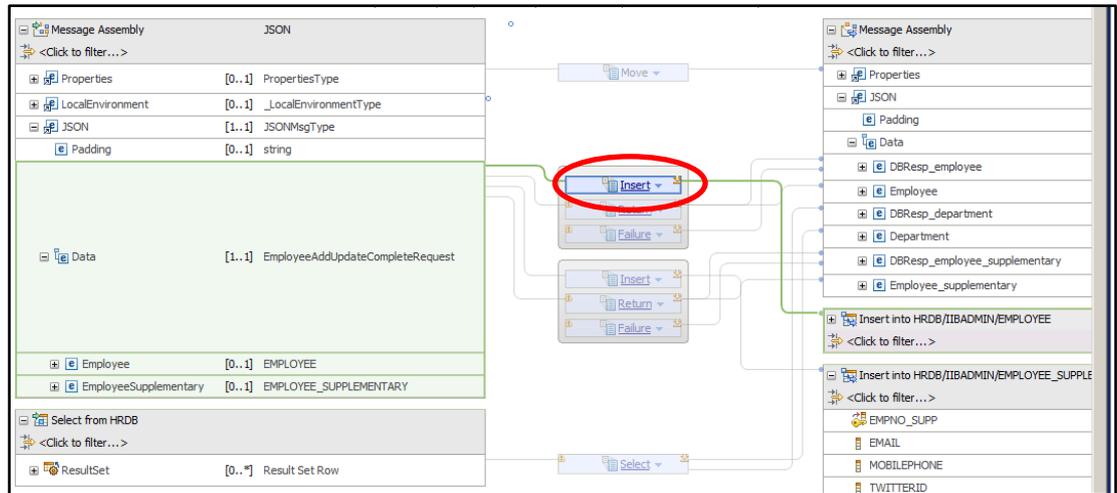
4.   Similarly, highlight (click) the second Insert transfom (not the "Insert" text). The same input will be highlighted, and the output assemly for the EMPLOYEE_SUPPLEMENTARY table will be highlighted.

Again, if required, click the word "Insert" to see the precise mappings.
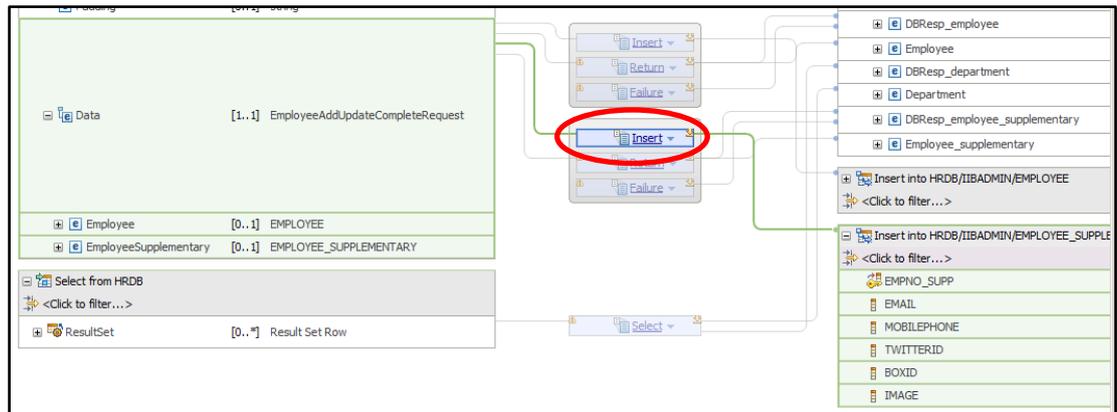
5.   Click one of the Return transforms. Note the input and output connections that have been made for this transform.

Page 20 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks
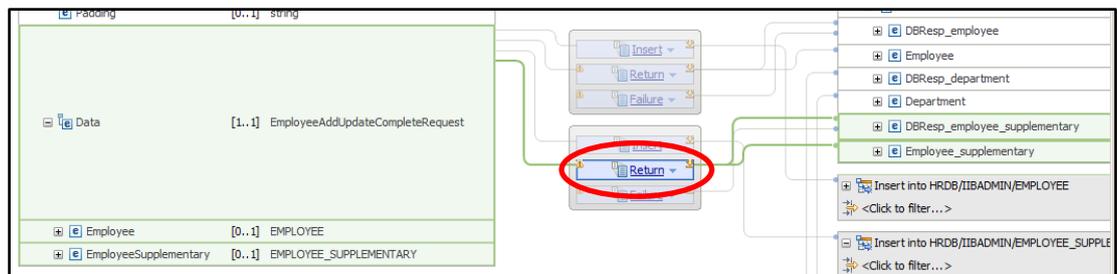
6. Finally in this map, click the word "Return" to see the precise element mappings.

    Note that the IMAGE element is not mapped. This is because this element will probably be quite large, and we don't want to send this back to the originating client.
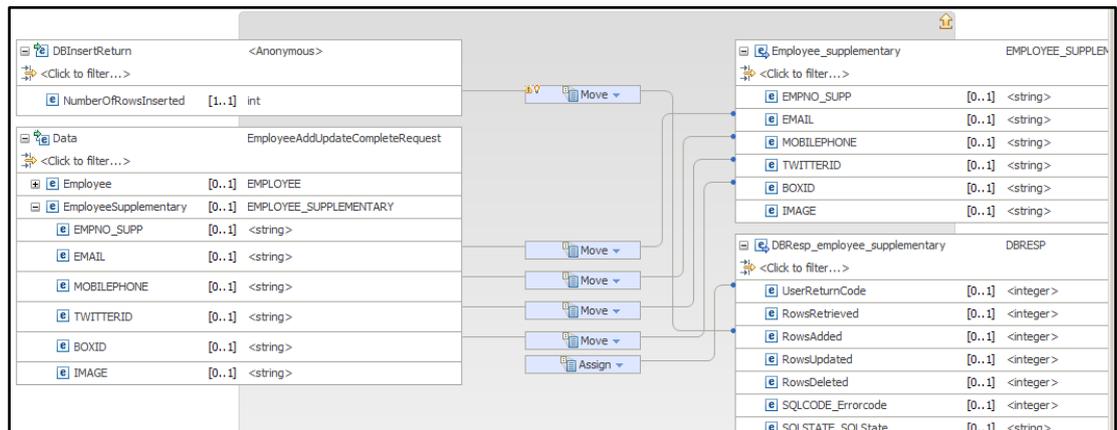
    Close the map.

7. Highlight the "Check success, duprec, or DB failure" Route node.

    In the properties of this node, the filter pattern has been set to check for successful inserts to both the EMPLOYEE and EMPLOYEE_SUPPLEMENTARY tables. If RowsAdded=1 for both of these tables, then the message will be sent to the Match terminal, and to the KafkaProducer node (to be added). All other responses represent failures of various kinds.

8. Open the prepareKafkaMessage map.

    The output message assembly has a single JSONObject element, employeeAdded. This element is set to the value of the employee number (EMPNO), before being published to the Kafka server in the subsequent node. Consuming applications will receive this small message, and be able to retrieve the full information from the EMPLOYEE and EMPLOYEE_SUPPLEMENTARY tables.

    Close the map.

Page 21 of 37   The KafkaProducer node   Version 10.0.0.7
                                Kafka servers and IBM MessageHub
                                   Provided by IBM BetaWorks
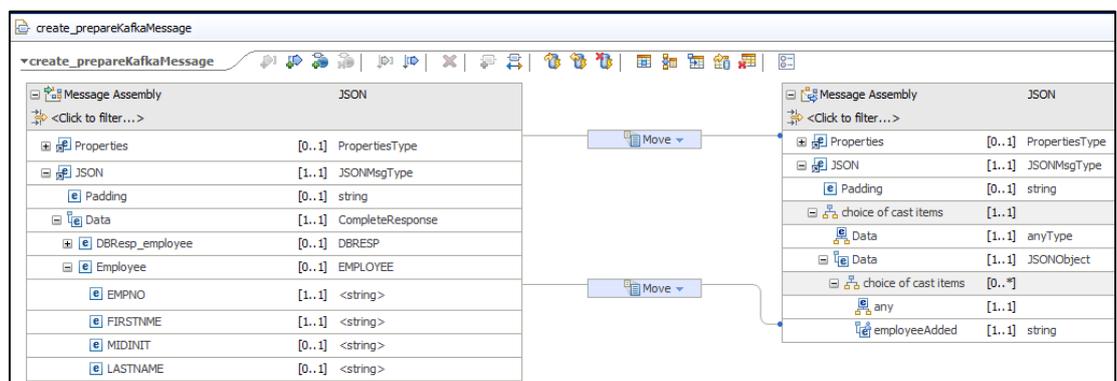
9. From the Kafka drawer, drop a KafkaProducer node onto the subflow, name it "KafkaProducer Local", and connect as shown.

   Connect the Failure terminal to the In terminal of the FileOutput node.
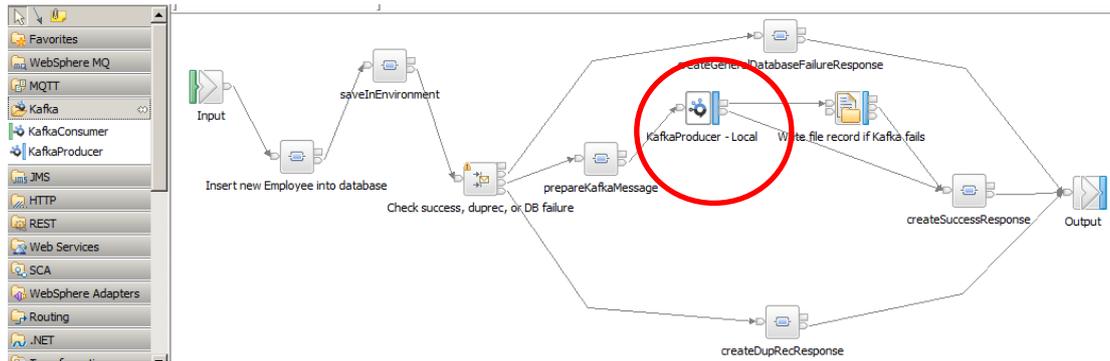
10. In the properties of the KafkaProducer node, set the following properties:

   - Topic: employee
   - Bootstrap server: localhost:9092
   - Acks: 1
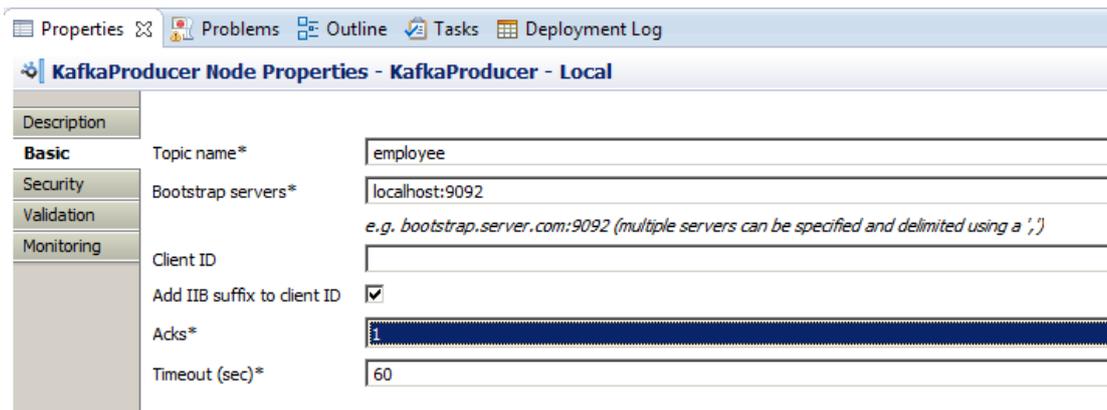
11. Review the properties of the FileOutput node.

The properties have been specified to work on a Windows system, and to write an output file to the folder **c:\kafka\files_noKafka**, if the KafkaProducer node should fail.

If you are running on a non-Windows system, or you want to change the output folder destination or file name, make appropriate changes now to these properties.
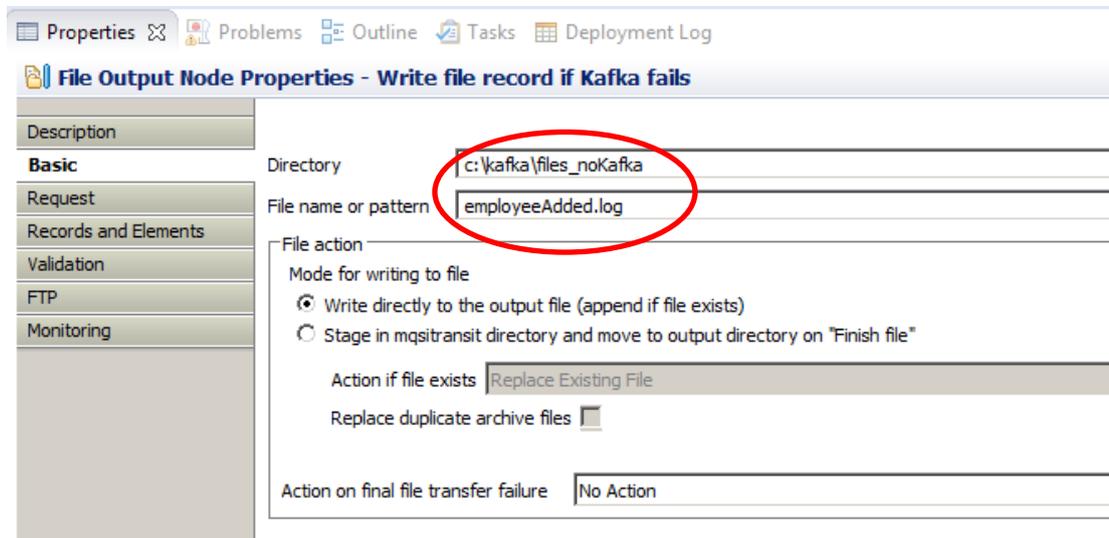
12. Save the subflow.

Deploy the **HRDB** Shared Library and then the **HR_Service** REST API to TESTNODE_iibuser/default in the usual way.

Page 23 of 37   The KafkaProducer node   Version 10.0.0.7
                          Kafka servers and IBM MessageHub
                             Provided by IBM BetaWorks

4. Test the updated HR_Service REST API

4.1 Using SOAPUI

1.   Open SOAPUI and create a new REST project.
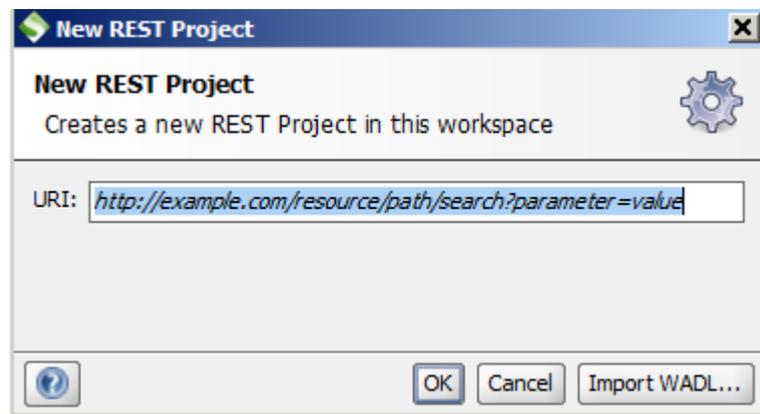
2.   In the URI field, specify the following URI, and click OK.

   http://localhost:7800/HR_Services/resources/employees/complete/embeddedimage

The KafkaProducer node
                          Kafka servers and IBM MessageHub
                              Provided by IBM BetaWorks
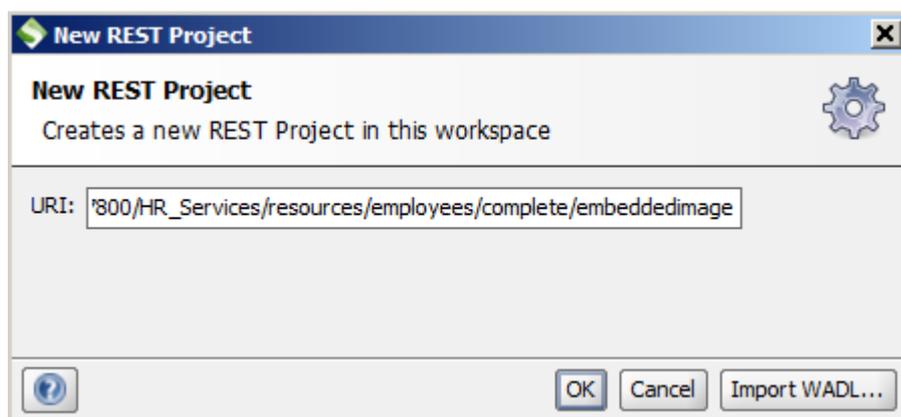
3.   Change the method to a POST.

    In the input data area, paste the entire contents of the file

**c:\student10\HR_complete\HR_Service\data\
EmbeddedImageWholeMessage.raw.json**

Note that the last data line (highlighted in yellow below) is a binary image of the new employee, encoded as a Base64 text encoded string.

Page 25 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks

4.   Change the EMPNO to an employee number that does not already exist in the EMPLOYEE
      table (000028 in this example).

      Click the green arrow to send the request.

Page 26 of 37   The KafkaProducer node   Version 10.0.0.7
                              Kafka servers and IBM MessageHub
                                 Provided by IBM BetaWorks

5.   If successful, the response message will be shown.

```
/HR_Services/resources/employees/complete/embeddedimage

<Response xmlns="http://localhost/HR_Services/resources/emp
    <DBResp_department>
        <RowsRetrieved>1</RowsRetrieved>
        <UserReturnCode>0</UserReturnCode>
    </DBResp_department>
    <DBResp_employee>
        <RowsAdded>1</RowsAdded>
        <UserReturnCode>0</UserReturnCode>
    </DBResp_employee>
    <DBResp_employee_supplementary>
        <RowsAdded>1</RowsAdded>
        <UserReturnCode>0</UserReturnCode>
    </DBResp_employee_supplementary>
    <Department>
        <ADMRDEPT>A00</ADMRDEPT>
        <DEPTNAME>SPIFFY COMPUTER SERVICE DIV.</DEPTNAME>
        <DEPTNO>A00</DEPTNO>
        <LOCATION null="true"/>
        <MGRNO>000010</MGRNO>
    </Department>
    <Employee>
        <BIRTHDATE>1879-03-14</BIRTHDATE>
        <BONUS>5000</BONUS>
        <COMM>1024</COMM>
        <EDLEVEL>9</EDLEVEL>
        <EMPNO>000028</EMPNO>
        <FIRSTNME>Albert</FIRSTNME>
        <HIREDATE>2016-12-12</HIREDATE>
```

6.   Switch back to the DOS command window which was consuming messages from the Kafka
     employee topic.

     Note that a message has been received with the employee number of the newly added
     employee (000028 in this example).

```
C:\student10\Kafka\commands>echo off
Enter topic that you want to consume from (default is employee): employee
Enter port that you want to connect to (server=9092, server1=9093, server2
092
test message 1
test message 2
final message
{"employeeAdded":"000026"}
{"employeeAdded":"000027"}
{"employeeAdded":"000028"}
```

Page 27 of 37   The KafkaProducer node   Version 10.0.0.7
                              Kafka servers and IBM MessageHub
                                  Provided by IBM BetaWorks

4.2 Using Postman

Although most REST testing tools, such as SOAPUI, are capable of generating and sending a REST request with an accompanying JSON payload, not all are capable of sending a MIME request in multipart format. The HR_Service REST API has two operations to create a new employee. The second of these expects such a message, so this section uses the Chrome Postman tool to send this request. Postman is described in more detail in the lab guide 10006_16L15 document.

1. Open the Postman tool and import the Postman collection "Complete". This is available in the export file

    c:\student10\HR_complete\HR_Service\postman\ Complete.postman_collection.json

2. Expand Complete, and select the "Add (Complete) – using multipart image" test.

    On the details pane, select the Body tab. The data format will be set to "form-data".
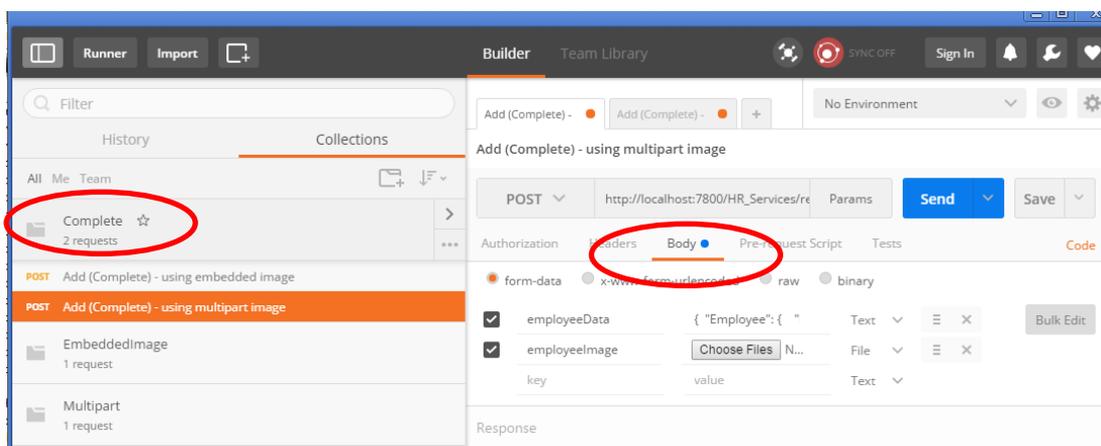
3. For the **employeeImage** part of the message, set the attached file to

    c:\student10\HR_complete\HR_service\data\Einstein.jpg

    Set the EMPNO element to an employee value that is not yet in the database. The example below shows "000029".

Page 28 of 37   The KafkaProducer node   Version 10.0.0.7
                          Kafka servers and IBM MessageHub
                             Provided by IBM BetaWorks
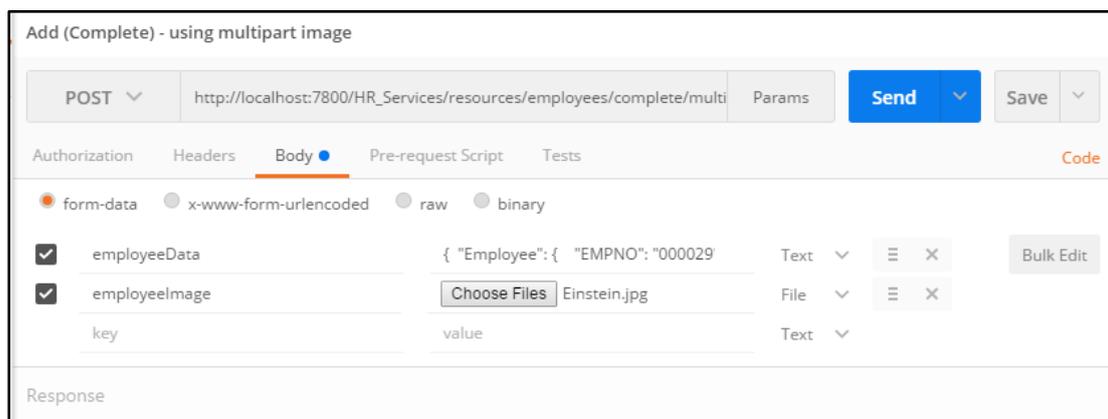
4.   Click Send.

The new employee will be added successfully.

The new employee will be published to Kafka, and the consuming application will receive the new notification, as before.

```
C:\student10\Kafka\commands>echo off
Enter topic that you want to consume from (default is employee): employee
Enter port that you want to connect to (server=9092, server1=9093, server2=9094,
092
test message 1
test message 2
final message
{"employeeAdded":"000026"}
{"employeeAdded":"000027"}
{"employeeAdded":"000028"}
{"employeeAdded":"000029"}
```

Page 29 of 37   The KafkaProducer node   Version 10.0.0.7
                                 Kafka servers and IBM MessageHub
                                     Provided by IBM BetaWorks

5. Using the Kafka nodes with IBM MessageHub

The IIB Kafka Producer and Consumer nodes can also be used with the MessageHub service provided on IBM Bluemix. MessageHub is IBMs implemention of Apache Kafka. More information can be found here: https://developer.ibm.com/messaging/message-hub/.

5.1 Explore and configure MessageHub

1. To use the Bluemix MessageHub service, you will need to login with an IBM ID. If you do not have an IBM ID, create one now, and then return to the next step.

2. Login to the MessageHub service on IBM Bluemix:

    https://console.ng.bluemix.net/catalog/services/message-hub

    When you have logged in, make sure that the Bluemix Region is set to the value that you want to use for your own testing. In the example below, the region for the BetaWorks organization is "United Kingdom".


Page 30 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks
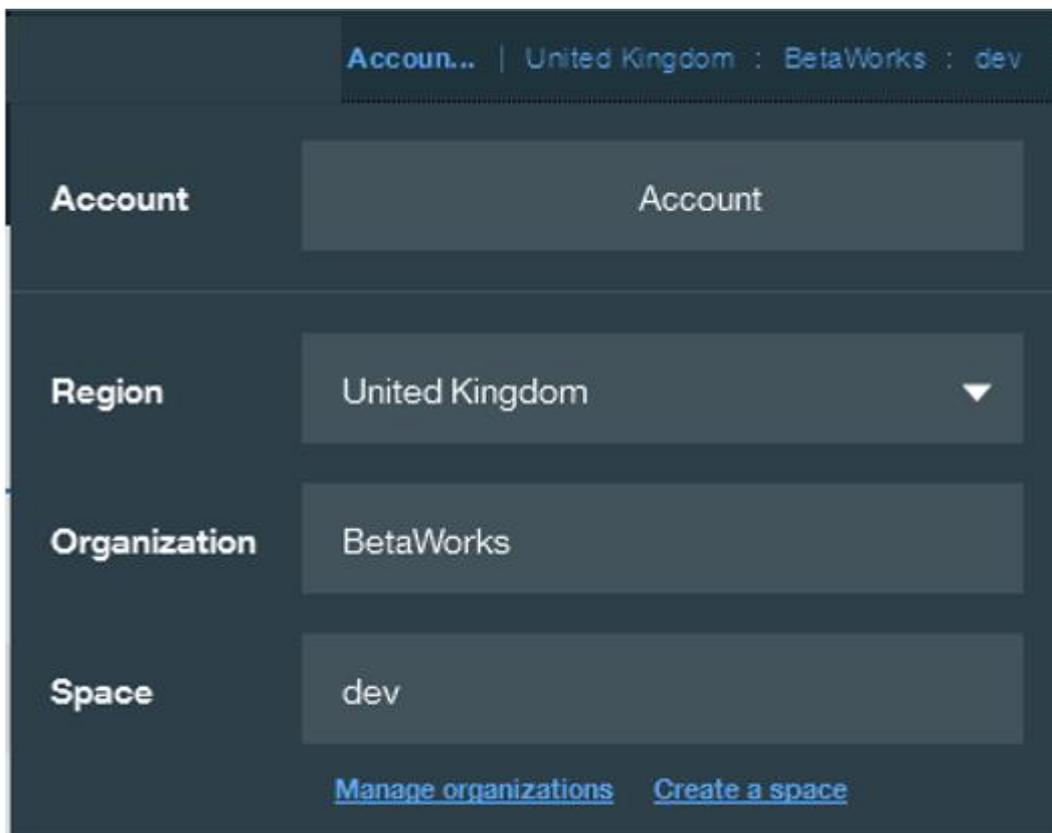
3.   In the Bluemix Catalog, type "Message Hub" into the search field.

       Click the Message Hub icon.
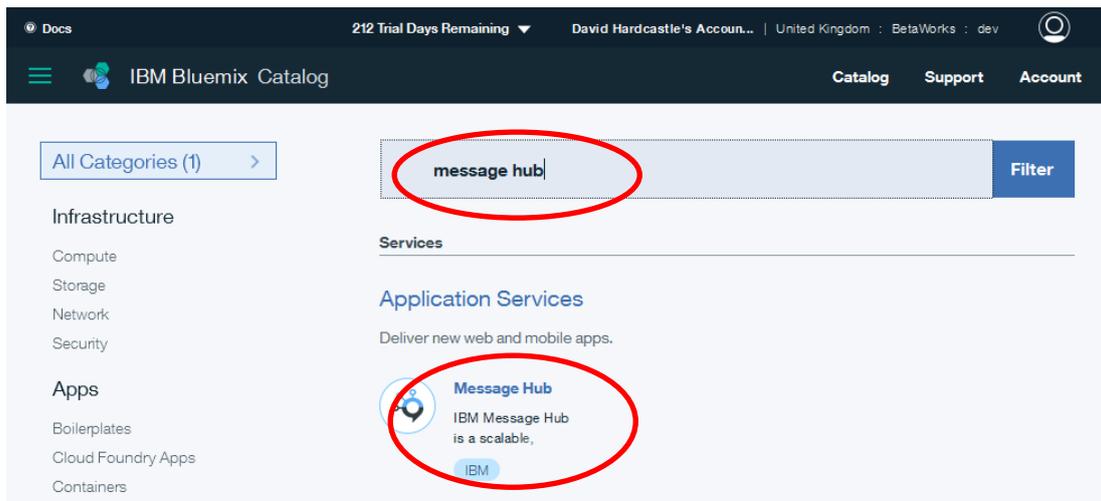
4.   Click Create to create a new Message Hub service.

Page 31 of 37   The KafkaProducer node   Version 10.0.0.7
                        Kafka servers and IBM MessageHub
                        Provided by IBM BetaWorks
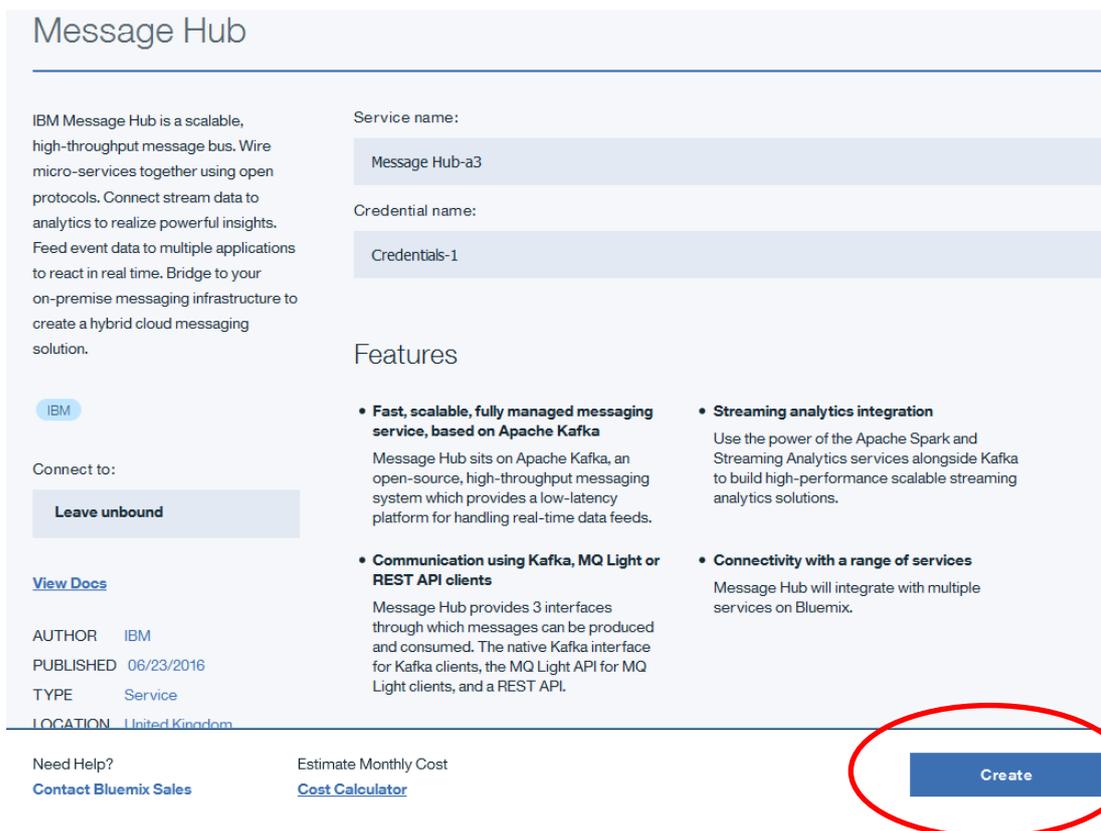
5.   The new Message Hub service will be created.
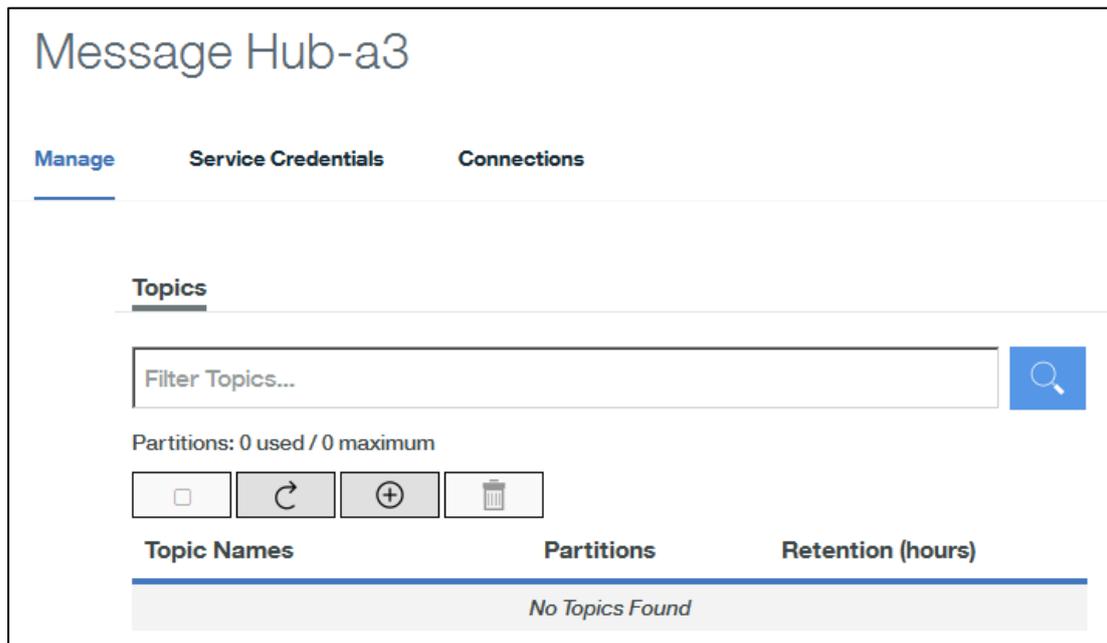
6.   Create a new topic, by clicking the "Create topic" icon.

Page 32 of 37   The KafkaProducer node   Version 10.0.0.7
                            Kafka servers and IBM MessageHub
                              Provided by IBM BetaWorks
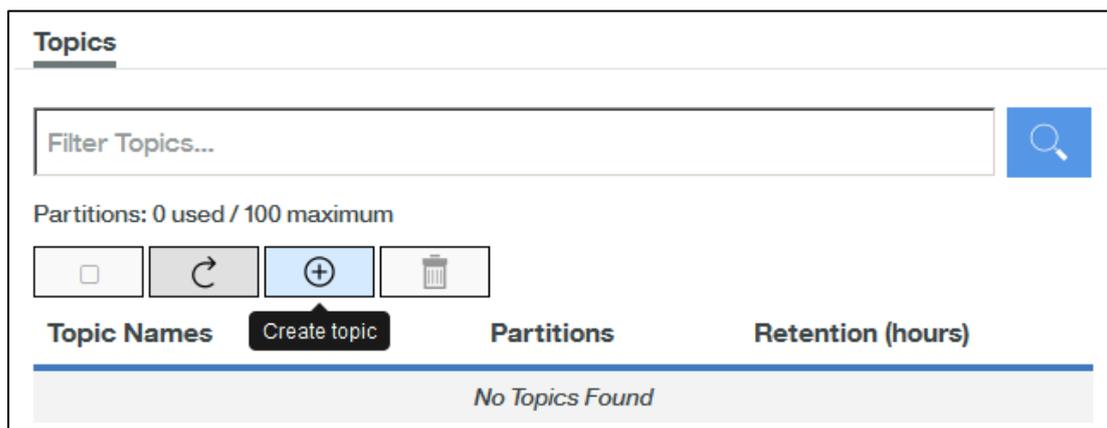
7.   Name the new topic "employee" and press Return.

Topic Names	Partitions	Retention (hours)		
employee	1	24	Save	⊗
No Topics Found				

The new topic will be created with a retention period of 24 hours.

Topic Names	Partitions	Retention (hours)	
Topic 'employee' created.			⊗
☐ employee	1	24	

8.   Click the "**Service Credentials**" tab.

Click the "View Credentials" tab.

Note the server URL values, and the user and password values. These will be used by the IIB KafkaProducer node that you update shortly.

Message Hub-a3

Manage   Service Credentials   Connections

Service Credentials

Credentials are provided in JSON format. The JSON snippet lists credentials, such as the API key and secret, as well as connection information for the service.

Service Credentials   New Credential

| ☐ KEY NAME	DATE CREATED	ACTIONS
☐ Credentials-1	Dec 14, 2016 - 05:05:39	View Credentials ▲

```
{
    "mqlight_lookup_url": "https://mqlight-lookup-prod02.messagehub.services.eu-gb.bluem
ix.net/Lookup?serviceId=845afee0-a856-4726-a2c5-af90b2e7ebd7",
    "api_key": "yt3ApcyClTbrJ7H9nb0zXz466PH0cRvDmNAhhwZ3FdBgJoRl",
    "kafka_admin_url": "https://kafka-admin-prod02.messagehub.services.eu-gb.bluemix.net
:443",
    "kafka_rest_url": "https://kafka-rest-prod02.messagehub.services.eu-gb.bluemix.net:4
43",
    "kafka_brokers_sasl": [
        "kafka01-prod02.messagehub.services.eu-gb.bluemix.net:9093",
        "kafka02-prod02.messagehub.services.eu-gb.bluemix.net:9093",
        "kafka03-prod02.messagehub.services.eu-gb.bluemix.net:9093",
        "kafka04-prod02.messagehub.services.eu-gb.bluemix.net:9093",
        "kafka05-prod02.messagehub.services.eu-gb.bluemix.net:9093"
    ],
    "user": "yt3ApcyClTbrJ7H9",
    "password": "nb0zXz466PH0cRvDmNAhhwZ3FdBgJoRl"
}
```

Page 33 of 37   The KafkaProducer node   Version 10.0.0.7
                              Kafka servers and IBM MessageHub
                                    Provided by IBM BetaWorks

9. Return to the Integration Toolkit, and edit the createEmployeeMain subflow.

    Add a new KafkaProducer node to the flow. Name it "KafkaProducer MessageHub", and connect it as shown.
    Be sure to remove the connection to the KafkaProducer Local node – or you will get logic errors at runtime.
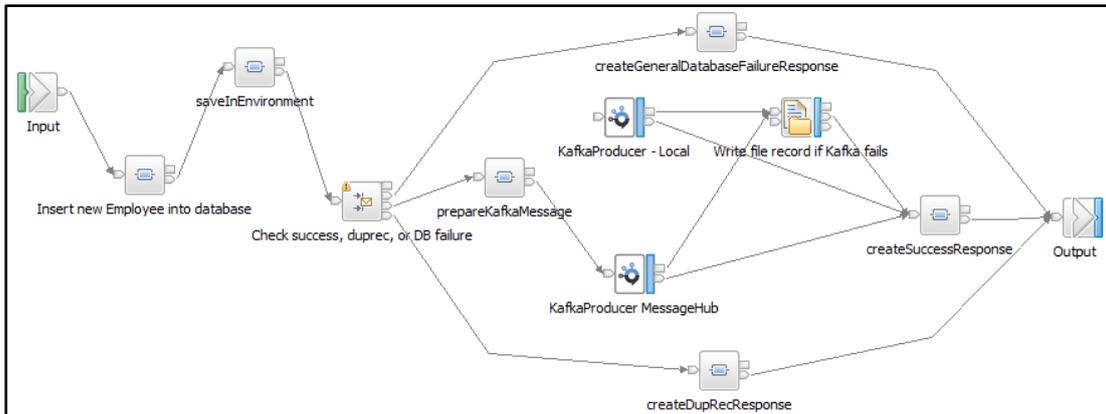
10. Select the properties of the KafkaProducer MessageHub node.

    On the Basic tab:
    - Topic: employee
    - Bootstrap server: kafka ……………bluemix.net:9093 (copy/paste the first url from the Service Credentials from your Bluemix MessageHub service)
    - Acks: 1
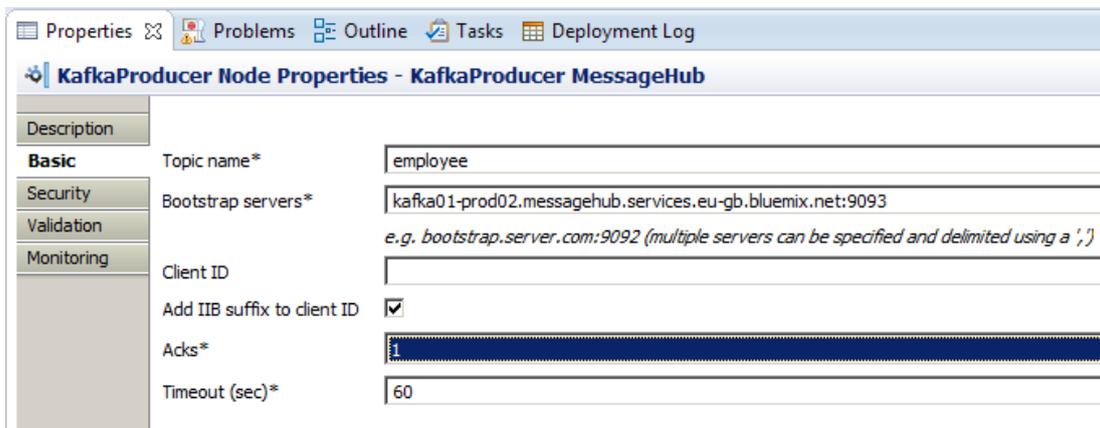
11. On the Security tab, set the security protocol to SASL_SSL.

Page 34 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks
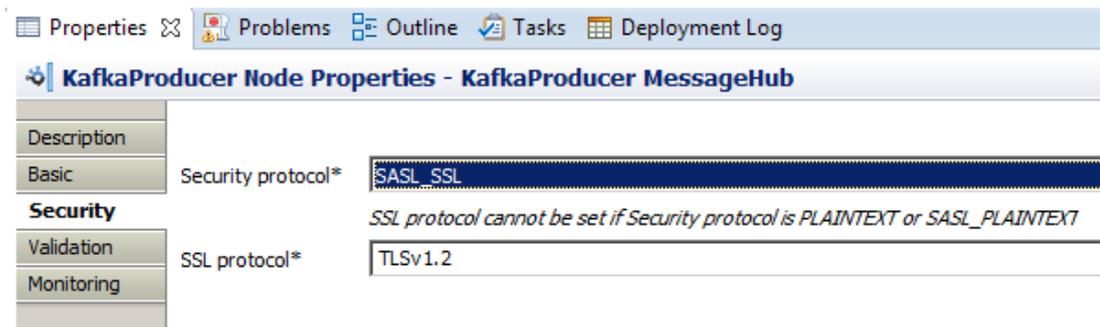
12. When connecting to a secure Kafka server, such as MessageHub, the required security
     credentials are provided by the IIB node. Security credentials are created with the
     "mqsisetdbparms" command.

     In an IIB Command Console, run the following command, substituting values for your IIB
     server, username and password as appropriate (user and password are those provided by
   the MessageHub credentials).

```
mqsisetdbparms TESTNODE_iibuser
      -n kafka::KAFKA::default
      -u yt3ApcyClTbrJ7H9
      -p nb0zXz466PH0cRvDmNAhhwZ3FdBgJoRl
```

13. Stop and restart the IIB node.

14. Save the subflow, and deploy the updated HR_Service in the usual way.

Page 35 of 37   The KafkaProducer node   Version 10.0.0.7
                                   Kafka servers and IBM MessageHub
                                       Provided by IBM BetaWorks

5.2 Test HR_Service with MessageHub

1.   Return to the SOAPUI application that was used earlier. Change the EMPNO to a value that
     has not been previously used.

     Click the green arrow to invoke.

Page 36 of 37   The KafkaProducer node   Version 10.0.0.7
                              Kafka servers and IBM MessageHub
                                  Provided by IBM BetaWorks
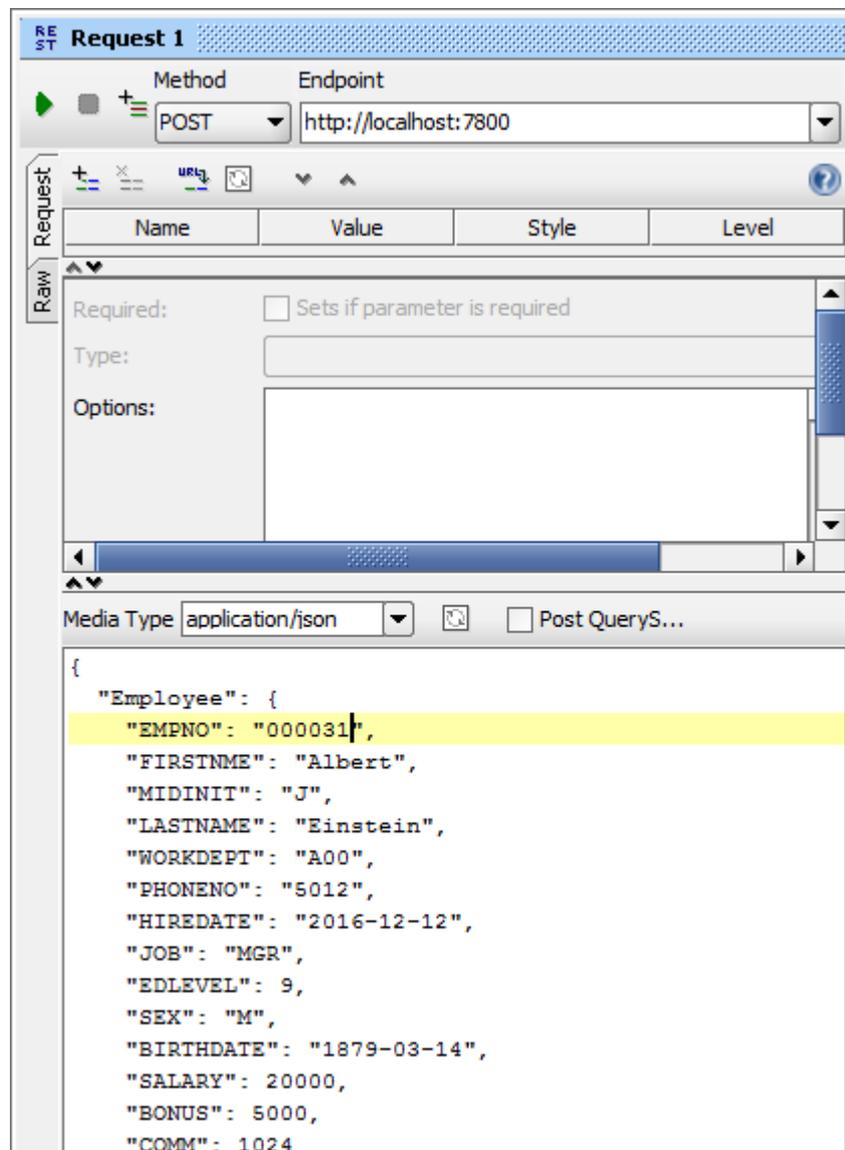
2.   In Bluemix, use the Grafana tools to view activity on the MessageHub system.

Note that to connect a client directly to the MessageHub system, you can use a variety of client tools. In all cases, you will have to configure appropriate security components to enable SSL or SASL-style connectivity, which is beyond the scope of this lab.

END OF LAB GUIDE

Page 37 of 37   The KafkaProducer node   Version 10.0.0.7
Kafka servers and IBM MessageHub
Provided by IBM BetaWorks