webMethods Knowledge Update 8.2 to 9.6 - Development

Exercise Guide

Software AG Internal Use Only

602D-66E

Software AG
Internal Use Only!

This publication is protected by international copyright law. All rights reserved. No part of this publication may be reproduced, translated, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of Software AG.

Software AG and all Software AG products are either trademarks or registered trademarks of Software AG. Other product or company names mentioned herein may be the trademarks of their respective owners.

 © 2000 – 2014 by Software AG

TABLE OF CONTENTS - BPM

Exercise 1: Process Modelling Improvements 5
Exercise 2: JMS and Universal Messaging Support 11
Exercise 3: Debugger Improvements 25
Exercise 4: BPMN Subprocesses 33
Exercise 5: EDA Process 39
Exercise 6: Predefined EDA Events 57
Exercise 7: Open Presentation Layer 63
Exercise 8a: Setup and test a BPM Process and User Task for Mobile Task Integration 69
Exercise 8b: Defining an MTI App in Mobile Administrator 79
Exercise 8c: Creating a Mobile Task Integration Application 81
Exercise 8d: Working on Tasks using your MTI App in Phoney 85
Exercise 8e (*optional*): Build and Publish Mobile Task Integration App 95

Software AG Internal Use Only!

This page intentionally left blank

Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 1:
PROCESS MODELLING IMPROVEMENTS

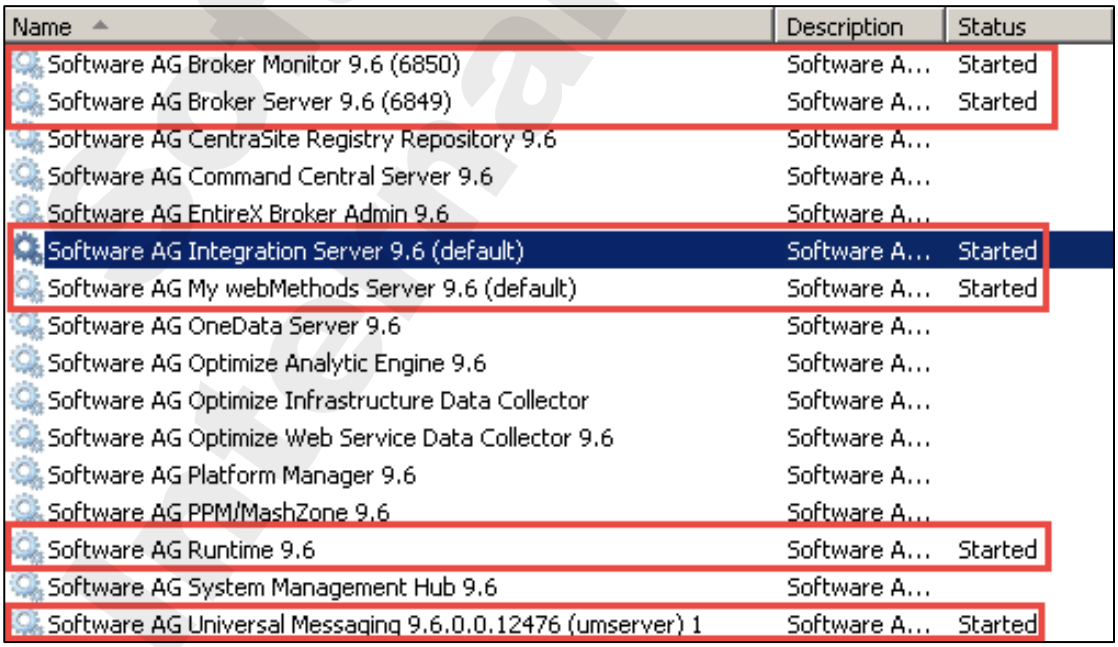
Objectives

In this exercise you will modify and enhance a provided process model in Designer to get familiar with improvements like speed panel, drag & drop support, resizing and label positioning. A skeleton of the process model has to be imported in advance.

In this exercise, please replace:
<workshop_dir> by **C:\Training\602D-66E\BPM**

Steps

1. Use the Service Control Panel and ensure the following Windows Services are up and running. If not, start them in the following sequence:
   - **Software AG Runtime 9.6**
   - **Software AG Broker Monitor 9.6**
   - **Software AG Broker Server 9.6**
   - **Software AG Universal Messaging 9.6 (umserver)**
   - **Software AG Integration Server**
   - **Software AG My webMethods Server 9.6 (default)**
   - **hMailServer**



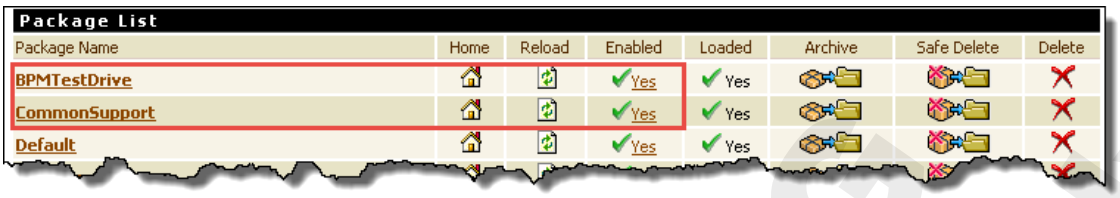
Name	Description	Status
Software AG Broker Monitor 9.6 (6850)	Software A...	Started
Software AG Broker Server 9.6 (6849)	Software A...	Started
Software AG CentraSite Registry Repository 9.6	Software A...	
Software AG Command Central Server 9.6	Software A...	
Software AG EntireX Broker Admin 9.6	Software A...	
Software AG Integration Server 9.6 (default)	Software A...	Started
Software AG My webMethods Server 9.6 (default)	Software A...	Started
Software AG OneData Server 9.6	Software A...	
Software AG Optimize Analytic Engine 9.6	Software A...	
Software AG Optimize Infrastructure Data Collector	Software A...	
Software AG Optimize Web Service Data Collector 9.6	Software A...	
Software AG Platform Manager 9.6	Software A...	
Software AG PPM/MashZone 9.6	Software A...	
Software AG Runtime 9.6	Software A...	Started
Software AG System Management Hub 9.6	Software A...	
Software AG Universal Messaging 9.6.0.0.12476 (umserver) 1	Software A...	Started

Important: These services should be up and running for all subsequent exercises.

Software AG Internal Use Only!

© 2000 – 2014 by Software AG

2. Use a browser tab and URL **http://localhost:5555** to open the Integration Server Administration console. Use **Administrator** | **manage** for authentication.
   In the IS Administration console visit **Packages** > **Management**. Ensure that packages **BPMTestDrive** and **CommonSupport** are enabled. If disabled, click the appropriate link in the **Enabled** column to enable.



3. Start Designer using the default workspace **C:\Users\Administrator\workspace96**.

   a. Open the **Process Development** perspective. Create a process project named **HRProcesses** in the workspace. If prompted for authentication for http://sagbase:8585 or http://localhost:8585 (MWS), provide **Sysadmin** | **manage** as user credentials.

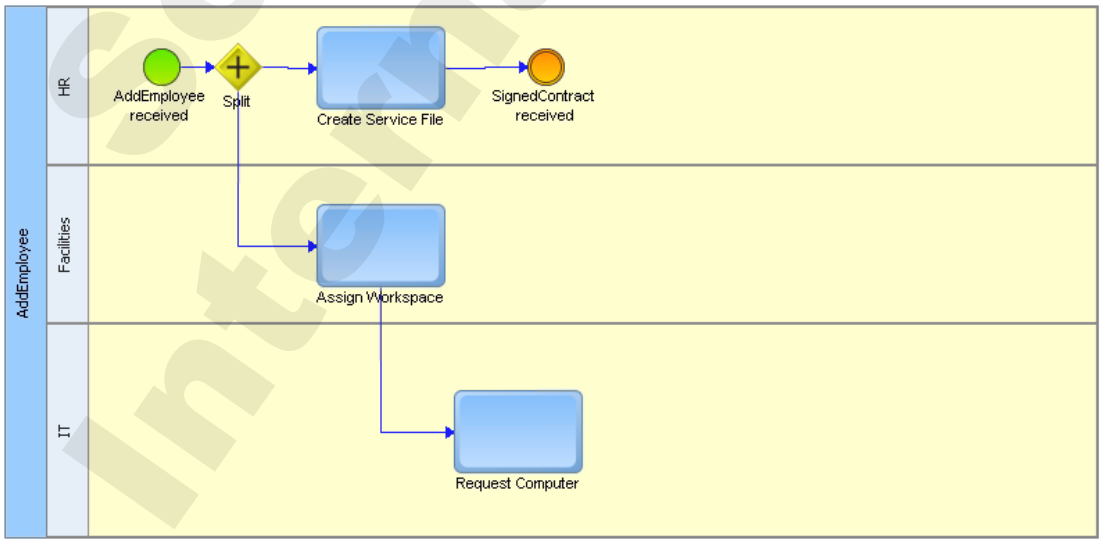
   b. Import process model **AddEmployeeNoTask** from the process file ***<workshop_dir>*\Ex1\Resources\Processes\AddEmployeeNoTask.process** into your process project created above.

   c. Open the process model **AddEmployee** in the process editor. Ensure you are working in the **Process Developer** mode .

   d. Use the Palette to add a Start None Event named **AddEmployee received** to the HR swimlane.

   e. Move the mouse over the Event to bring up the speed panel. From the speed panel, choose a Gateway to be added next to your Start Event in the HR swimlane. The transition will be added automatically. Right-click the Event to change the Gateway type to a **Parallel**. Rename the Gateway to **Split**.

   f. By using the speed panel, add three more Abstract Task Activities and an Intermediate None Event next to the Gateway. Arrange and rename them to correspond to the following screen shot:



Software AG Internal Use Only!

© 2000 – 2014 by Software AG

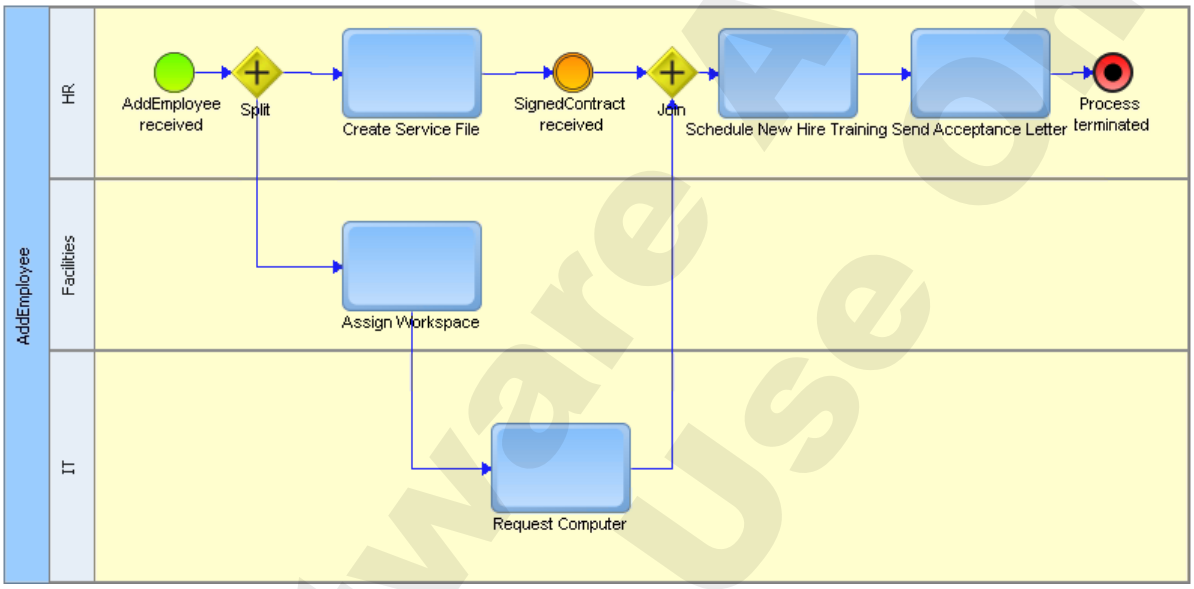
g. In the HR swimlane, add a Parallel Gateway joining Intermediate Event **SignedContract received** and Activity **Request Computer**. Name it **Join**.

h. Using the speed panel, add and connect two more Task Activities next to the joining Gateway in the HR swimlane. Name them **Schedule New Hire Training** and **Send Acceptance Letter.**

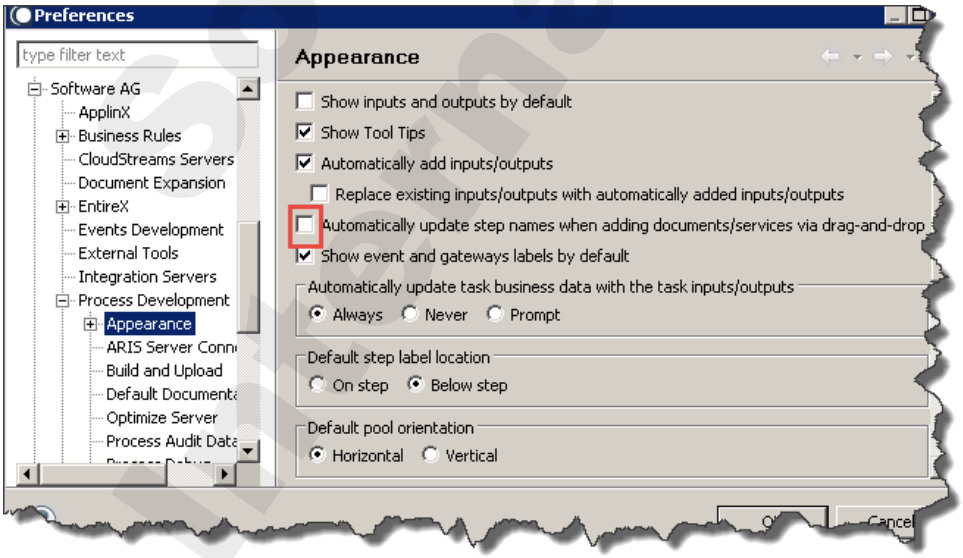
i. Using the speed panel to add and connect a final End Event next to Task **Send Acceptance Letter** in the HR swimlane. Rename your End None Event into **Process terminated**.

j. Right-click **Process terminated** to change the Event type to an End Terminate Event.

You process should now look like this:

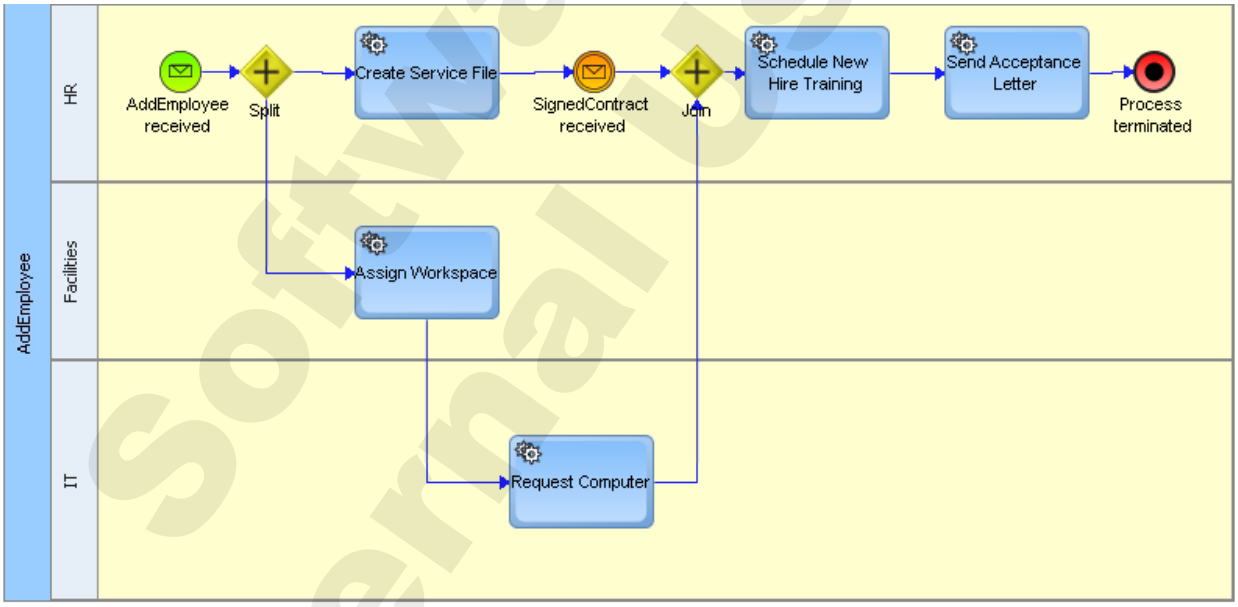


4. Ensure the Designer preference **Window** > **Preferences** > **Software AG** > **Process Development** > **Appearance**: **Automatically update step names…** is unchecked.



Software AG Internal Use Only!

5. From the Package Navigator view, drag IS services
   a. **bpmTestDrive.services:CreateServiceFile** onto your Task Activity **Create Service File**
   b. **bpmTestDrive.services:ScheduleNewHireTraining** onto your Task Activity **Schedule NewHire Training**
   c. **bpmTestDrive.services:SendAcceptanceLetter** onto your Task Activity **Send Acceptance Letter**
   d. **bpmTestDrive.services:AssignWorkspace** onto your Task Activity **Assign Workspace**
   e. **bpmTestDrive.services:RequestComputer** onto your Task Activity **Request Computer**
6. From the Package Navigator view, drag IS document **bpmTestDrive.docs:AddEmployee** onto your Start None Event **AddEmployee received**. This will change the Event type to a Start Message Event.
7. Right-click intermediate Event **SignedContract received,** and change Event type to **Message**. In the Properties view, set the Event to be a catching event (General tab) and browse for the Receive document of type **bpmTestDrive.docs:SignedContract** (Implementation tab). Leave all protocol properties unchanged.
8. Click the icon from the menu bar to position all labels on the corresponding steps. Your process model should now look like this:

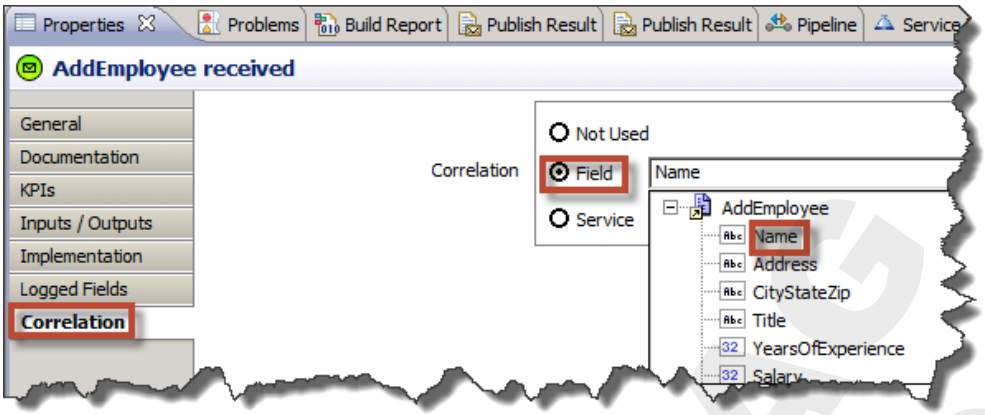


9. For all Task Activities and Start Message Event **AddEmployee received**, open the Inputs/Outputs tab of the Properties view to ensure the generated document names are always **AddEmployee** (not AddEmployee_1).
   Change the input/output parameter names if necessary.

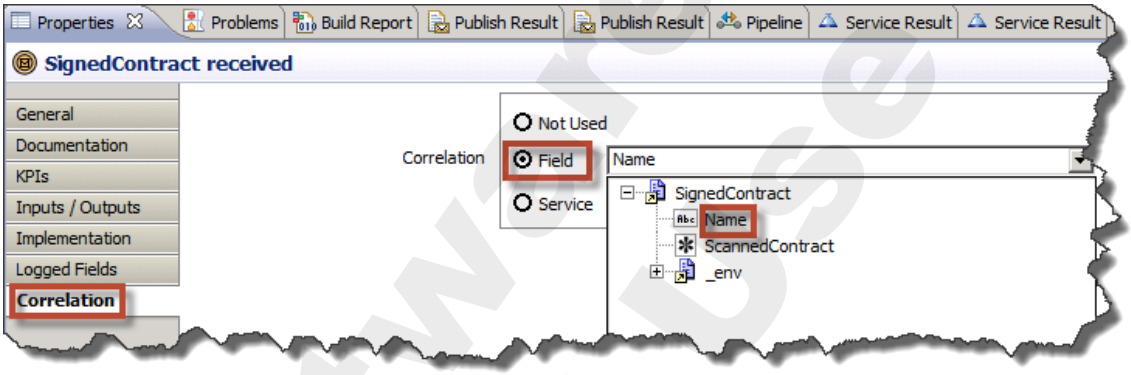
Software AG
Internal use only!

© 2000 – 2014 by Software AG

10. Use the properties view of **AddEmployee received** to specify a correlation ID based on the field **AddEmployee/Name**:

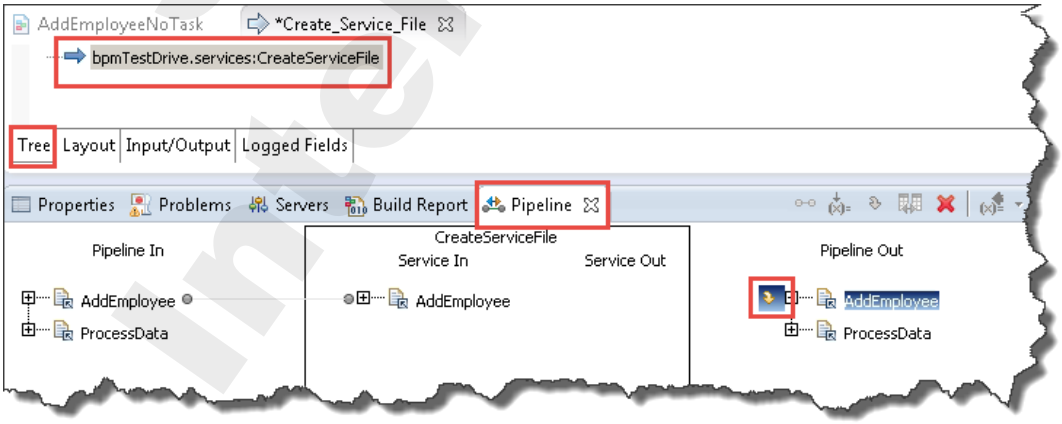


11. Use the properties view of **SignedContract received** to specify a correlation ID based on the field **SignedContract/Name**.



12. Save your process model.

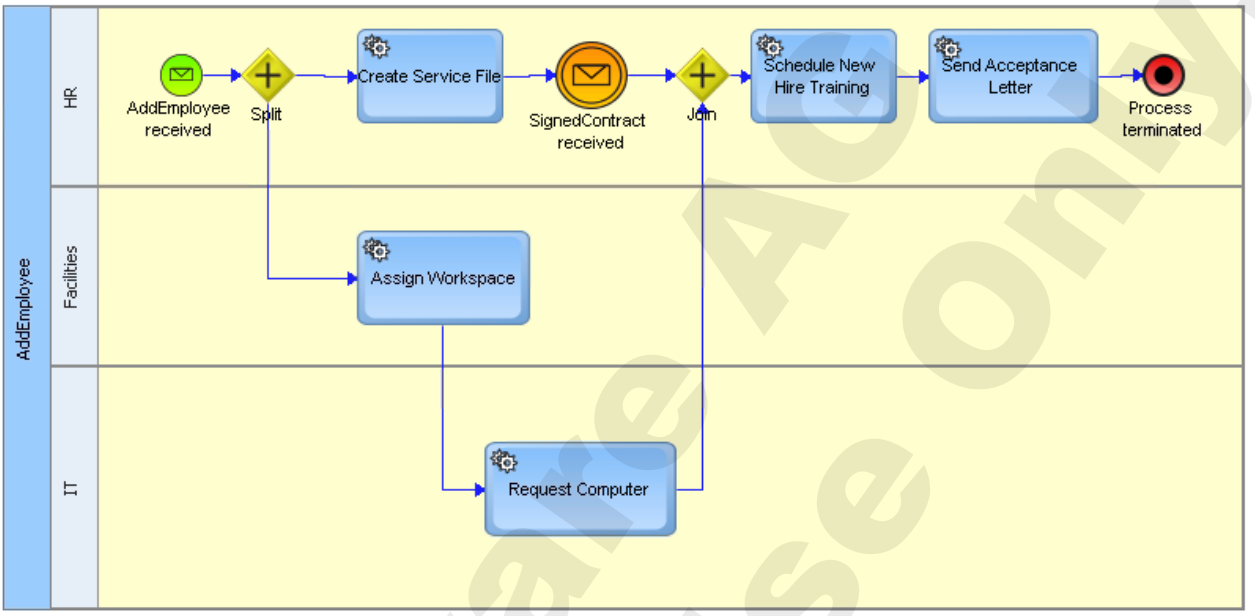
13. Right-click Task Activity **Create Service File** in the process editor and select **Edit Data Mapping**. You will get the (generated) flow service **Create_Service_File** opened in the flow service editor. Highlight the Invoke step **bpmTestDrive.services:CreateServiceFile** in the Tree tab and drop document reference **AddEmployee** from the output pipeline in the Pipeline view. Save your changes.



Software AG Internal Use Only!

14. *For extra credit*:
    Resize all Task Activities, Intermediate Event **SignedContract received**, and both Gateways. Moreover adjust the docking location of the incoming and outgoing transitions for Task Activity Request Computer.

    Finally your process should look like this:



Software AG Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 2:
JMS AND UNIVERSAL MESSAGING SUPPORT

Objectives

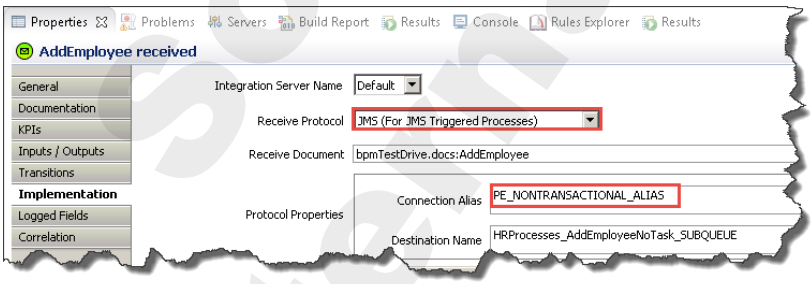
In this exercise you will execute a JMS-based process model (the one you created in exercise 1). In a first test you will use will use the preconfigured JMS settings that are using Universal Messaging as a JMS provider. To test/execute your process, you will use an IS service that sends a JMS message to the Process Engine and use the ability of Designer to send a document via JMS to the Process Engine.
Finally you will customize your IS/PE environment to use a different JMS Provider (e.g. webMethods Broker). Then you will run your process again.

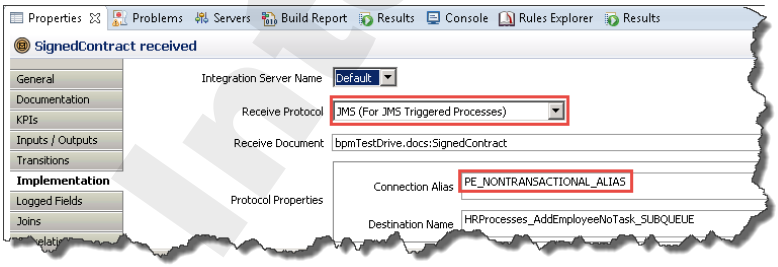
In this exercise, please replace:
<workshop_dir> by **C:\Training\602D-66E\BPM**

Steps

1. You will use your process model **AddEmployeeNoTask** created in exercise 1. As JMS is the default receive protocol in webMethods 9.5 and above, this process is configured to use JMS as receive protocol.
   Note: Only in the case you have skipped exercise 1, first import process project **HRProcesses.zip** provided at ***<workshop_dir>*\Ex1\Solution\Process Projects.**
   Also use the IS Administration Console to load IS package **HRProcesses.zip** provided at ***<workshop_dir>*\Ex1\Solution\IS Packages** and enable IS package **BPMTestDrive.**

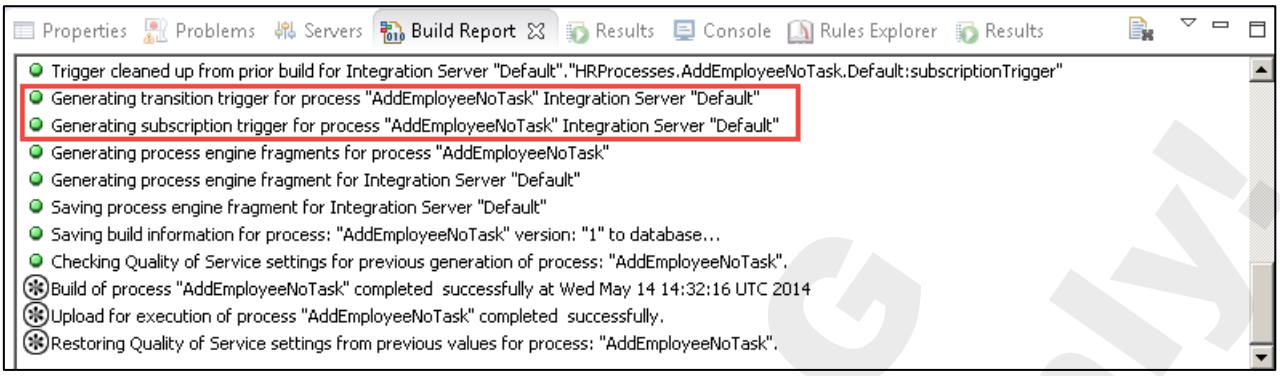
2. Open process model **AddEmployeeNoTask** as contained in process project **HRProcesses** in the process editor. Use the **Implementation** tab in the **Properties** view to double-check that both Events **AddEmployee received** and **SignedContract received** are configured to use **JMS** as Receive protocol and are using **PE_NONTRANSACTIONAL_ALIAS** as JMS Connection Alias.



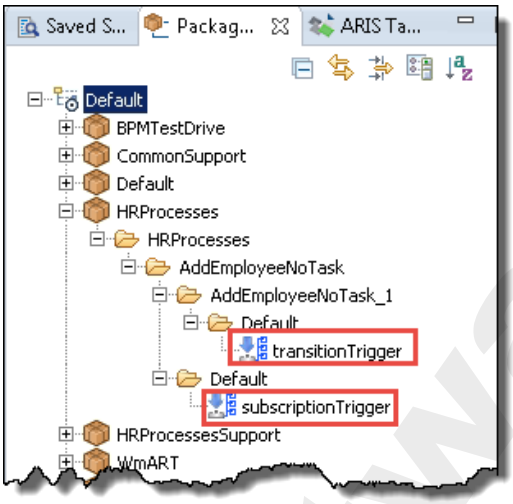


Software AG Internal Use only!

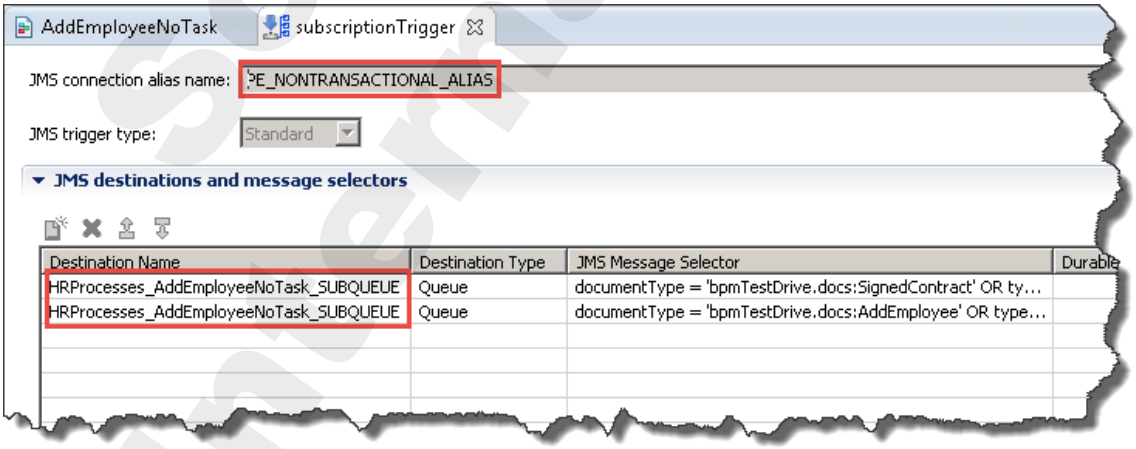
3. Build and upload your process model. When get prompted, allow enabling the process model for execution. As a result, two new JMS Triggers should have been generated:



4. Refresh the Package Navigator view in Designer. Have a look for both JMS Triggers (subscription, transition) been generated in Package **HRProcesses**:



5. Open the subscription trigger from the Package Navigator view. Check the used JMS Connection Alias name (**PE_NONTRANSACTIONAL_ALIAS)** and the (generated) Destination Name **HRProcesses_AddEmployeeNoTask_SUBQUEUE** of type **Queue**. Ensure the name reflects your process project and process model name.

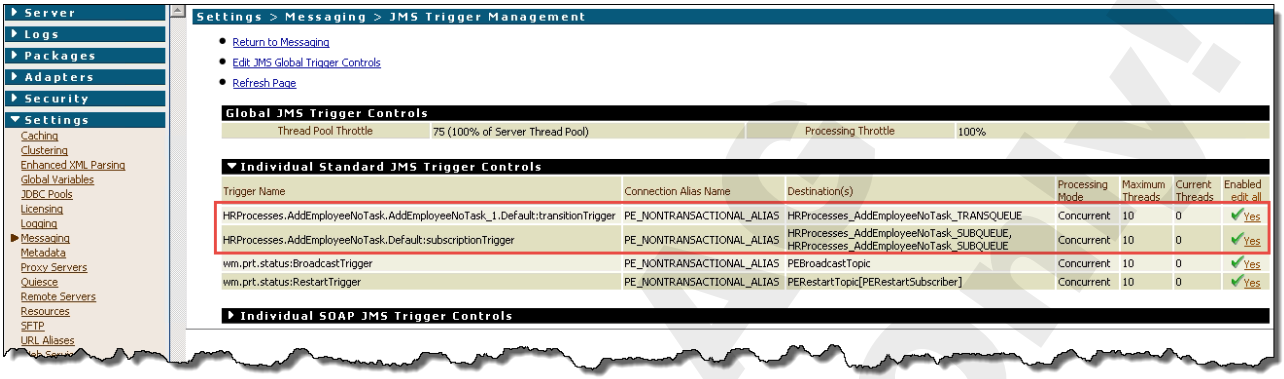


Software AG
Internal Use Only!

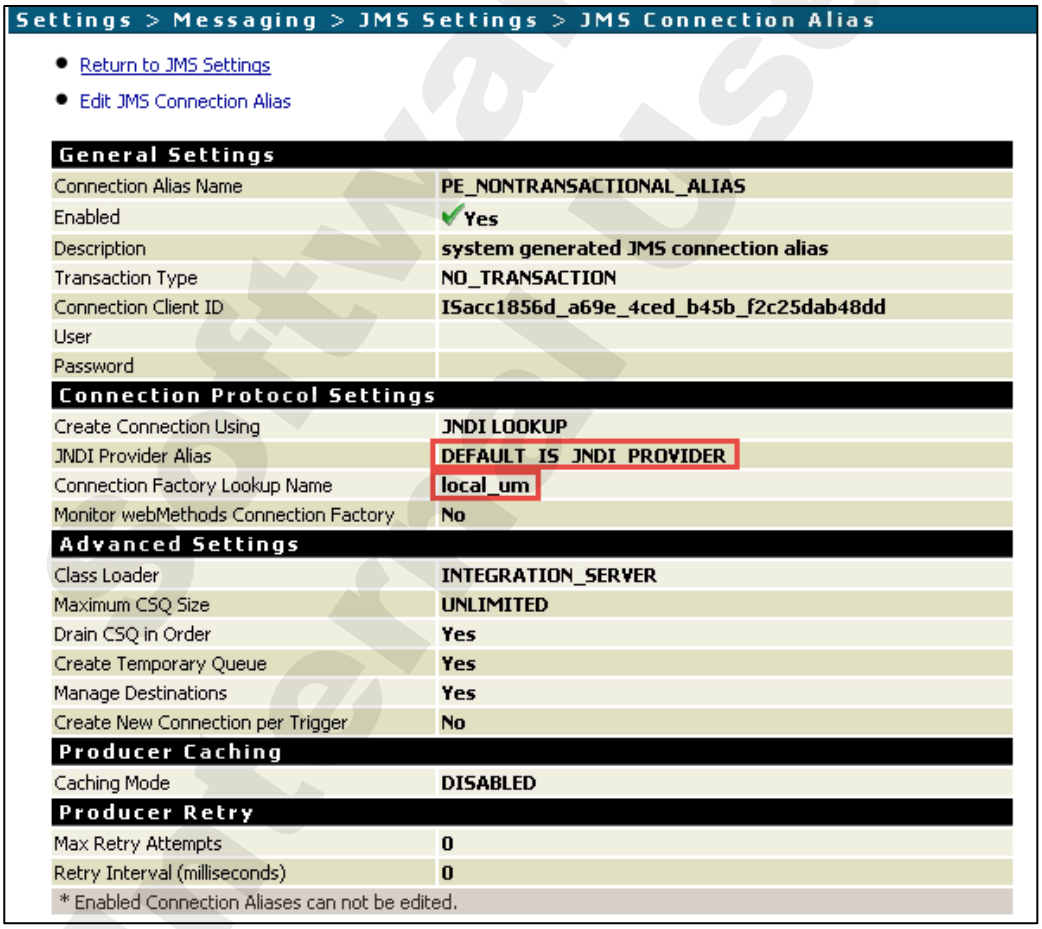
© 2000 – 2014 by Software AG

6. Open a browser tab and login to IS Administration console (http://localhost:5555) as **Administrator | manage**.

   a. Navigate to **Settings** > **Messaging** > **JMS Trigger Management.** Ensure both JMS Triggers are available in IS and enabled:

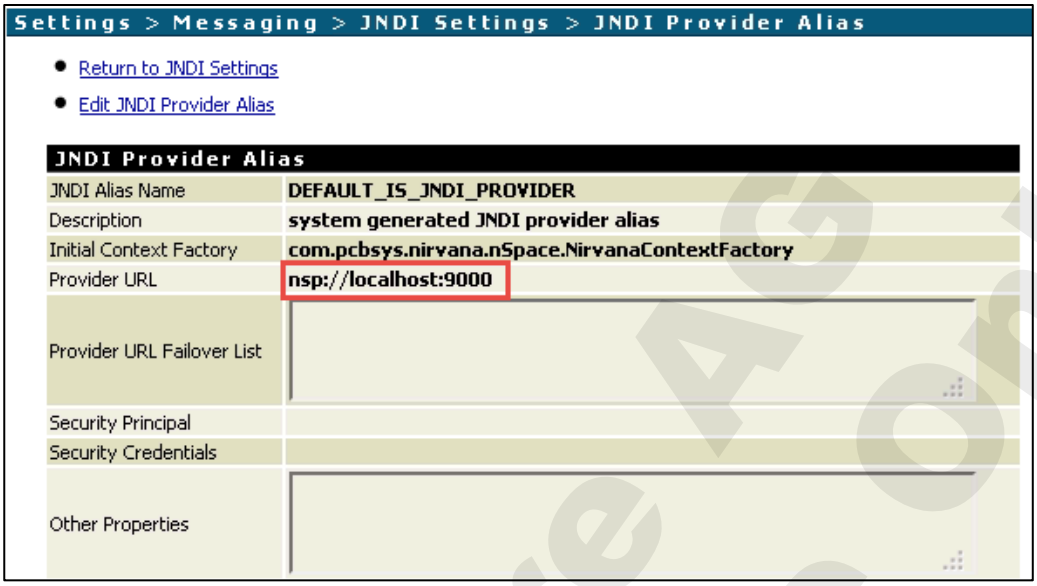


   b. In the IS Administration console, navigate to **Settings** > **Messaging** > **JMS Settings**. Inspect the default JMS Connection alias **PE_NONTRANSACTIONAL_ALIAS** to investigate the used JNDI Provider Alias and Connection Factory Lookup Name in the JMS provider.

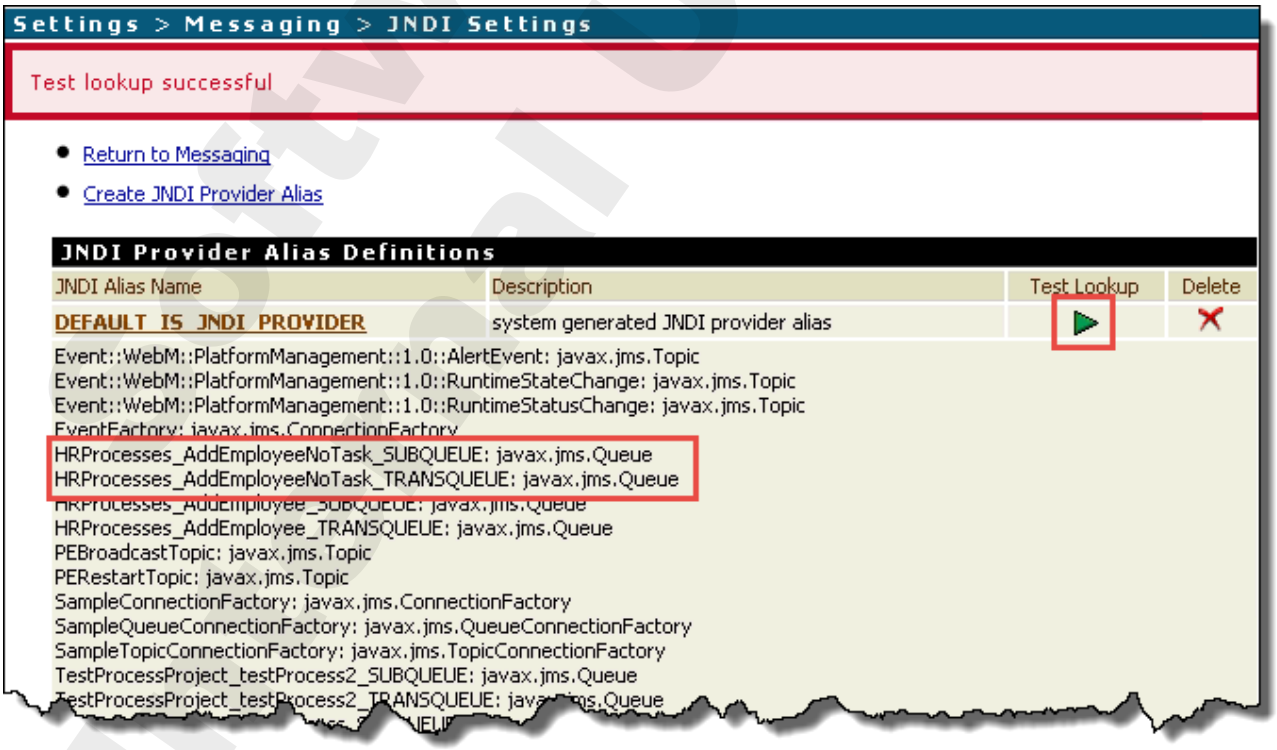


Internal Use Only! Software AG

c. In the IS Administration console, navigate to **Settings** > **Messaging** > **JNDI Settings.** Inspect the used JNDI Provider Alias **DEFAULT_IS_JNDI_PROVIDER** for the used JNDI Provider URL (Universal Massaging realm URL):



d. Test the JNDI Provider Alias **DEFAULT_IS_JNDI_PROVIDER**. Among others results you should see JNDI entries in Universal Messaging for subscription and transition queues generated for your process model during build and upload:



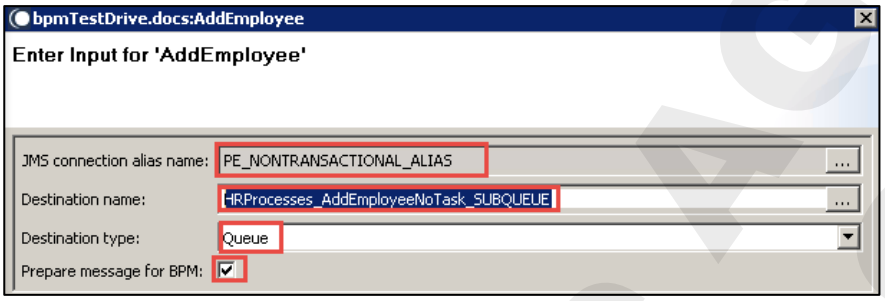
Internal Use Only! Software AG

© 2000 – 2014 by Software AG

7. Switch back to Designer. Test your process **AddEmployeeNoTask** by simply sending a JMS message containing the required documents to the Process Engine. To do so:

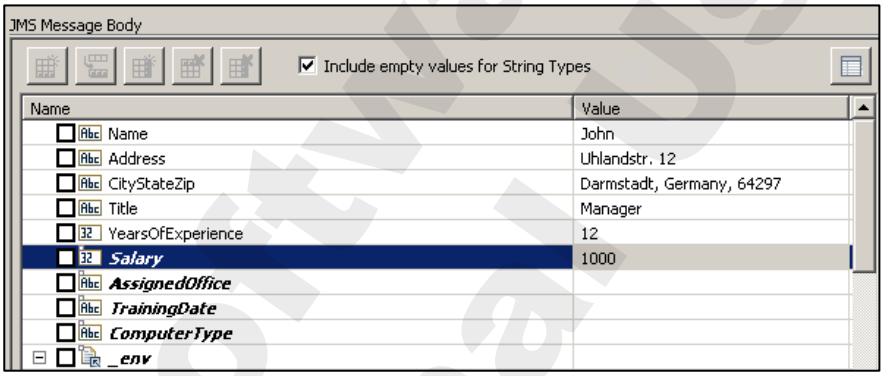
   a. Right-click **bpmTestDrive.docs:AddEmployee,** select **Run As > Publish as JMS Message** to send it as a JMS message.

   b. In the upper portion of the input panel, browse for JMS Connection Alias **PE_NONTRANSACTIONAL_ALIAS**, Destination name **HRProcesses_AddEmployeeNoTask_SUBQUEUE** and select Destination type **Queue**. To send the request to the Process Engine, check **Prepare for BPM**.



   c. Leave all **JMS Message Header and Properties** empty.

   d. In the portion titled JMS Message Body, provide some meaningful input, especially choose an unused value for field **name,** as this field value is used as correlation id and specify input for **YearsOfExperience** as well as **Salary**. As an easy alternative, load the provided input from **<*workshop_dir*>\Ex2\Resources\AddEmployee_Input.txt.**



   e. Click **OK** to send an instance of document type AddEmployee as a JMS message to your Process Engine.

   f. To provide input for the catching Intermediate Message Event Signed Contract received, right-click **bpmTestDrive.docs:SignedContract** from the Package Navigator view. Select **Run As > Publish as JMS Message** to send it as a JMS message.

   g. In the upper portion of the input panel, provide the same input as above (step 7b)).

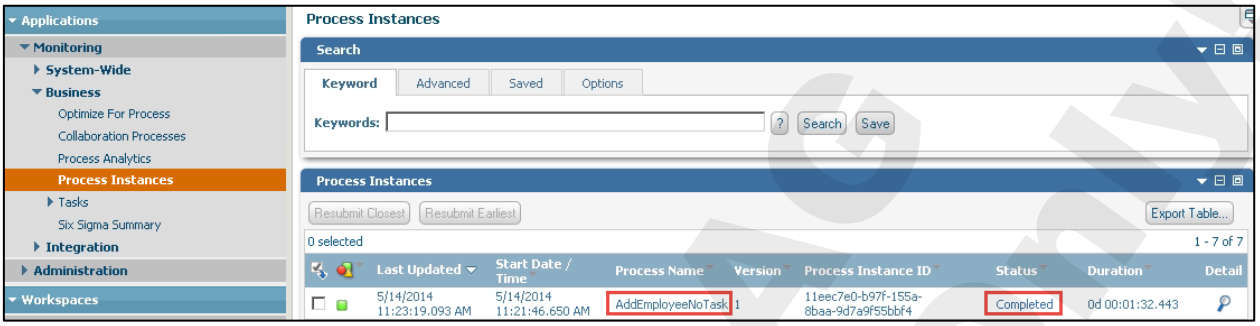
   h. Leave all **JMS Message Header and Properties** empty.

   i. In the portion titled JMS Message Body, provide the same value for field **name** as done before (step 7d) to enforce correlation. As an easy alternative, load the provided sample input from **<*workshop_dir*>\Ex2\Resources\ SignedContract_Input.txt.**

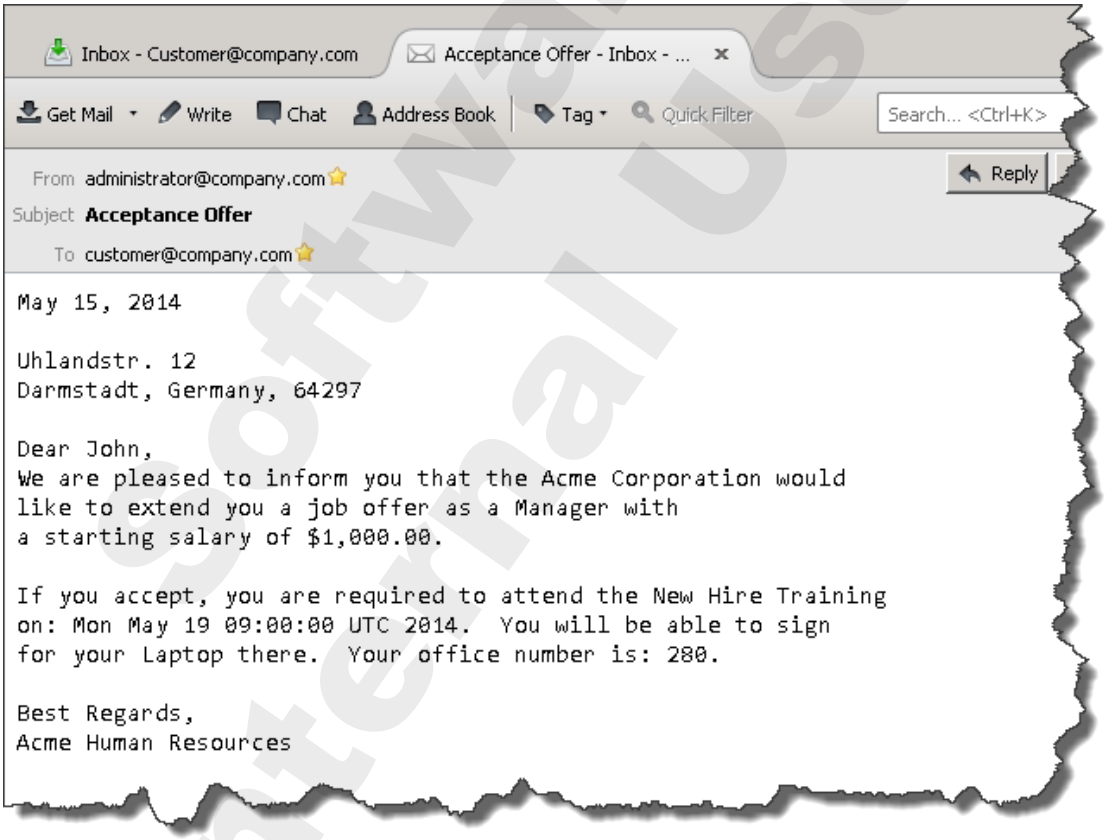
   j. Click **OK** to send an instance of document type SignedContract as a JMS message to your Process Engine.

Software AG Internal Use Only!

8. Use a browser tab (http://localhost:8585) to login to My webMethods as **Administrator | manage**. Drill down to **Applications** > **Monitoring** > **Business** > **Process Instances.** Ensure **Server** is selected to be **BPM only**. Hit **Search** in the upper Search portlet to refresh the process list in the lower Process Instances portlet. You should find a process instances of type **AddEmployeeNoTask** with status **Completed**:



9. *Optionally*: Start **Mozilla Thunderbird** from the Windows Start menu. Check for a new email for user account customer@company.com (password: **manage**). This email has been sent by Task Activity Send Acceptance Letter.



Software AG Internal Use Only!

© 2000 – 2014 by Software AG

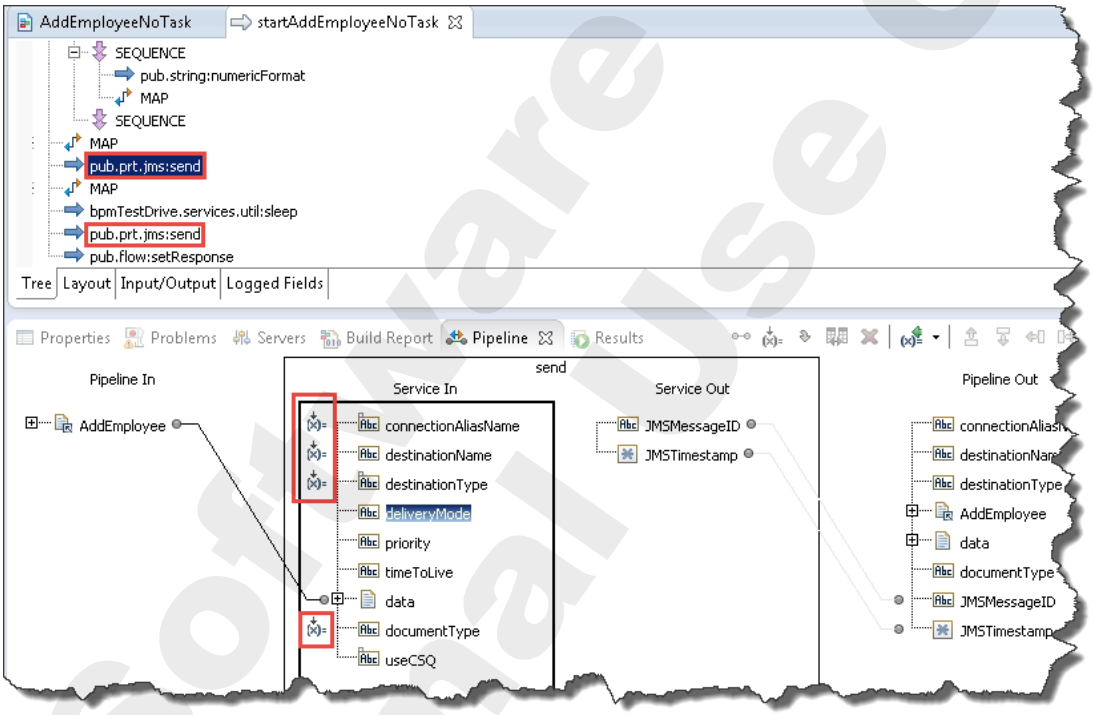
10. Switch back to Designer. Perform another test by running a prepared IS service which sends input for both process events via JMS by using the **pub.prt.jms.send** service. To do so:

    a. Switch to the Process Development perspective and open IS service **bpmTestDrive.services.util:startAddEmployeeNoTask** from the Package Navigator view.

    b. Highlight the first invocation of **pub.prt.jms.send** in the tree tab and check the service input assignments for

    - **connectionAliasName**
    - **destinationName**
    - **destinationType**
    - **data**
    - **documentType**

    in the Pipeline view. Adapt them, if necessary.



    c. Repeat the previous step for the second invocation of **pub.prt.jms.send** within service **bpmTestDrive.services.util:startAddEmployeeNoTask.**

    d. Save (if necessary) and run service **bpmTestDrive.services.util:startAddEmployeeNoTask** from the Package Navigator view as a Flow service. Provide some input data or load the provided sample input from ***<workshop_dir>*\Ex2\Resources\ startAddEmployeeNoTask_Input.txt.** Hit **OK.**

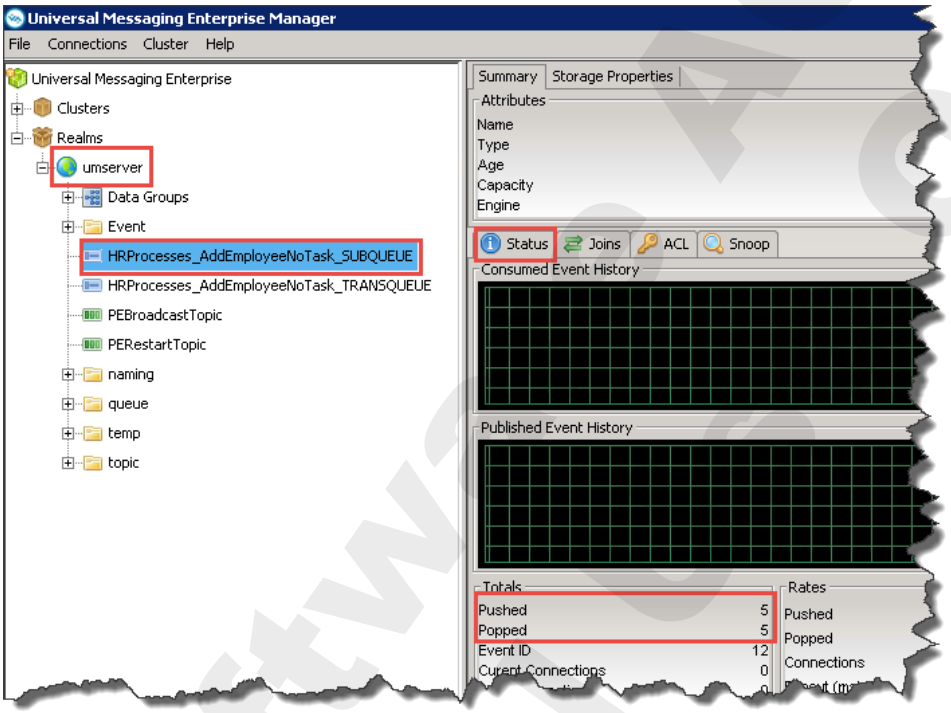
11. Reuse your browser tab (**http://localhost:8585**) connected to My webMethods as **Administrator | manage**. On the Business Processes page, hit **Search** in the upper Search portlet to refresh the process list in the lower Process Instances portlet. You should find another process instance of type **AddEmployeeNoTask** with status **Completed**.

Software AG Internal Use Only!

12. Start Universal Messaging Enterprise Manager for realm **umserver** (**Start** > **All Programs** > **Software AG** > **Administration** > **Universal Messaging Administration 9.x** > **umserver** > **Enterprise Manager**).

    a. In Enterprise Manager, use the left-hand navigation panel navigate to **Realms** > **umserver.** Expand the tree view and find the corresponding queues named **HRProcesses_AddEmployeeNoTask_SUBQUEUE** and **HRProcesses_AddEmployeeNoTask_TRANSQUEUE** in the view.

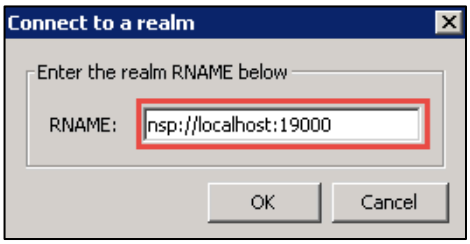
    b. Highlight queue **HRProcesses_AddEmployeeNoTask_SUBQUEUE** in the Navigator for realm **umserver** and visit the **Status** tab. Depending on your tests made before, you should see at two (or more) pushed events.



13. Let's setup a custom JMS Connection to another JMS Provider (e.g. Universal Messaging realm or webMethods Broker):

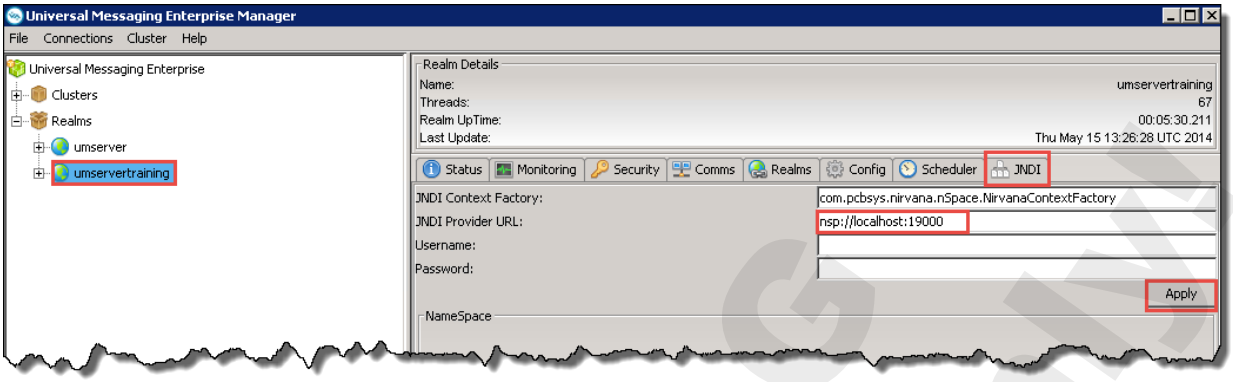
    a. Start another Universal Messaging realm (**Start** > **All Programs** > **Software AG** > **Start Servers** > **Start Universal Messaging Realm Server 9.x** > **Start umservertraining**)

    b. In Enterprise Manager, connect to the additional realm named **umservertraining** using the RNAME URL: **nsp://localhost:19000**

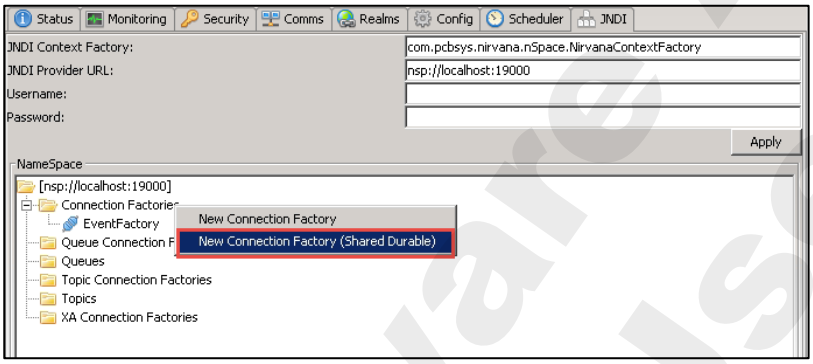


© 2000 – 2014 by Software AG

Select realm **umservertraining** in the left-hand navigation panel, select tab **JNDI** and modify the JNDI Provider URL to the value **nsp://localhost:19000.** Hit **Apply.**

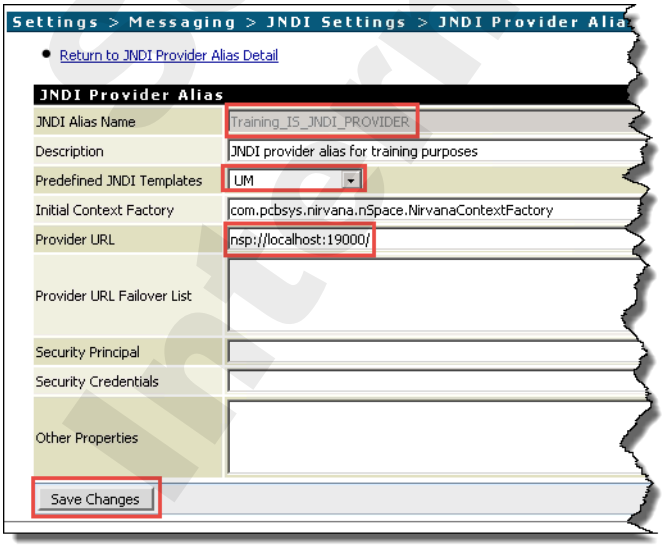


In the retrieved JNDI namespace, right-click Connection Factories to add a shared durable Connection factory named **training_um**.



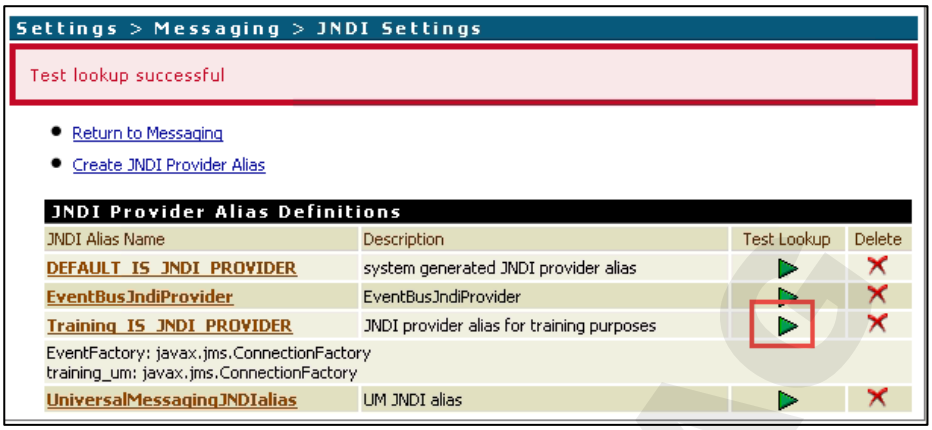
Note: The prepared realm **umservertraining** already contains topics named **PEBroadcastTopic** and **PERestartTopic** and a Connection Factory named **EventFactory** which are required by the Process Engine.

c. Use a browser tab to login to IS Administration console (**http://localhost:5555**) as **Administrator** | **manage**. In the IS Administration console, navigate to **Settings** > **Messaging** > **JNDI Settings.** Create a new JNDI Provider Alias named **TRAINING_IS_JNDI_PROVIDER** for Universal Messaging access (**UM**), specify the JNDI Provider URL **nsp://localhost:19000** and a meaningful description. **Save** it.



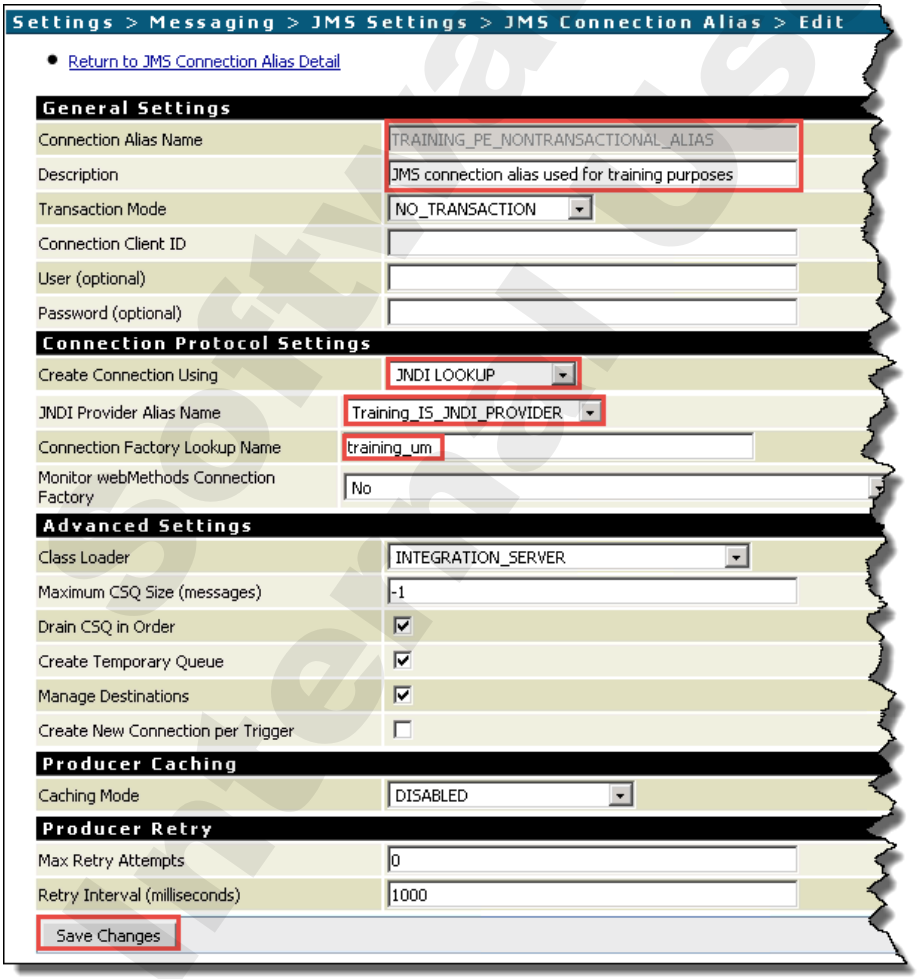
Software AG Internal Use Only

Navigate back to the page **Settings** > **Messaging** > **JNDI Settings** and test your JNDI Provider Alias **TRAINING_IS_JNDI_PROVIDER**:



You should see a message “Test lookup successful” and at least your connection factory lookup name defined in realm umservertraining being returned.

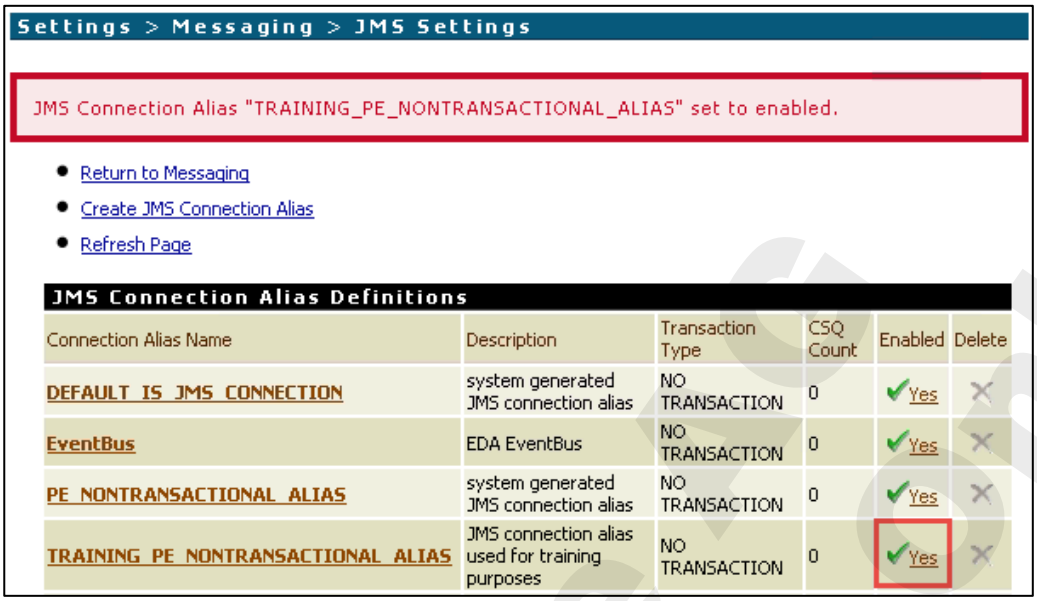
d. In IS Administration console, navigate to **Settings** > **Messaging** > **JMS Settings.** Create a JMS Connection Alias named **TRAINING_PE_NONTRANSACTIONAL_ALIAS.** Specify a meaningful description and select **JNDI LOOKUP** as connection type. Choose your custom JNDI Provider Alias **TRAINING_IS_JNDI_**PROVIDER and specify **training_um** as Connection Factory Lookup Name. **Save** your definition.



Internal Use Only! Software AG

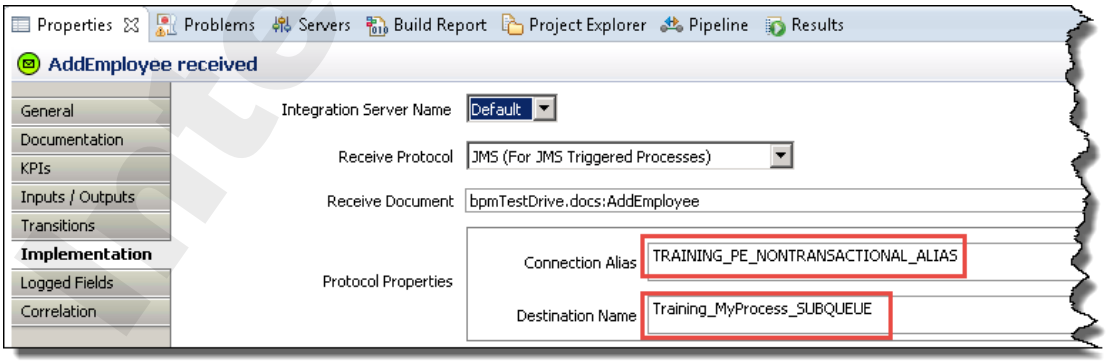
© 2000 – 2014 by Software AG

Return to the JMS Settings page and enable your JMS Connection Alias **TRAINING_PE_NONTRANSACTIONAL_ALIAS**:



You should see a message "JMS Connection Alias "TRAINING_PE_NONTRANSACTIONAL_ALIAS" set to enabled."

14. In Designer, modify your process to use your custom JMS connectivity:
    a. In the Process Development perspective, additionally open the **Project Explorer** view.
    b. In the Project Explorer, right-click **AddEmployeeNoTask.process** within project **HRProcesses**, copy and paste it to the same project as **AddEmployeeNoTaskCustomJMS.process**.
    c. Open **AddEmployeeNoTaskCustomJMS** from the Solutions view.
    d. Click into the whitespace in the process editor to set the focus to the entire process model. On the **General** tab of the Properties view change the process name to **AddEmployeeNoTaskCustomJMS.**
    e. Select step **AddEmployee received** and adjust its JMS properties on the Implementation tab of the Properties view like this:
        - Connection Alias: browse for **TRAINING_PE_NONTRANSACTIONAL_ALIAS**
        - Destination Name: type in: **Training_MyProcess_SUBQUEUE** (name doesn't exist yet)



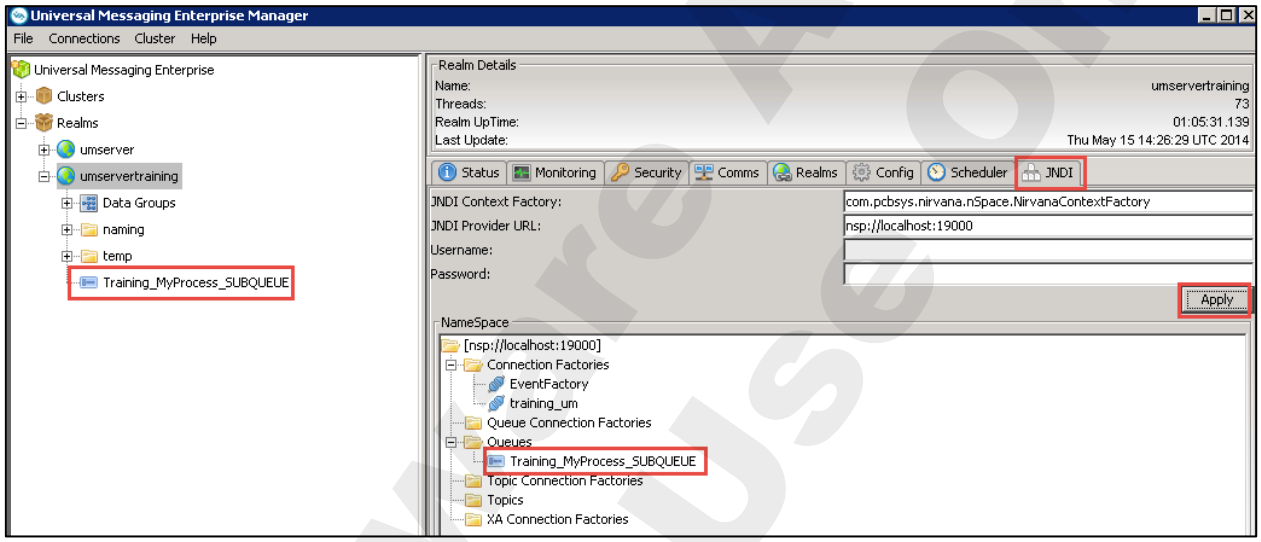
Software AG Internal Use only!

f. Adjust the JMS properties of step **SignedContract received** in the same way.

g. Save, build and upload your process model **AddEmployeeNoTaskSubscription**. When get prompted, allow enabling the process model for execution.

h. Right-click Task Activity **Create Service File** in the process editor and select **Edit Data Mapping**. You will get the (generated) flow service **Create_Service_File** opened in the flow service editor. Highlight the Invoke step **bpmTestDrive.services:CreateServiceFile** in the Tree tab. Ensure document reference **AddEmployee** is dropped from the output pipeline in the Pipeline view. Save your changes.

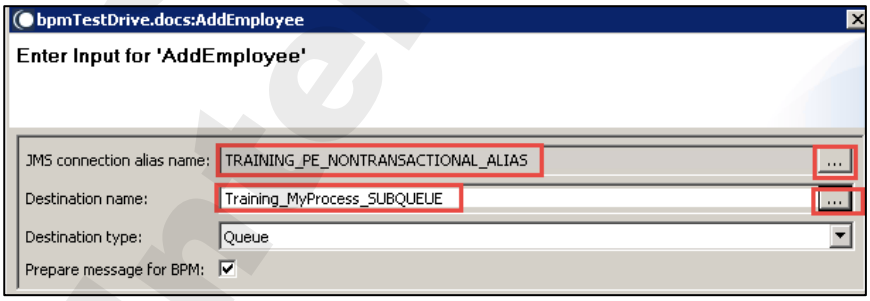
15. *Optional*: Use Enterprise Manager of Universal Messaging to verify a new JMS queue named **Training_MyProcess_SUBQUEUE** has been created at realm **unservertraining**. Hit **Apply** (twice) to refresh the JNDI settings view:



16. Test your process **AddEmployeeNoTaskCustomJMS** in Designer using your custom JMS settings:

a. Right-click **bpmTestDrive.docs:AddEmployee,** select **Run As > Publish as JMS Message** to send it as a JMS message.

b. In the upper portion of the input panel, browse for JMS Connection Alias **TRAINING_PE_NONTRANSACTIONAL_ALIAS**, Destination name **Training_MyProcess_SUBQUEUE** and select Destination type **Queue**. To send the request to the Process Engine, check **Prepare for BPM**.



c. Leave all **JMS Message Header and Properties** empty.

Internal Use only! Software AG

 © 2000 – 2014 by Software AG

d. In the portion titled JMS Message Body, reload the provided input from ***<workshop_dir>*\Ex2\Resources\AddEmployee_Input.txt.** Because PE caches used correlation values, provide a different value for field **Name** as used before. Click **OK**.

e. To provide input for the catching Intermediate Message Event Signed Contract received, right-click **bpmTestDrive.docs:SignedContract** from the Package Navigator view. Select **Run As > Publish as JMS Message**.

f. In the upper portion of the input panel, browse for the same input as above (step 16b)).

g. Leave all **JMS Message Header and Properties** empty.

h. In the portion titled JMS Message Body, provide the same value for field **Name** as specified above (step 16d) to enforce correlation. Click **OK.**

17. Use a browser tab (http://localhost:8585) to login to My webMethods as **Administrator | manage**. Drill down to **Applications** > **Monitoring** > **Business** > **Process Instances.** Hit **Search** in the upper Search portlet to refresh the process list in the lower Process Instances portlet. You should find a new process instances of type **AddEmployeeNoTaskCustomJMS** with status **Completed**:

18. *Housekeeping*: Stop Universal Messaging realm **umservertraining** (**Start** > **All Programs** > **Software AG** > **Stop Servers** > **Stop Universal Messaging Realm Server 9.x** > **Stop umservertraining**)

Software AG Internal Use Only!

This page is intentionally left blank

Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 3: DEBUGGER IMPROVEMENTS

Objectives

In this exercise you will use the new Process Debugger to debug a JMS-based process model (the one you created in exercise 1). To use the Process Debugger you have to create a Run Configuration first. Using the Debug Configuration, you can debug the process step by step, run it with breakpoints, and modify the process pipeline.
After changing the receive protocol of your process model from JMS to Subscription, you will reuse the same Debug Configuration to test your process again.

In this exercise, please replace:
<workshop_dir> by **C:\Training\602D-66E\BPM**

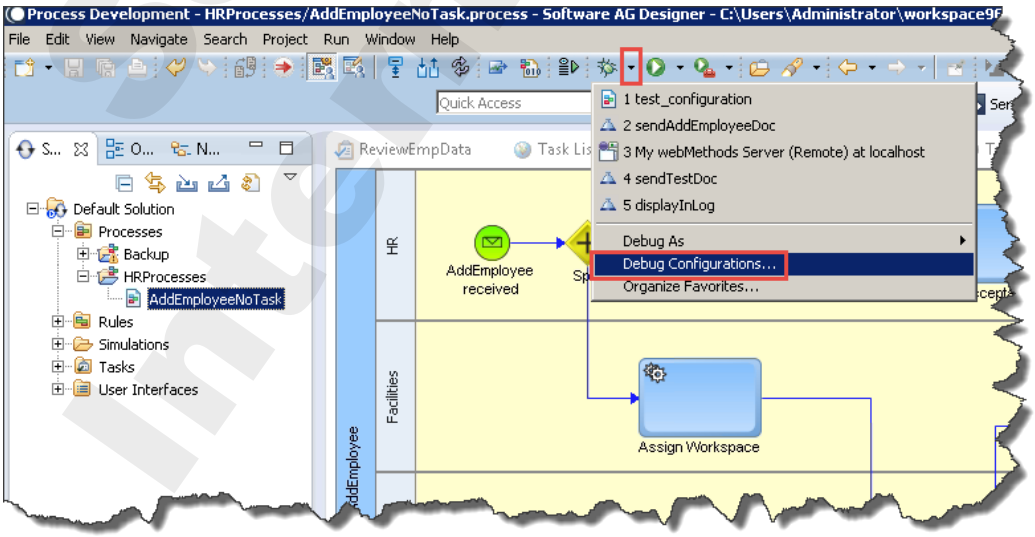
Steps

1. You will use your process model **AddEmployeeNoTask** created in exercise 1. This process is configured to use JMS as receive protocol (default).

   Note: Only in the case you have skipped exercise 1, first import process project **HRProcesses.zip** provided at ***<workshop_dir>*\Ex1\Solution\Process Projects.**
   Also use the IS Administration Console to load IS package **HRProcesses.zip** provided at ***<workshop_dir>*\Ex1\Solution\IS Packages** and enable IS package **BPMTestDrive.**
   Finally build and upload the process model **AddEmployeeNoTask** in Designer.

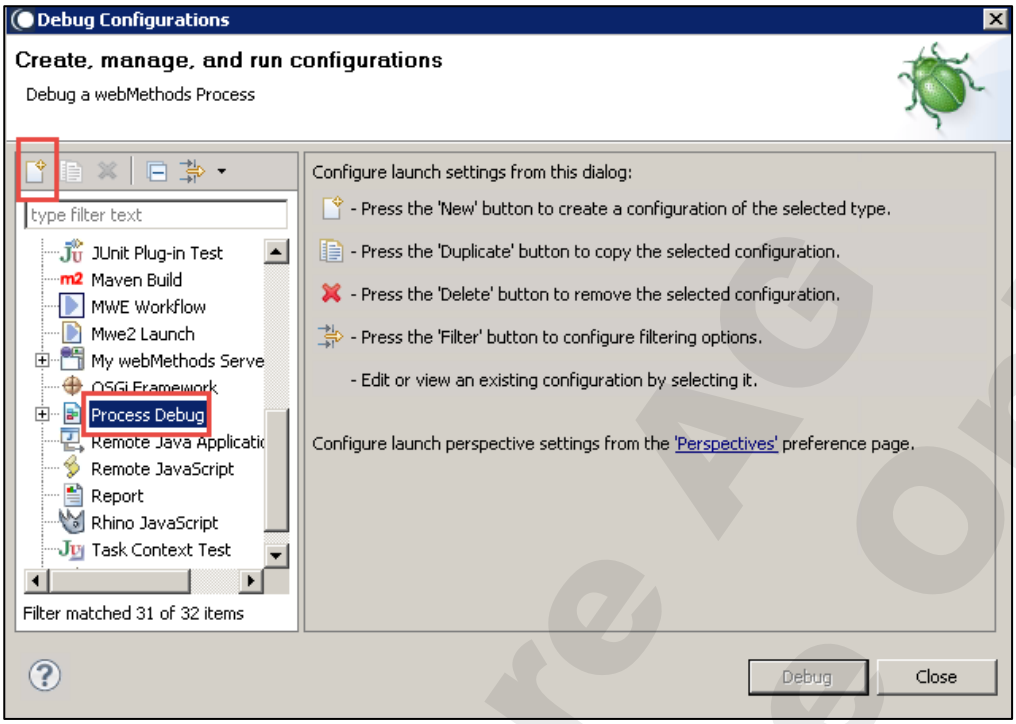
2. Debug your JMS-based process in Designer using the new Process Debugger:

   a. Before you can debug a process model for the very first time, you have to create a Debug Configuration in your Designer workspace. To do so, click the triangle next to the Debug icon having the process model loaded in the process editor. Select **Debug Configurations…** .

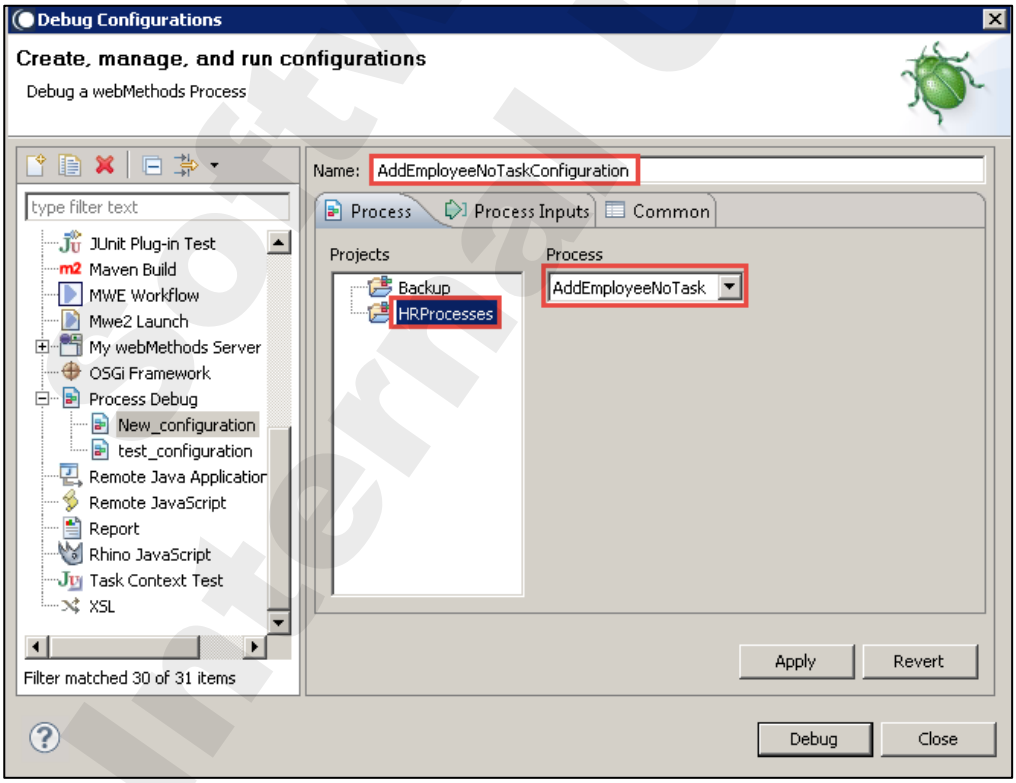


Software AG Internal Use Only!

b. Select asset type **Process Debug** and hit the **New** button.



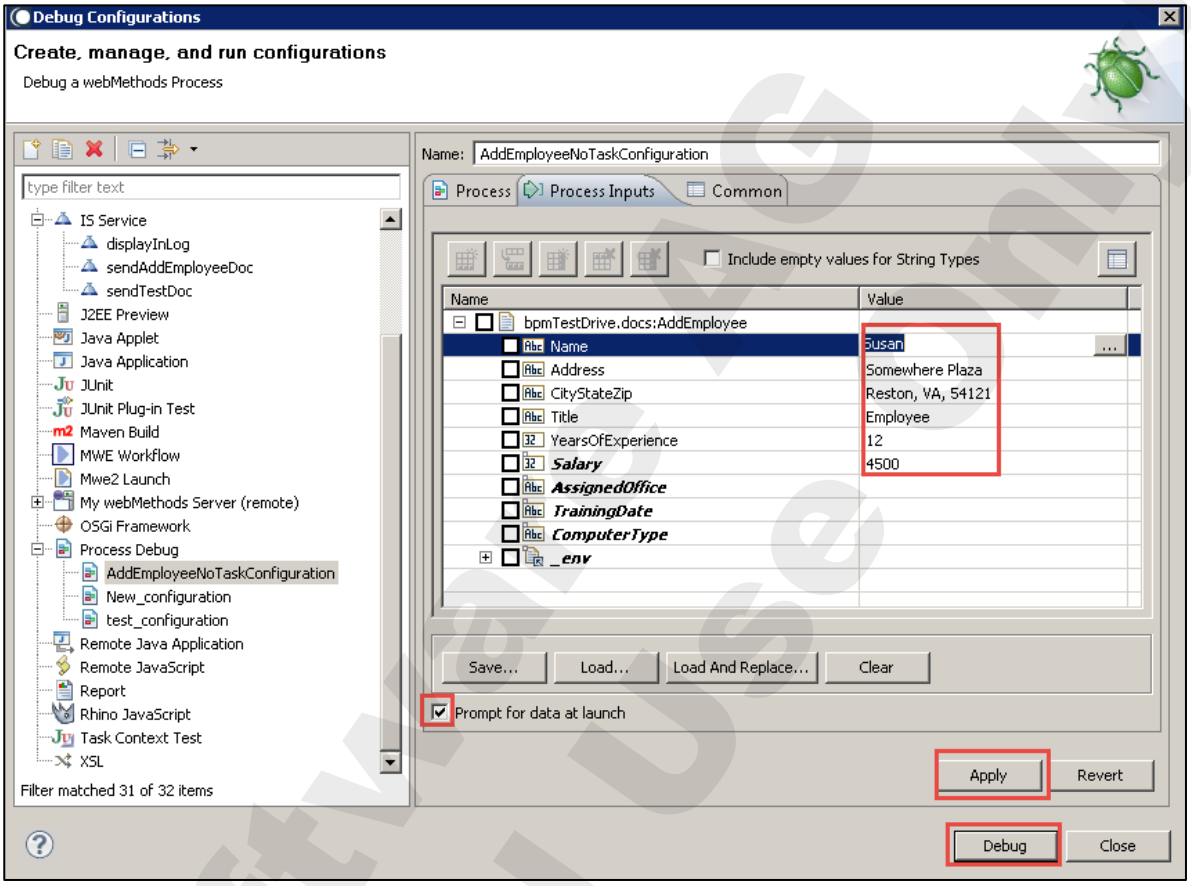
c. On the **Process** tab of your new Run Configuration, specify Run Configuration name **AddEmployeeNoTaskConfiguration** and select the process model **AddEmployeeNoTask** within the process project **HRProcesses**.



Internal Use Only! Software AG

© 2000 – 2014 by Software AG

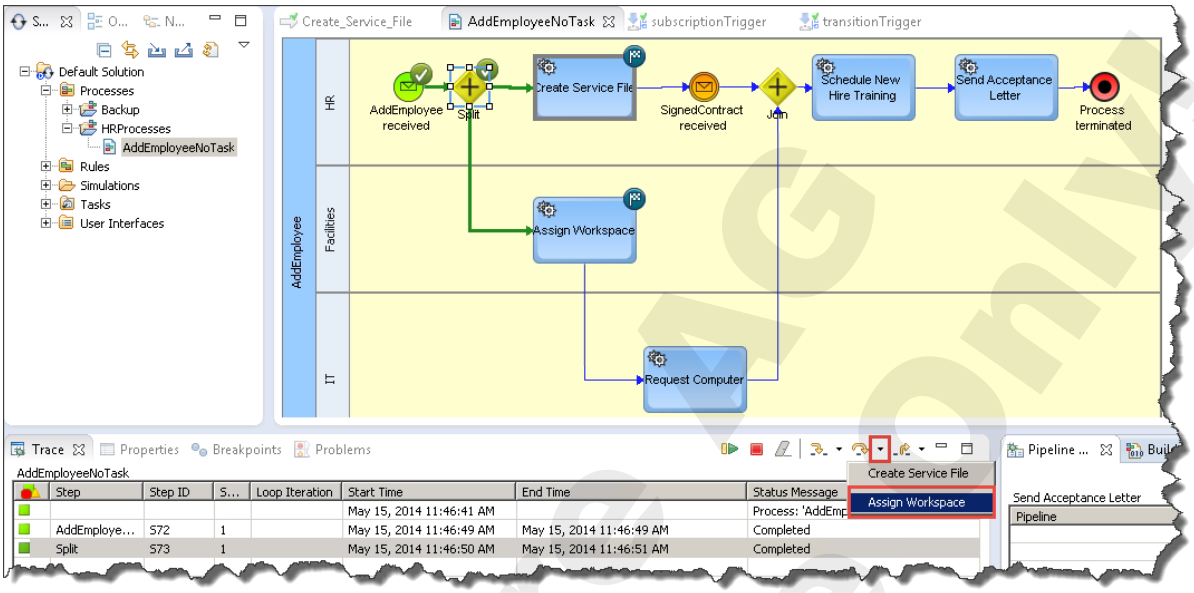
d. On the **Process Inputs** tab, provide input data for the Start Message Event and check to get prompted to type/alter those data when debugging starts.
Instead of typing you can load input data from
***<workshop_dir>*\Ex3\Resources\AddEmployee_ConfigInput.txt**. Click **Apply** to save the Debug Configuration. Then click **Debug** to start process debugging.



e. You should get prompted to switch to the Process Debug perspective. Click **Yes** to switch. Because of the settings you made in the Debug Configuration you get prompted for the input data having your data specified as in the Run Configuration preloaded. Sample input can also be loaded from
***<workshop_dir>*\Ex3\Resources\AddEmployee_DebugInput.txt**.
Modify them, if desired, and click **OK**.
Important: If you have already performed exercise 2 a few minutes ago, ensure you use a different value for **Name** as used before. Hit **OK**.

Software AG Internal Use Only!

f. Use the **Step Over (F6)** command of the **Trace** view to debug your process step by step. After execution of the Split Gateway you should be able to decide which of the parallel steps should be executed next.



Execute the steps in the sequence **Assign Workspace** > **Create Service File** > **Request Computer.** Mark step **Request Computer** in the Trace view to inspect its outgoing business and process data in the Pipeline Data view.

g. To execute eligible step **SignedContract received**, we have to send an instance of document type SignedContract to the Process Engine. To do so:

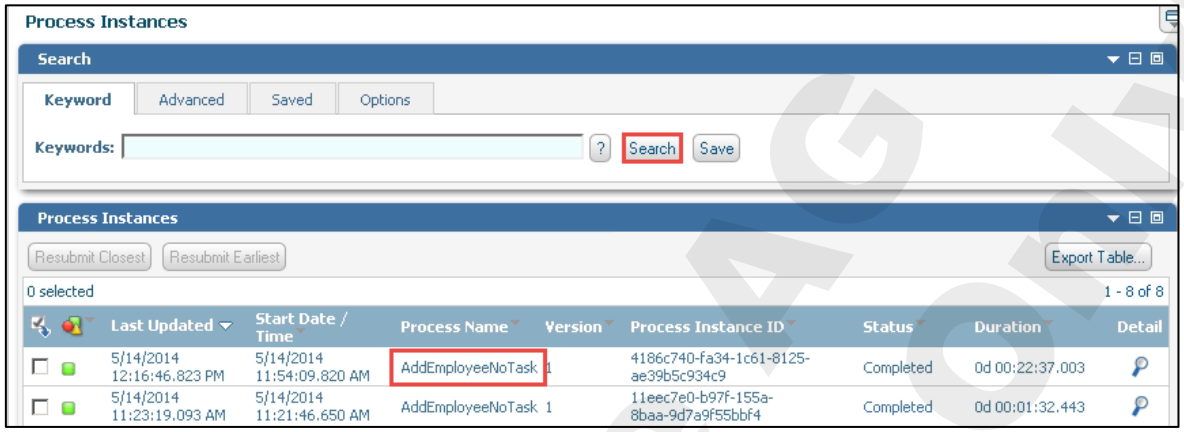
- First add the Package Navigator view to your Process Debug perspective (**Window** > **Show View** > **Other…**, type as filter **Package N**, select **Package Navigator** and click **OK**).
- In the Package Navigator view, right-click **bpmTestDrive.docs:SignedContract** and select **Run As > Publish as JMS Message** to send it as a JMS message.
- In the upper portion of the input panel, browse for JMS Connection Alias **PE_NONTRANSACTIONAL_ALIAS**, Destination name **HRProcesses_AddEmployeeNoTask_SUBQUEUE** and select Destination type **Queue**. To send the request to the Process Engine, check **Prepare for BPM**.
- Leave all JMS Message Header and Properties empty.
- In the portion titled JMS Message Body, provide the <u>same</u> value for field **Name** as specified in step 2e to enforce correlation.
- Click **OK** to send an instance of document type SignedContract as a JMS message to your Process Engine.

h. Execute all remaining steps step by step until your process completes successfully.

Software AG Internal Use Only

© 2000 – 2014 by Software AG

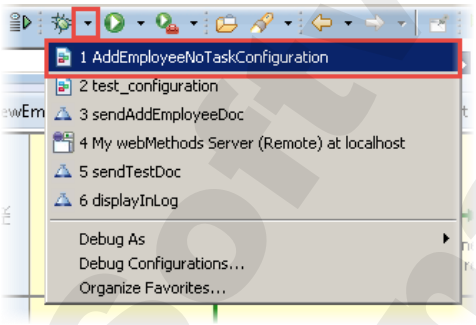
3. Because Process Debugger executed your process in the Process Engine, monitoring information is available. Use a browser tab to login to My webMethods (http://localhost:8585) as **Administrator** | **manage**. Drill down to **Applications** > **Monitoring** > **Business** > **Process Instances.** Ensure **Server** is selected to be **BPM only**. Hit **Search** in the upper Search portlet to refresh the process list in the lower Process Instances portlet. Look for your debugged process instance:



4. *Optionally*: Use Mozilla Thunderbird to look for a new confirmation mail sent to Customer@company.com.

5. Perform a second debug session:

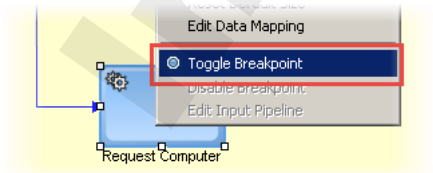
   a. Select your Debug Configuration **AddEmployeeNoTaskConfiguration** from the drop down next to the Debug icon.



   b. If not already in, accept to switch to the Process Debug perspective.

   c. Provide input for AddEmployee or reload input data from ***<workshop_dir>*\Ex3\Resources\AddEmployee_DebugInput.txt**. *Important*: Ensure you use a different value for **Name** as used before. Hit **OK**.

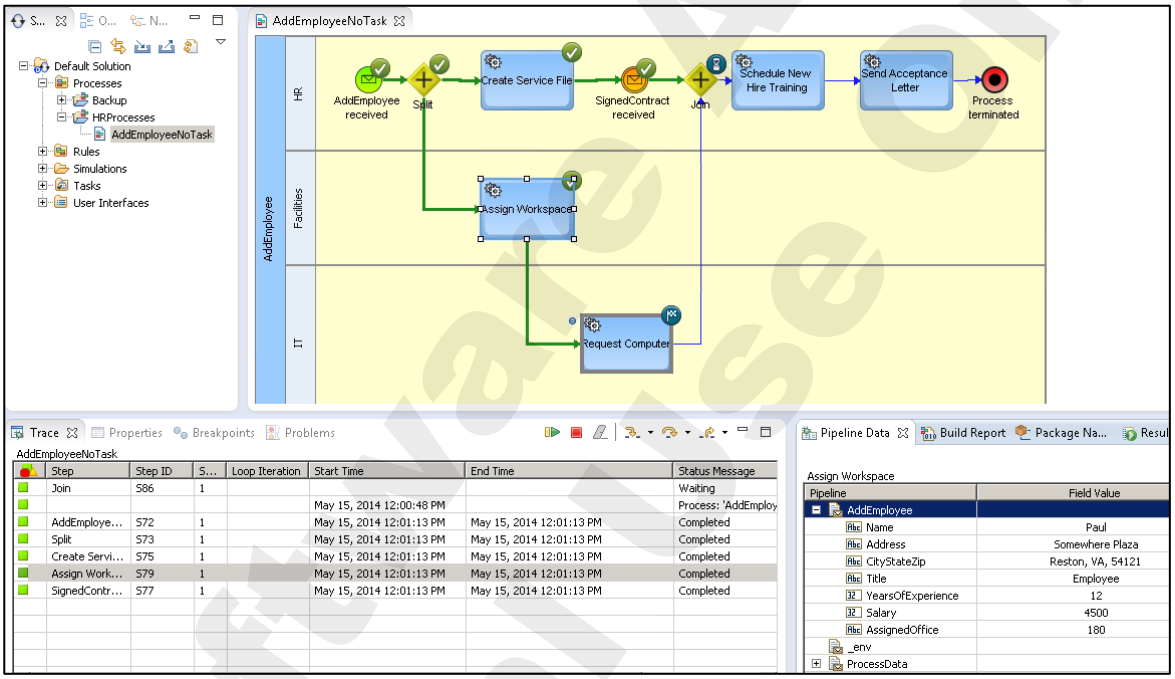
   d. Right-click Service Task Activity **Request Computer** in the process editor and select **Toggle Breakpoint** to set a breakpoint.



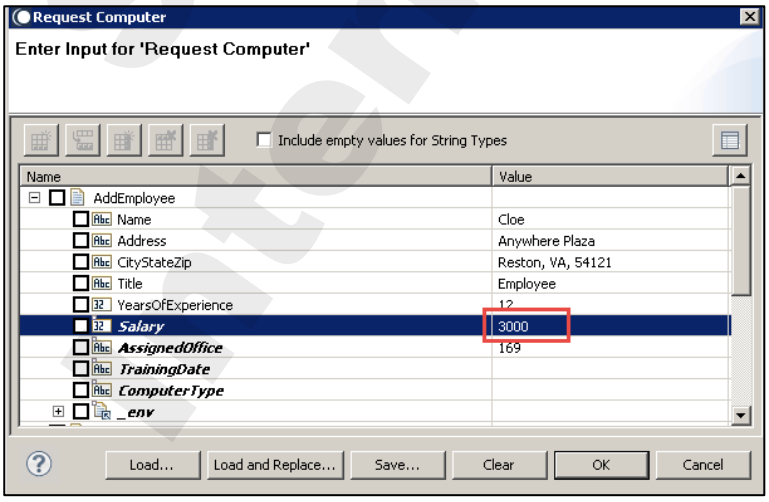
Software AG Internal Use Only!

e. As done before, use the Package Navigator view, right-click **bpmTestDrive.docs:SignedContract,** and select **Run As > Publish as JMS Message** to send another document instance for the Intermediate Message Event. On the input panel, provide input as above and ensure you use the same value for field **Name** as provided in step 5c to allow correlation. As a result, Intermediate Message Event **SignedContract received** should become active and waiting.

f. Use the Run button in the Trace view to execute all steps of the process instance. Inspect the Trace view to see how Debugger simulates parallelism.



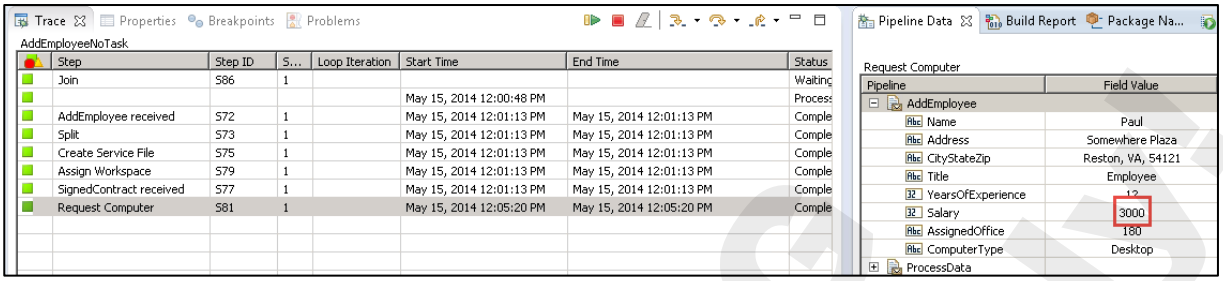
g. Because of your breakpoint, execution is paused at step Request Computer. Right-click step **Request Computer** in the Design canvas and select **Edit Input Pipeline**. Modify the value of **AddEmployee/Salary** in the Pipeline Data view and hit **OK**.



Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

h. In the Trace view, use command **Step Over (F6)** to execute step Request Computer. Mark step **Request Computer** in the Trace and inspect its output data in the Pipeline Data view. You should see your changed salary value.



i. Execute all remaining steps step by step until your process completes successfully.

j. Right-click step **Request Computer** in the process editor. Select **Toggle Breakpoint** again to remove its breakpoint.

6. Adjust your process model **AddEmployeeNoTask** to use Subscription instead of JMS as receive protocol:

   a. Switch to the Process Development perspective.

   b. Select step **AddEmployee received** of process model **AddEmployeeNoTask** in the process editor. In its Properties view, change the **Receive** protocol on the Implementation tab to **Subscription**.

   c. Select step **SignedContract received** in the process editor. In its Properties view, also change the **Receive** protocol on the Implementation tab to **Subscription**.

7. Save, build and upload your process model again. When get prompted, allow enabling the process model for execution.

8. Perform a last debug session using the same Debug Configuration:

   a. Select your Debug Configuration **AddEmployeeNoTaskConfiguration** from the drop down next to the Debug icon.

   b. If not already in, accept to switch to the Process Debug perspective.

   c. Provide input for AddEmployee or load input data from ***<workshop_dir>*\Ex3\Resources\AddEmployee_DebugInput.txt**. *Important*: Ensure you use again a different value for **Name** as used before. Hit **OK**.

   d. Use the Package Navigator view and right-click **bpmTestDrive.docs:SignedContract.** This time select **Run As > Publishable Document** to send a document instance via Subscription for the Intermediate Message Event. On the input panel, provide the same value for field **Name** as provided in step 8c to allow correlation. As a result, Intermediate Message Event **SignedContract** received should become active and waiting.

   e. Use the Run button in the Trace view to execute all steps of the process instance.

   Note: This should prove you can use a Debug Configuration to debug a process independent of the used receive protocol.

Software AG Internal Use Only

This page is intentionally left blank

Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 4: BPMN SUBPROCESSES

Objectives

In this exercise you will work on a process project that contains several process models which use the new BPMN subprocess. You will modify, execute and debug them to get to know the behavior of interrupting Boundary Events, parallel execution and looping.
Project process and a supporting IS package are provided and have to be imported in advance.

In this exercise, please replace:
<workshop_dir> by **C:\Training\602D-66E\BPM**

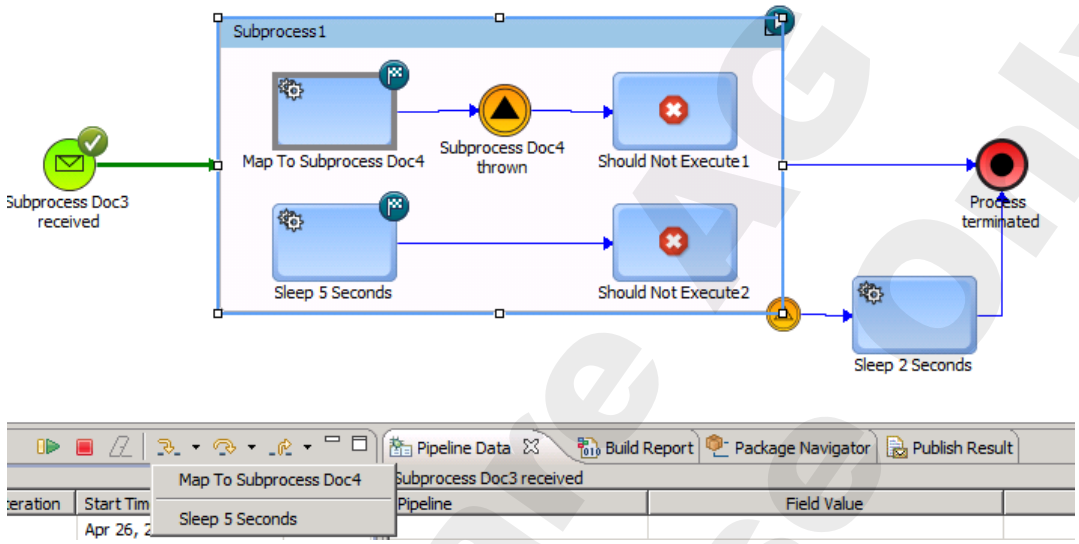
Steps

1. In a browser tab, login to the IS Administration Console (http://localhost:5555) as **Administrator** | **manage.** Navigate to **Packages** > **Management** to enable the already loaded IS package **SubprocessesSupport**.
2. Start Designer using workspace **C:\Users\Administrator\workspace96**.
   a. Switch to the **Process Development** perspective.
   b. Use **File** > **Import** > **General** > **Existing Projects into Workspace** to import a process project from the archive file ***<workshop_dir>*\Ex4\Resources\Process Projects\Subprocesses.zip** into your workspace.
3. Refresh the Package Navigator view and push (**Sync Document Type**) the IS document types **subprocessesSupport.docs:SubprocessDoc**<*n*> (*n*=1..3, 5..6) to the webMethods Messaging Provider (currently: webMethods Broker).
4. Test process model **SubprocessWithParallelism** in Designer:
   a. Open process model **SubprocessWithParallelism** contained in the process project **Subprocesses**.
   b. Build and upload the process model. When get prompted, allow enabling the process model for execution. Ignore warnings in the Build Report view.
   c. Create a Debug Configuration in your Designer workspace. To do so, click the triangle next to the Debug icon. Select **Debug Configurations…** . Select asset type **Process Debug** and hit the **New** button.
   On the **Process** tab of your new Run Configuration, specify Run Configuration name **SubprocessWithParallelismConfiguration** and select the process model **SubprocessWithParallelism** within the process project **Subprocesses**.
   On the **Process Inputs** tab, load some input data for the Start Message Event from ***<workshop_dir>*\Ex4\Resources\SubprocessDoc3_ConfigInput.txt**.
   Additionally check to get prompted to type/alter those data when debugging starts.
   Then click **Debug** to start process debugging.

Software AG Internal Use Only!

d. You should get prompted to switch to the Process Debug perspective. Click **Yes** to switch. Because of the settings you made in the Debug Configuration you get prompted for the input data having your sample data preloaded. Ensure you specify an unused value for field **CorrId**. Click **OK**.

e. Use the **Step Into** (F5) command of the **Trace** view to debug your process step by step. After starting the execution of the Subprocess you should be able to decide which of the parallel steps should be executed next:



Execute the steps in the sequence **Map To Subprocess Doc4** > **Sleep 5 Seconds** > **Subprocess Doc 4 thrown** and inspect the process data in the Pipeline Data view.

f. After executing the throwing Message Event, the Boundary Message Event of the subprocess should get triggered and interrupts all remaining activities within the subprocess. The subprocess gets decorated with an appropriate icon:

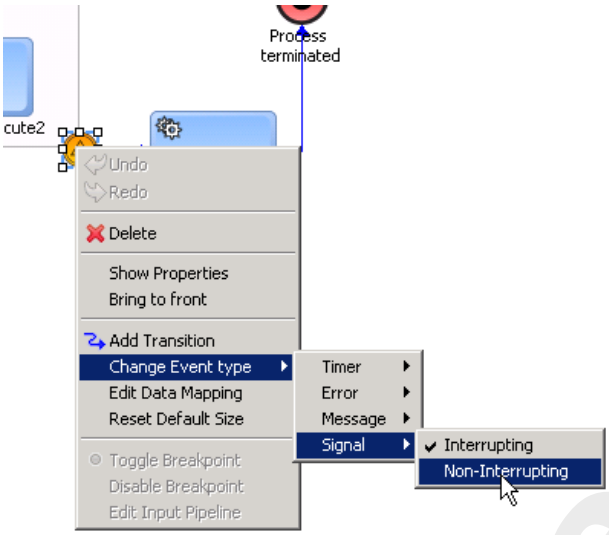
g. Execute the two remaining steps (**Sleep 2 Seconds** > **Process terminated)** to terminate your process successfully.

h. Because Debugger executes your process in the Process Engine, monitoring information is available. Logon to My webMethods in a browser tab (http://localhost:8585) to look for your debugged process instance of type **SubprocessWithParallelism** (**Applications** > **Monitoring** > **Business** > **Process Instances).**

Internal Use Only! Software AG

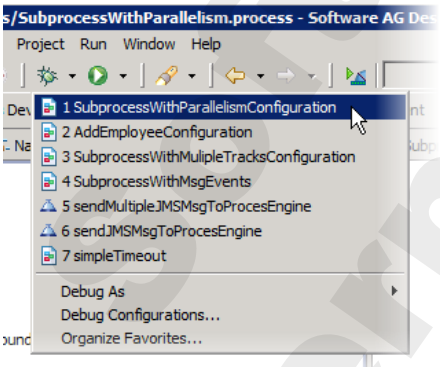
© 2000 – 2014 by Software AG

5. Modify and test process model **SubprocessWithParallelism** in Designer:
   a. Right-click the Boundary Signal Event on the Subprocess and change the behavior to become a non-interrupting Signal Event.



   Ensure **subprocessesSupport.docs:SubprocessDoc4** is still assigned as Receive document on the Implementation tab of the Boundary Signal Event's properties and the Receive protocol is **Subscription**. If not, change it.

   b. Change the Join operator of the final **End Terminate Event** to become **And**.
   c. Save, build and upload the modified process model **SubprocessWithParallelism**. When get prompted, allow enabling the process model for execution.
   d. Select your Debug Configuration **SubprocessWithParallelismConfiguration** from the drop down next to the Debug icon to start a new debug session.

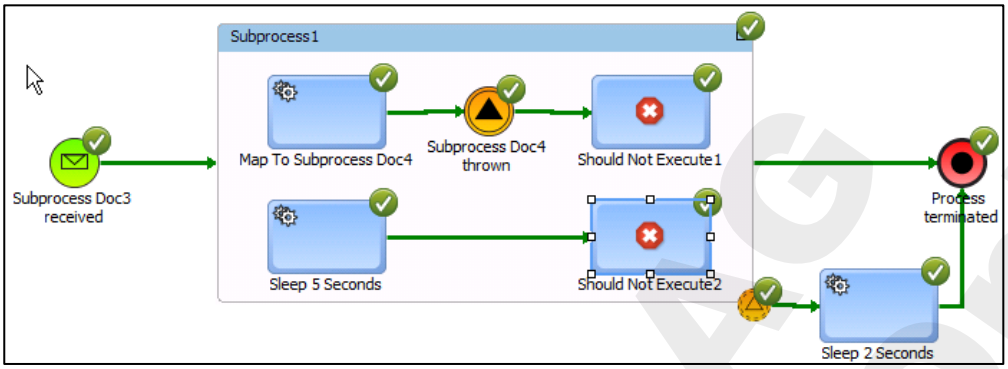


   When you get prompted for the input data you can reuse the sample data from the previous debug session, or load sample data from ***<workshop_dir>*\Ex4\Resources\SubprocessDoc3_DebugInput.txt,** but ensure you use an unused value for field **CorrId**. Click **OK**. If not already in, switch to the Process Debug perspective.

Software AG Internal Use Only!

e. This time use the Run icon to execute all the steps automatically. Inspect the sequence of the executed steps modeled to be in parallel. Steps **Should Not Execute 1** and **Should Not Execute 2** should have been executed. Why?

_____



6. Test process model **SubprocessWithMutlipleTracks** in Designer:

   a. Open process model **SubprocessWithMutlipleTracks** contained in the process project **Subprocesses**.

   b. Build and upload the process model. When get prompted, allow enabling the process model for execution.

   c. Create another Debug Configuration of asset type **Process Debug** in your Designer workspace. On the **Process** tab of your new Run Configuration, specify Run Configuration name **SubprocessWithMutlipleTracksConfiguration** and select the process model **SubprocessWithMutlipleTracks** within the process project **Subprocesses**.
   On the **Process Inputs** tab, load some input data for the Start Message Event from ***<workshop_dir>*\Ex4\Resources\SubprocessDoc5_ConfigInput.txt**.
   Additionally check to get prompted to type/alter those data when debugging starts.
   Then click **Debug** to start process debugging. Switch to the Process Debug perspective.

   d. When you get prompted for the input data, you can use the sample data configured in the Debug configuration. Click **OK**.
   Note: If you repeat debugging, always specify an unused value for field **CorrId**.

   e. Use the Run icon to execute all the steps automatically. Inspect the sequence of the executed steps modeled to be in parallel. Step **Should Not Execute** should not execute. Why?

   _____

   f. What would happen if you change the interrupting Boundary Event into a non-interrupting one?

   _____

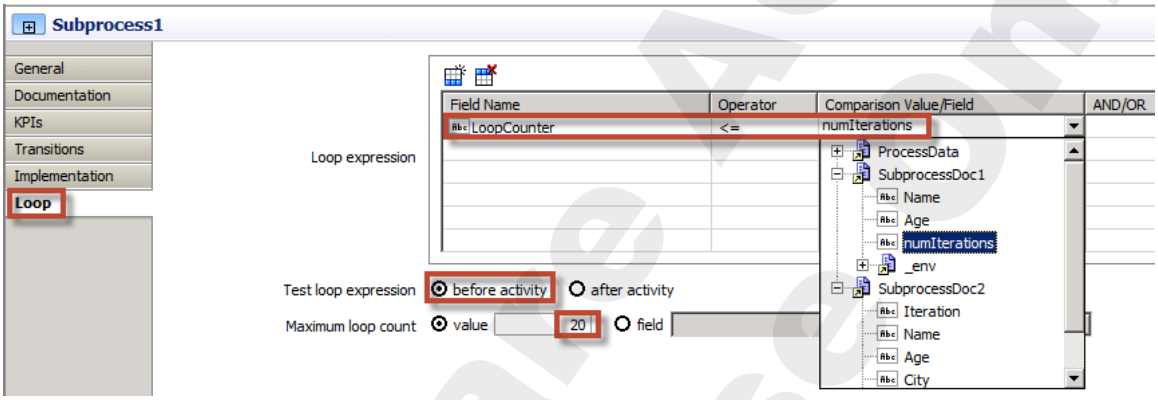
   g. What will be in the pipeline leaving the subprocess after the subprocess terminates successfully?

   _____

Software AG Internal use only!

© 2000 – 2014 by Software AG

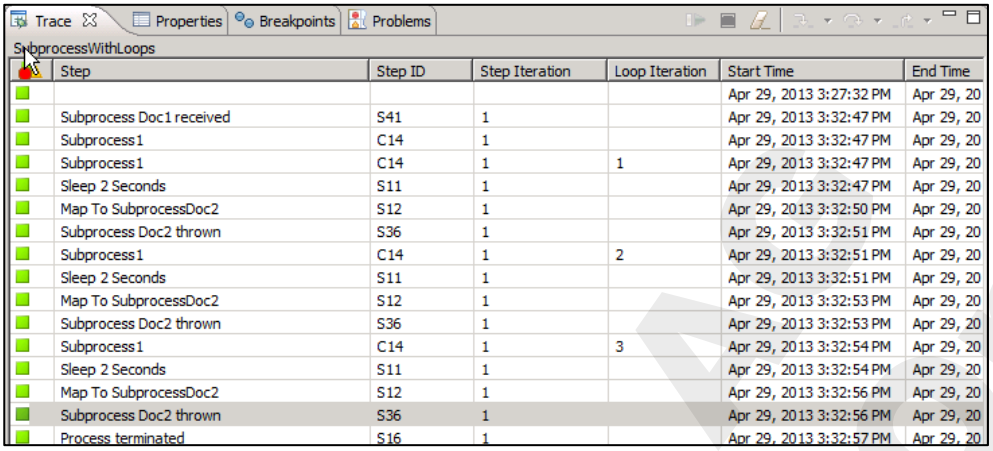
7. Test process model **SubprocessWithLoops** invoking process **ProcessDoc2** in Designer:
   a. Open process model **ProcessDoc2** contained in the process project **Subprocesses**.
   b. Build and upload the process model **ProcessDoc2**. When get prompted, allow enabling the process model for execution.
   c. Open process model **SubprocessWithLoops** contained in the process project **Subprocesses**. Select the BPMN subprocess **Subprocess1** in the process editor.
      In the Properties view, open the **Loop** tab and add a loop condition for the subprocess. Add the following loop expression:
      ProcessData/LoopCounter <= SubprocessDoc1/numIterations
      The expression should be tested **before** the subprocess is executed. Moreover specify a maximum loop count of **20** to avoid endless loops.



   d. Save, build and upload the process model. When get prompted, allow enabling the process model for execution.
   e. Create another Debug Configuration of asset type **Process Debug** in your Designer workspace. On the **Process** tab of your new Run Configuration, specify Run Configuration name **SubprocessWithLoopsConfiguration** and select the process model **SubprocessWithLoops** within the process project **Subprocesses**.
      On the **Process Inputs** tab, load some input data for the Start Message Event from ***<workshop_dir>*\Ex4\Resources\SubprocessDoc1_ConfigInput.txt**.
      Additionally check to get prompted to type/alter those data when debugging starts.
      Then click **Debug** to start process debugging. Switch to the Process Debug perspective.
   f. When you get prompted for the input data, you can use the sample data configured in the Debug configuration. Click **OK**.

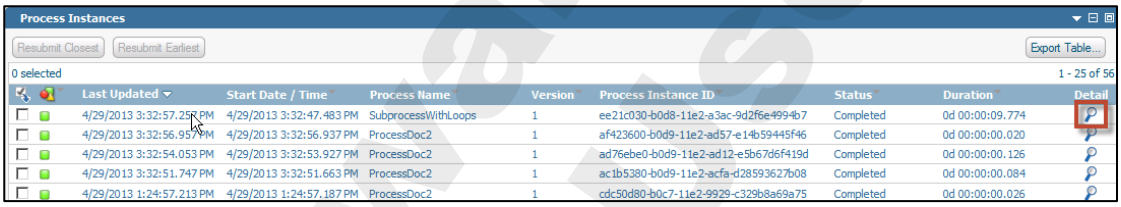
Software AG Internal use only!

g. Use the Step or Run icons to debug your process **SubprocessWithLoops.** Depending on the provided value of **SubprocessDoc1/numIterations** the process should execute the subprocess multiple times. This should be reflected by the **Loop Iteration** column in the Trace view:



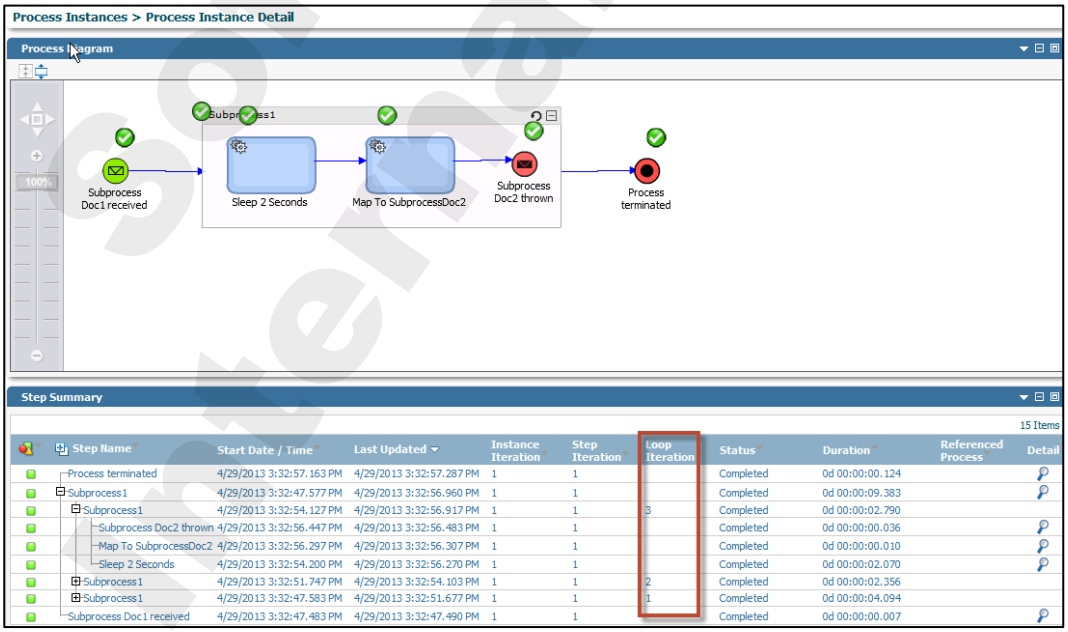
Step	Step ID	Step Iteration	Loop Iteration	Start Time	End Time
				Apr 29, 2013 3:27:32 PM	Apr 29, 20
Subprocess Doc1 received	S41	1		Apr 29, 2013 3:32:47 PM	Apr 29, 20
Subprocess1	C14	1		Apr 29, 2013 3:32:47 PM	Apr 29, 20
Subprocess1	C14	1	1	Apr 29, 2013 3:32:47 PM	Apr 29, 20
Sleep 2 Seconds	S11	1		Apr 29, 2013 3:32:47 PM	Apr 29, 20
Map To SubprocessDoc2	S12	1		Apr 29, 2013 3:32:50 PM	Apr 29, 20
Subprocess Doc2 thrown	S36	1		Apr 29, 2013 3:32:51 PM	Apr 29, 20
Subprocess1	C14	1	2	Apr 29, 2013 3:32:51 PM	Apr 29, 20
Sleep 2 Seconds	S11	1		Apr 29, 2013 3:32:51 PM	Apr 29, 20
Map To SubprocessDoc2	S12	1		Apr 29, 2013 3:32:53 PM	Apr 29, 20
Subprocess Doc2 thrown	S36	1		Apr 29, 2013 3:32:53 PM	Apr 29, 20
Subprocess1	C14	1	3	Apr 29, 2013 3:32:54 PM	Apr 29, 20
Sleep 2 Seconds	S11	1		Apr 29, 2013 3:32:54 PM	Apr 29, 20
Map To SubprocessDoc2	S12	1		Apr 29, 2013 3:32:56 PM	Apr 29, 20
Subprocess Doc2 thrown	S36	1		Apr 29, 2013 3:32:56 PM	Apr 29, 20
Process terminated	S16	1		Apr 29, 2013 3:32:57 PM	Apr 29, 20

h. In a browser tab logon to My webMethods (http://localhost:8585) as **Administrator | manage** to look for your debugged process instance of type **SubprocessWithLoops** (**Applications** > **Monitoring** > **Business** > **Process Instances).**
Because the looping subprocess throws multiple documents of type SubprocessDoc2, several process instances of type **ProcessDoc2** have been started also.



Last Updated	Start Date / Time	Process Name	Version	Process Instance ID	Status	Duration	Detail
4/29/2013 3:32:57.25[illegible] PM	4/29/2013 3:32:47.483 PM	SubprocessWithLoops	1	ee21c030-b0d8-11e2-a3ac-9d2f6e4994b7	Completed	0d 00:00:09.774	
4/29/2013 3:32:56.957 PM	4/29/2013 3:32:56.937 PM	ProcessDoc2	1	af423600-b0d9-11e2-ad57-e14b59445f46	Completed	0d 00:00:00.020	
4/29/2013 3:32:54.053 PM	4/29/2013 3:32:53.927 PM	ProcessDoc2	1	ad76ebe0-b0d9-11e2-ad12-e5b67d6f419d	Completed	0d 00:00:00.126	
4/29/2013 3:32:51.747 PM	4/29/2013 3:32:51.663 PM	ProcessDoc2	1	ac1b5380-b0d9-11e2-acfa-d28593627b08	Completed	0d 00:00:00.084	
4/29/2013 1:24:57.213 PM	4/29/2013 1:24:57.187 PM	ProcessDoc2	1	cdc50d80-b0c7-11e2-9929-c329b8a69a75	Completed	0d 00:00:00.026	

i. Click the magnifying glass icon to open the details of your debugged process **SubprocessWithLoops**. Have a look for the loop counter in the step list here, too.



Step Name	Start Date / Time	Last Updated	Instance Iteration	Step Iteration	Loop Iteration	Status	Duration	Referenced Process	Detail
Process terminated	4/29/2013 3:32:57.163 PM	4/29/2013 3:32:57.287 PM	1	1		Completed	0d 00:00:00.124		
Subprocess1	4/29/2013 3:32:47.577 PM	4/29/2013 3:32:56.960 PM	1	1		Completed	0d 00:00:09.383		
Subprocess1	4/29/2013 3:32:54.127 PM	4/29/2013 3:32:56.917 PM	1	1	3	Completed	0d 00:00:02.790		
Subprocess Doc2 thrown	4/29/2013 3:32:56.447 PM	4/29/2013 3:32:56.483 PM	1	1		Completed	0d 00:00:00.036		
Map To SubprocessDoc2	4/29/2013 3:32:56.297 PM	4/29/2013 3:32:56.307 PM	1	1		Completed	0d 00:00:00.010		
Sleep 2 Seconds	4/29/2013 3:32:54.200 PM	4/29/2013 3:32:56.270 PM	1	1		Completed	0d 00:00:02.070		
Subprocess1	4/29/2013 3:32:51.747 PM	4/29/2013 3:32:54.103 PM	1	1	2	Completed	0d 00:00:02.356		
Subprocess1	4/29/2013 3:32:47.583 PM	4/29/2013 3:32:51.677 PM	1	1	1	Completed	0d 00:00:04.094		
Subprocess Doc1 received	4/29/2013 3:32:47.483 PM	4/29/2013 3:32:47.490 PM	1	1		Completed	0d 00:00:00.007		

Software AG Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 5: EDA PROCESS

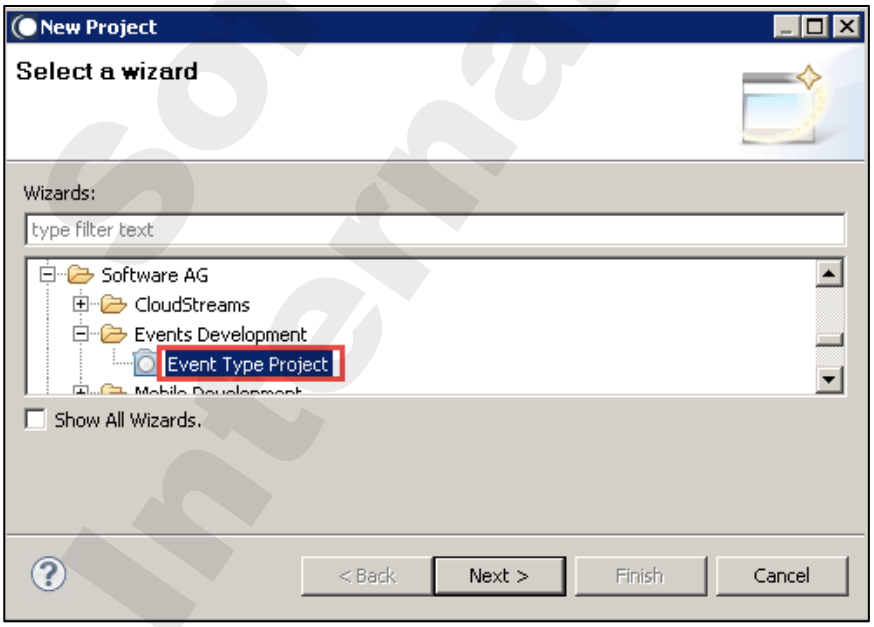
Overview

In this exercise you will create a process model that gets triggered by an EDA Event already defined in the Common Event Type Store. The process will emit a second custom EDA Event that has to be defined at the Event Type Store first before the process can be executed and emits the custom EDA Event. Starting the process will be accomplished via an IS service using NERV, emitted EDA Events will we caught by running the Event Bus Console.
Optionally, you will adapt the underlying infrastructure to use webMethods Broker instead of Universal Messaging for EDA connectivity.

In this exercise, please replace:
<workshop_dir> by **C:\Training\602D-66E\BPM**

Steps

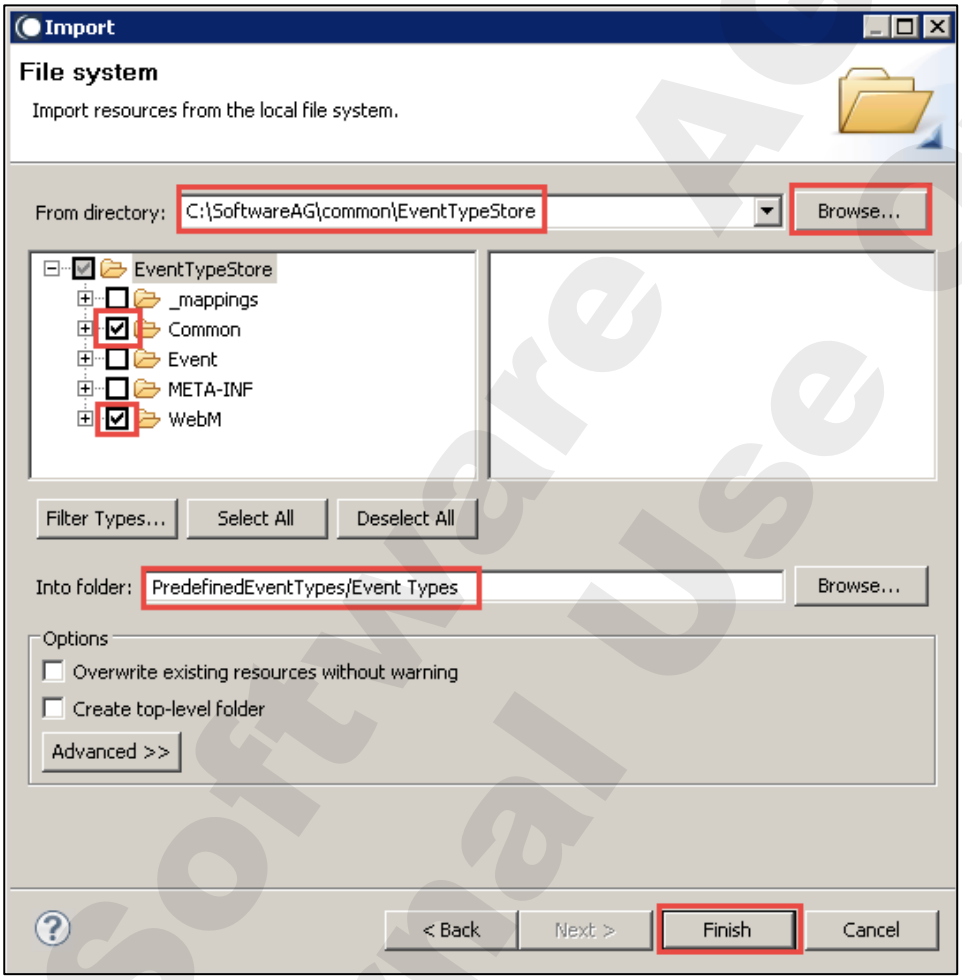
1. In a browser tab, login to the IS Administration Console (http://localhost:5555) as **Administrator | manage.** Navigate to **Packages** > **Management** to enable the already loaded IS package **EDAProcessesSupport**.
2. Start Designer using workspace **C:\Users\Administrator\workspace96.** Import provided sample Event Types into an Event Type Project in Designer. To do so:
   a. In Designer, switch to **the Process Development** perspective. Use **Window** > **Show View** > **Other…** to add the **Project Explorer** view to your perspective.
   b. In the Project Explorer view, add a new project of type **Software AG > Events Development > Event Type Project.** Click **Next.**



Software AG Internal Use Only!

Specify **PredefinedEventTypes** as project name and hit **Finish**.

c. Right-click the generated empty system folder **Event Types** within the project **PredefinedEventTypes** and select **Import**. Choose **General** > **File System** as import source and browse for the directory **C:\SoftwareAG\common\EventTypeStore**. Expand the directory tree and check subfolders **Common** and **WebM** only. Double-check that folder **PredefinedEventTypes/Event Types** becomes the target folder. Click **Finish** to import the Event Types from the selected subfolders.



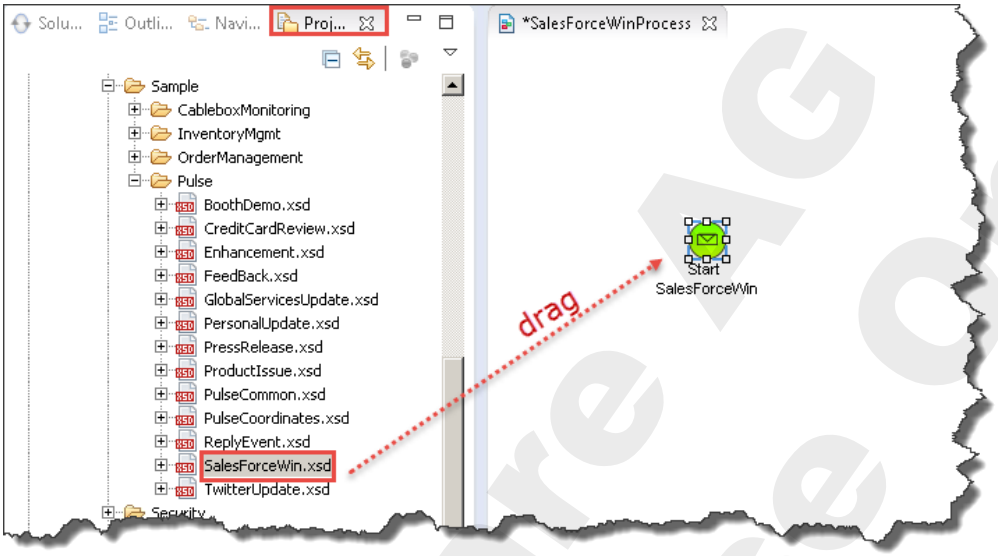
3. Switch to the Solutions view and create a new process project named **EDAProcesses**.
4. Add a new process model named **SalesForceWinProcess** to your process project **EDAProcesses.**

Software AG Internal Use Only!

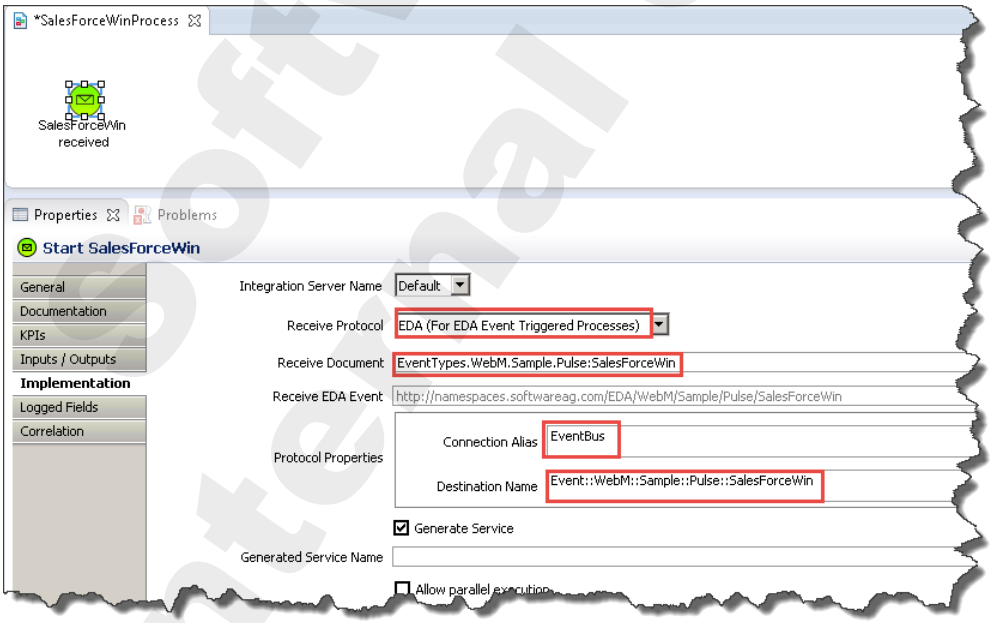
© 2000 – 2014 by Software AG

5. Enhance the empty process model **SalesForceWinProcess** like this:

   a. Use the Project Explorer and open the imported Event Type project **PredefinedEventTypes.** Drill down to folder **Event Types/WebM/Sample/Pulse.** Drag the Event Type **SalesForceWin.xsd** from the Project Explorer into your empty process model **SalesForceWinProcess** still opened in the process editor**.** You'll get prompted for an Integration Server which will be used to store a new IS document type related to the chosen Event Type. Click **OK** to accept the **Default** Integration Server.

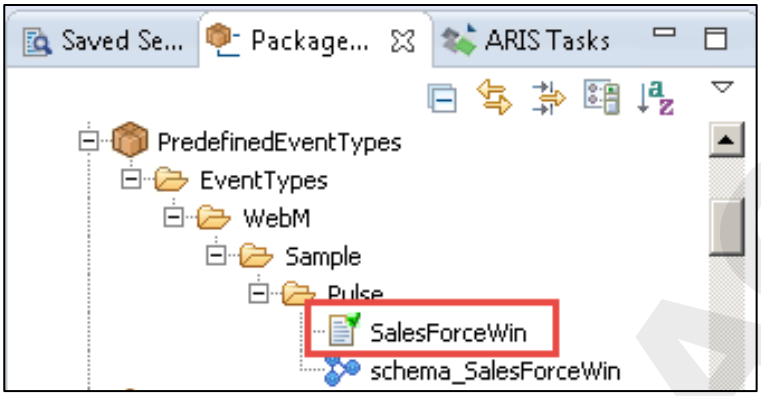


   b. In the Process editor, rename the generated Start Message Event to **SalesForceWin received.** In the Properties view of the Start Message Event, open the **Implementation** tab. Double-check all EDA properties have been set as expected:



Internal Use Only! Software AG

c. Refresh the Package Explorer view. Have a look for the generated IS document type **EventTypes.WebM.Sample.Pulse:SalesForceWin** within IS package **PredefinedEventTypes** reflecting the namespace of the Event Type in the Event Type project.



d. Drag the provided IS service **EDAProcessesSupport.services:mapToSalesforceWinClassified** from the Package Navigator into the process editor to create and configure a Service Task Activity. Rename the Service Task Activity into **Classify SalesForceWin.**
Note: This step maps the incoming document type **EventTypes.WebM.Sample.Pulse:SalesForceWin** as generated in the IS package **PredefinedEventTypes** to a document of type **EventTypes.Sales:SalesForceWinClassified**. This IS document type already exists in the IS provided in package **EDAProcessesSupport**. Additionally, the step writes a message to the IS Server log reporting about the classification of the Sales Force win.

e. Using the speed panel, add an **End None Event** and a transition next to the Service Task Activity **Classify SalesForceWin.** Right-click the event to change its type to **End Terminate.** Rename the End Terminate Event to **Process terminated.**

f. Add a transition from the Start Message Event **SalesForceWin received** to the Service Task Activity **Classify SalesForceWin.**
Your process model should now look like this:

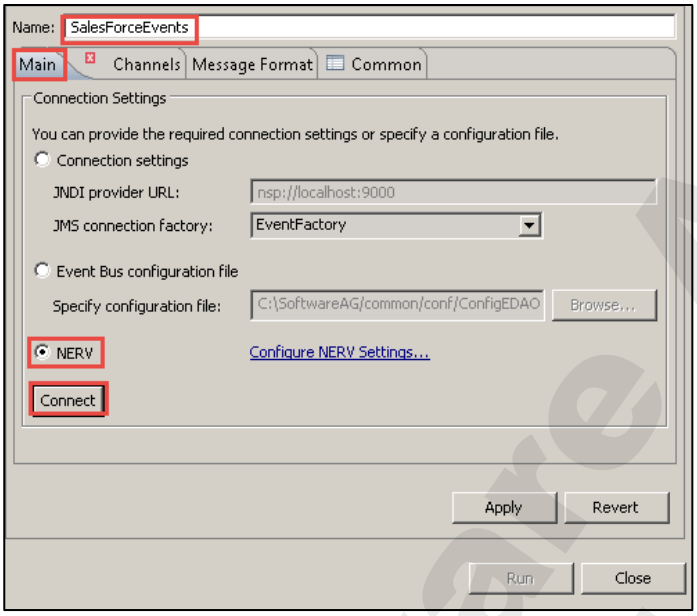
6. Save, build and upload your process model. When get prompted, allow enabling the process model for execution.

Internal Use Only! Software AG

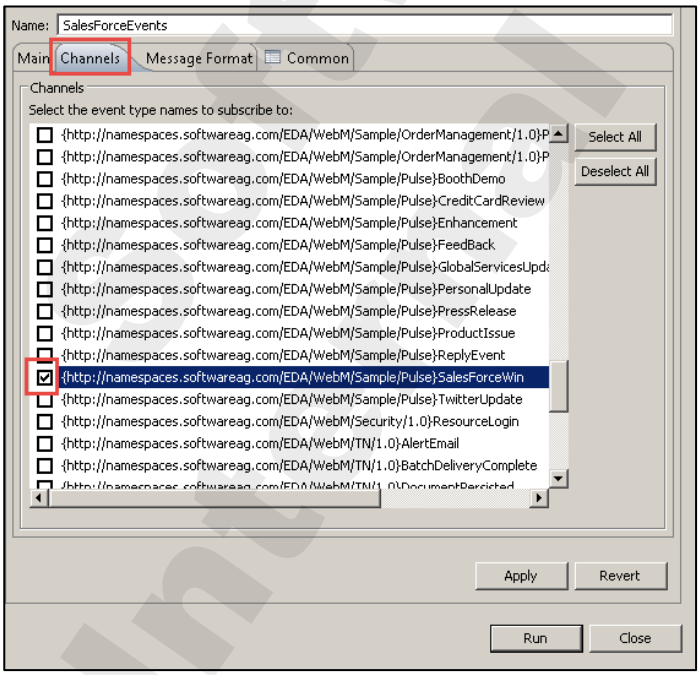
© 2000 – 2014 by Software AG

7. Test the process model **SalesForceWinProcess:**

   a. Switch to the **Events Development** perspective. Use the icon to open the Event Bus Console. Click to create a new Event Bus configuration. On the **Main** tab specify **SalesForceEvents** as configuration name, select NERV as access method to connect to the Event Bus and an Event Type Store. Then hit **Connect.**

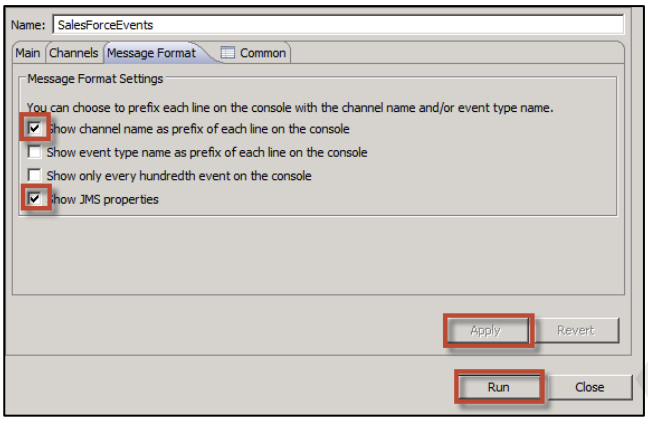


On the **Channels** tab, check the Event Type **SalesForceWin** in the list of Event Types retrieved from the Event Type Store.



Internal Use Only! Software AG

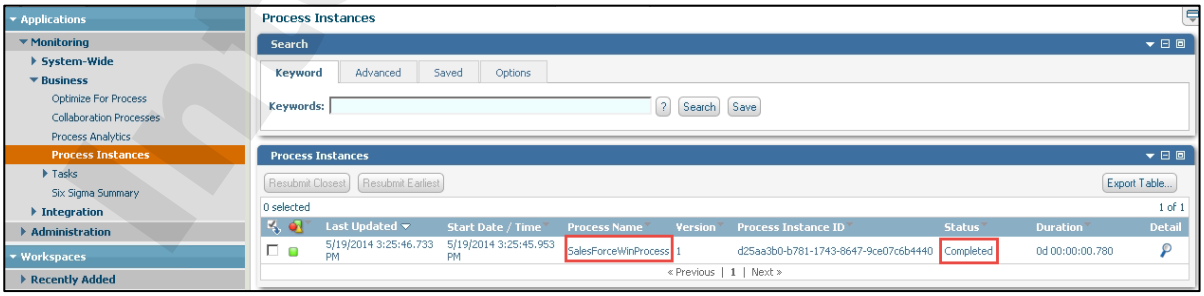
On the **Message Format** tab, only select **Show channel name as prefix…** and **Show JMS properties**. Hit **Apply** and **Run** to start the Event Bus Console listening via NERV for incoming events of type **SalesforceWin**.



b. Switch to the **Process Development** perspective. From the Package Navigator, run the provided IS service **EDAProcessesSupport.services:emitSalesForceWinViaNERV** contained in the IS package **EDAProcessesSupport** (**Run As > Run Flow Service**).
You can use file ***<workshop-dir>*\Ex5\Resources\ SalesForceWin_Input.txt** to provide input.

c. Switch back to the **Events Development** perspective. Use the **Console** view to look for thrown events of type **SalesForceWin.**

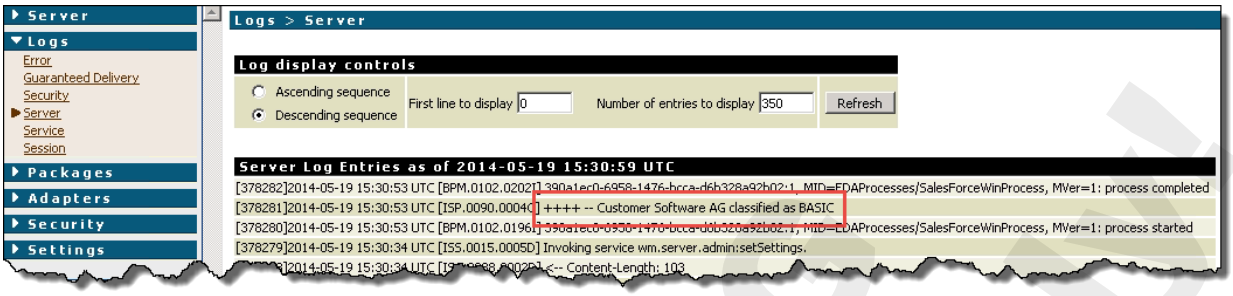
d. Use a browser tab (http://localhost:8585) to login to My webMethods as **Administrator | manage**. Drill down to **Applications** > **Monitoring** > **Business** > **Process Instances** and ensure a process instance of type **SalesForceWinProcess** has been started end executed successfully.



Internal Use Only! Software AG

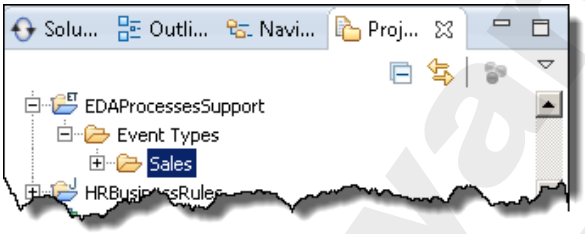
© 2000 – 2014 by Software AG

e. In a second browser tab (http://localhost:5555), open the IS Administration console as **Administrator** |**manage**. Navigate to **Logs** > **Server** to see the message written by the IS service EDAProcessesSupport.services:mapToSalesforceWinClassified in the IS Server log.

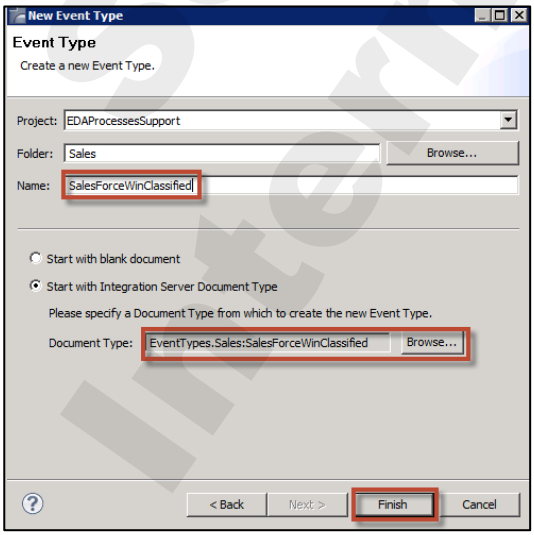


8. Create and deploy a new Event Type based on the existing IS document type **SalesForceWinClassified:**

   a. Open the Process Development perspective in Designer. Use the **Project Explorer** view (as added in step 2a) to add a new project of type **Software AG > Events Development > Event Type Project** named **EDAProcessesSupport.** Add a subfolder named **Sales** to the auto-generated folder **Event Types.**
   Note: This is to be in sync with the IS package that contains the IS document type.

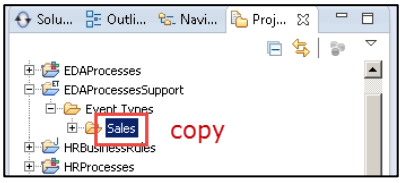


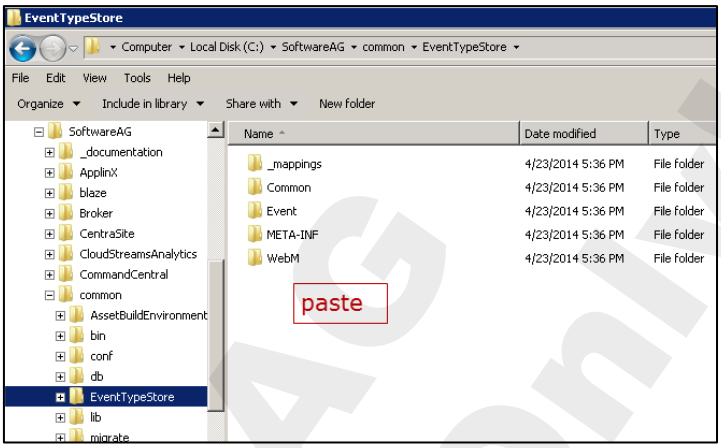
   b. Right-click folder **Sales** and select **New > Other… .** Browse for asset type **Software AG** > **Events Development** > **Event Type** to create a new Event Type named **SalesForceWinClassified** based on an IS document type.
   Browse for IS document type **EventTypes.Sales:SalesForceWinClassified** contained in the IS package **EDAProcessesSupport** and click **Finish**.



Software AG Internal Use Only!

c. To deploy the new Event Type **SalesForceWinClassified** to the common Event Type Store, simply copy the entire folder **Sales** in Designer's Project Explorer and paste it to the folder **C:\SoftwareAG\common\EventTypeStore** using a Windows Explorer.

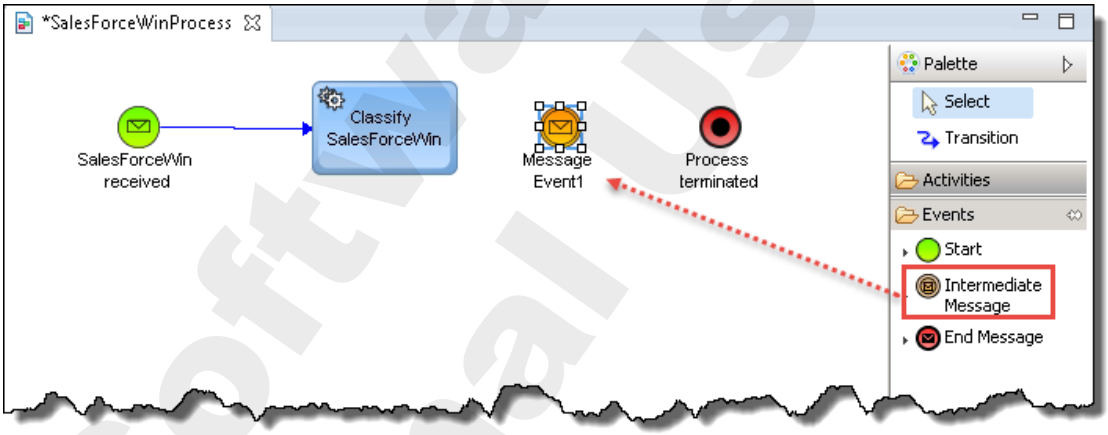




9. In Designer using the Process Development perspective, modify and enhance your process model **SalesForceWinProcess** like this:

a. Delete the transition from Service Task Activity **Classify SalesForceWin** to End Terminate Event **Process terminated.**

b. Use the palette to add an **Intermediate Message Event** and insert it next to Service Task Activity **Classify SalesForceWin.**



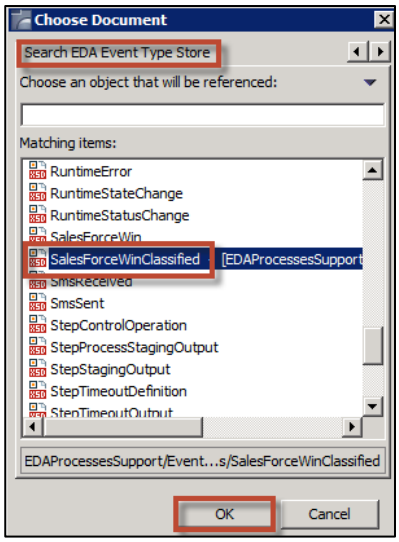
c. Use Properties view for the Intermediate Message Event to customize its **Implementation** properties:

- Set the Protocol to **EDA.**

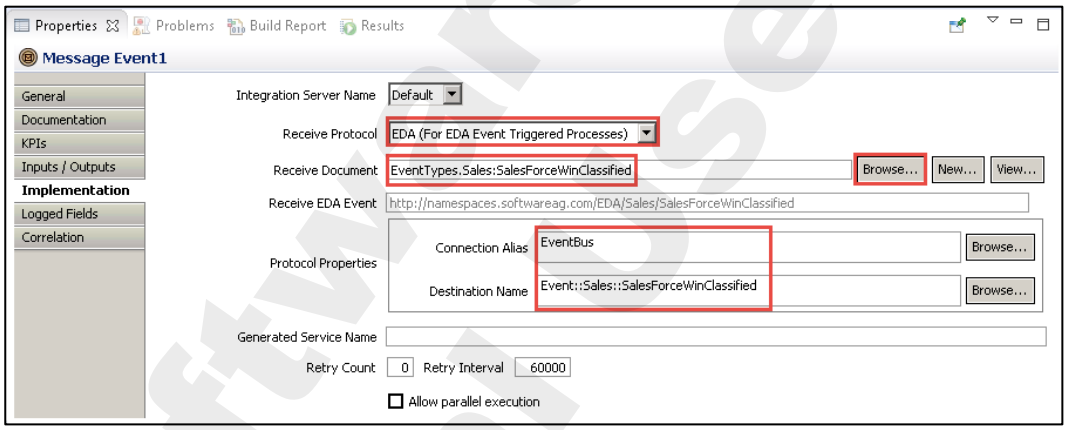
Software AG Internal Use Only!

© 2000 – 2014 by Software AG

- Use the Browse button for field **Receive Document**. Browse for your new Event Type **SalesForceWinClassified** in the Event Type Store and click **OK.**

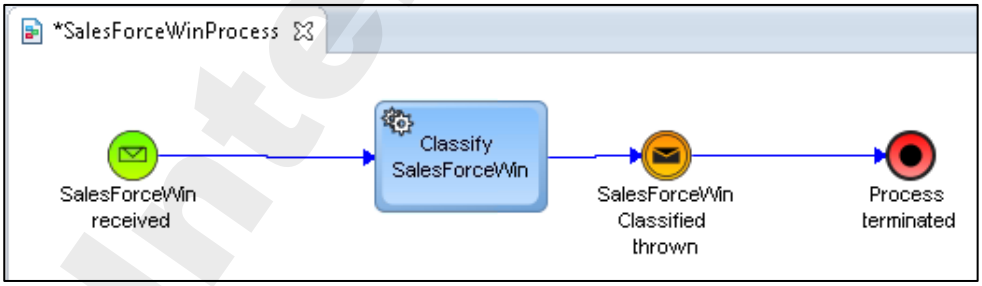


- Steps above should also fill the values for the properties **Receive EDA Event, Connection Alias,** and **Destination Name** like this:



d. Use the **General** tab of the Properties view to rename your Intermediate Message Event to **SalesForceWinClassified thrown.** Change the Event to become a **throwing** Message Event.

e. Use the speed panel to add a transition from Service Task Activity **Classify SalesForceWin** to Message Event **SalesForceWinClassified thrown** and another from Event **SalesForceWinClassified thrown** to End Terminate Event **Process terminated.**



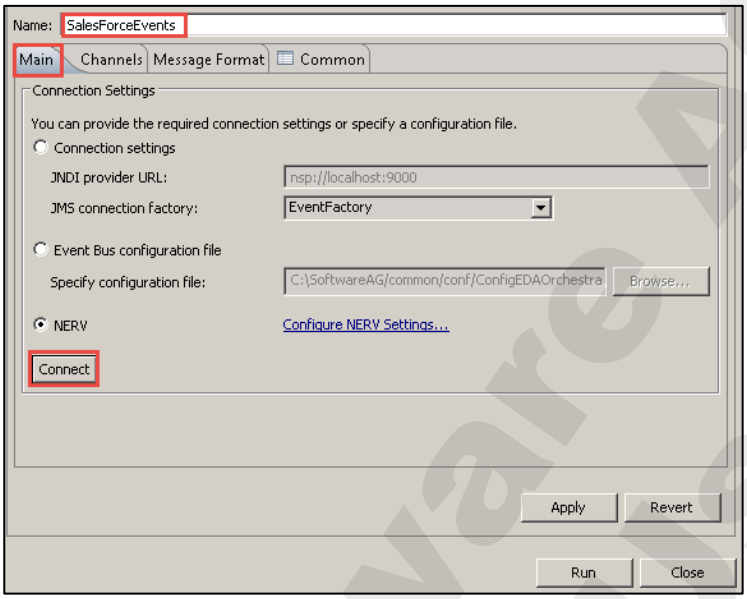
Software AG Internal Use Only!

10. Inspect the Properties of Service Task Activity **Classify SalesForceWin**.
    Double-check the step creates the necessary output document of type **EventTypes.Sales:SalesForceWinClassified** named **SalesForceWinClassified.**

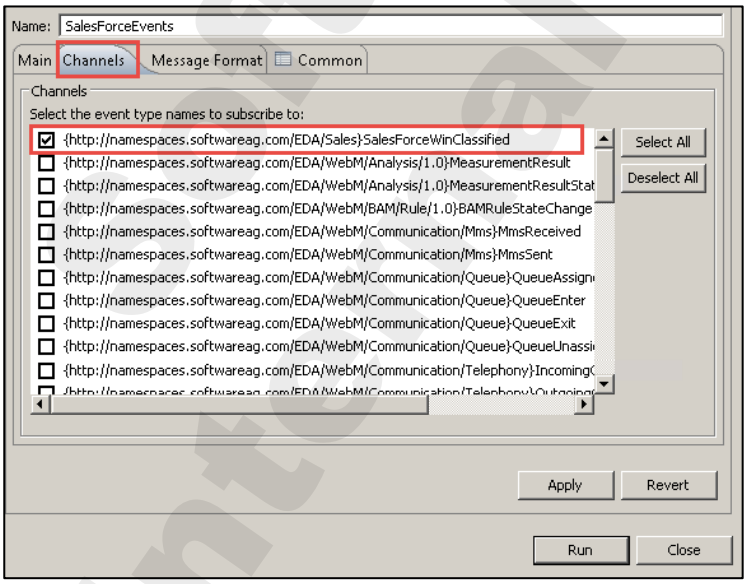
11. Save, build and upload your process model **SalesForceWinProcess**.

12. Test the process model **SalesForceWinProcess** again:

    a. Switch to the **Events Development** perspective. Use the icon to modify your existing Event Bus Console configuration **SalesForceEvents**.
    On the **Main tab,** hit **Connect** to reload the existing Events from the Event Type Store.



On the **Channels** tab, additionally check the Event Type **SalesForceWinClassified** in the list of Event Types retrieved from the Event Type Store (leave **SalesForceWin** checked).



Hit **Apply** and **Run** to start the Event Bus Console listening via NERV for incoming events of both types, **SalesforceWin** and **SalesForceWinClassified.**

Internal Use Only! Software AG

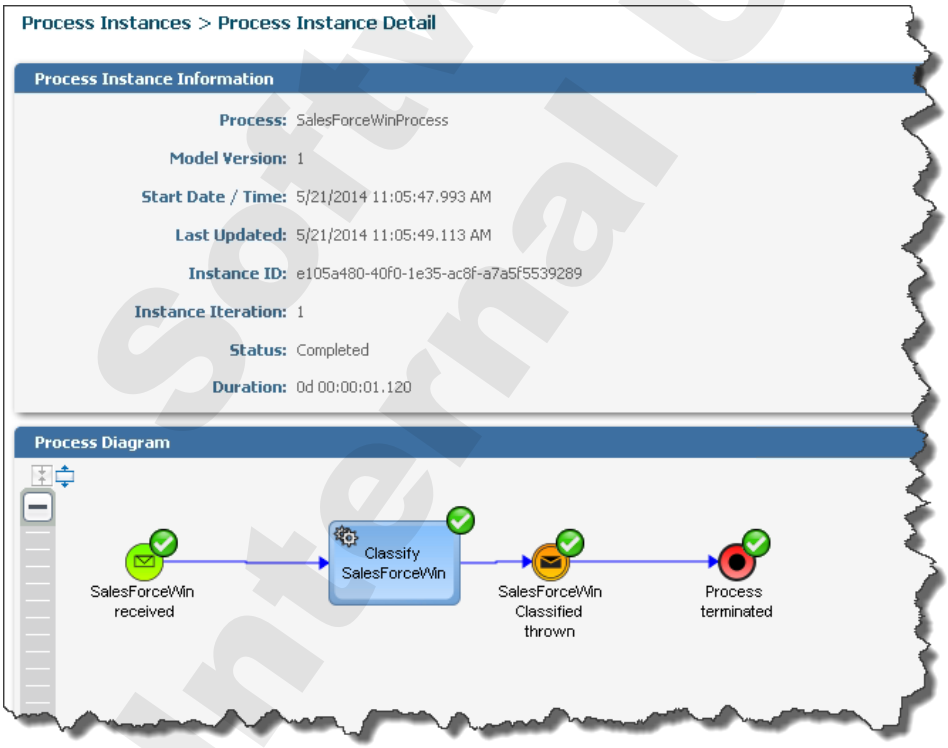
© 2000 – 2014 by Software AG

b. Switch to the **Process Development** perspective. From the Package Navigator, rerun the provided IS service **EDAProcessesSupport.services:emitSalesForceWinViaNERV** contained in the IS package **EDAProcessesSupport** (**Run As > Run Flow Service**).
You can again use file ***<workshop-dir>*\Ex5\Resources\SalesForceWin_Input.txt** to provide the same input.

c. Switch back to the **Events Development** perspective. Use the **Console** view to look for thrown events of types **SalesForceWin** and **SalesForceWinClassified.**

```
Problems  Properties  Console
Event Bus Console [SalesForceEvents]
[Event::Sales::SalesForceWinClassified] [6] breadcrumbId = ID-sgcloud20-49171-1367992133878-0-26
[Event::Sales::SalesForceWinClassified] [7] JMSXGroupID = null
[Event::Sales::SalesForceWinClassified] [8] JMSCorrelationID = null
[Event::Sales::SalesForceWinClassified] [9] JMSPriority = 4
[Event::Sales::SalesForceWinClassified] [10] JMSXDeliveryCount = 1
[Event::Sales::SalesForceWinClassified] [11] JMSDestination = com.webmethods.jms.impl.WmTopicImpl[name=Event::Sales::SalesForceWinClas
[Event::Sales::SalesForceWinClassified] [12] JMSDeliveryMode = 2
[Event::Sales::SalesForceWinClassified] [13] $Event$Start = 2013-05-08T16:24:12.029+01:00
[Event::Sales::SalesForceWinClassified] [14] $Event$EventID = a5eb18eb-2208-41a6-aec9-8ae1a095a9f8
[Event::Sales::SalesForceWinClassified] [15] JMSRedelivered = false
[Event::Sales::SalesForceWinClassified] [16] JMSReplyTo = null
[Event::Sales::SalesForceWinClassified] [17] JMSType = null
[Event::Sales::SalesForceWinClassified] <SalesForceWinClassified>
[Event::Sales::SalesForceWinClassified]   <CustomerName>Software AG</CustomerName>
[Event::Sales::SalesForceWinClassified]   <Amount>1000000</Amount>
[Event::Sales::SalesForceWinClassified]   <Region>Europe</Region>
[Event::Sales::SalesForceWinClassified]   <Products>IT</Products>
[Event::Sales::SalesForceWinClassified]   <SalesRep>obr</SalesRep>
[Event::Sales::SalesForceWinClassified]   <CustomerRating>BASIC</CustomerRating>
[Event::Sales::SalesForceWinClassified] </SalesForceWinClassified>
```

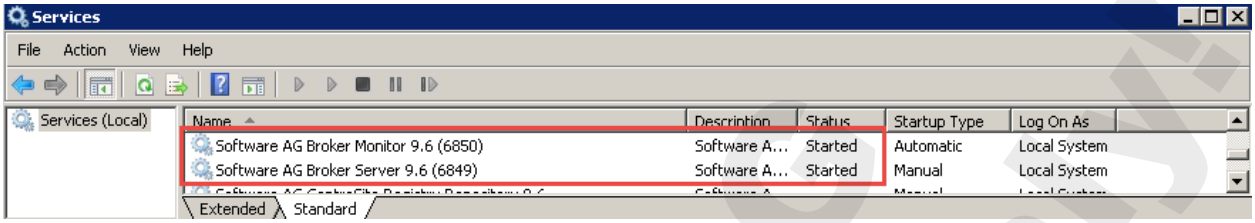
13. Use a browser tab (http://localhost:8585) to login to My webMethods as **Administrator | manage**. Drill down to **Applications** > **Monitoring** > **Business** > **Process Instances** and ensure a process instance of type **SalesForceWinProcess** has been started end executed successfully.



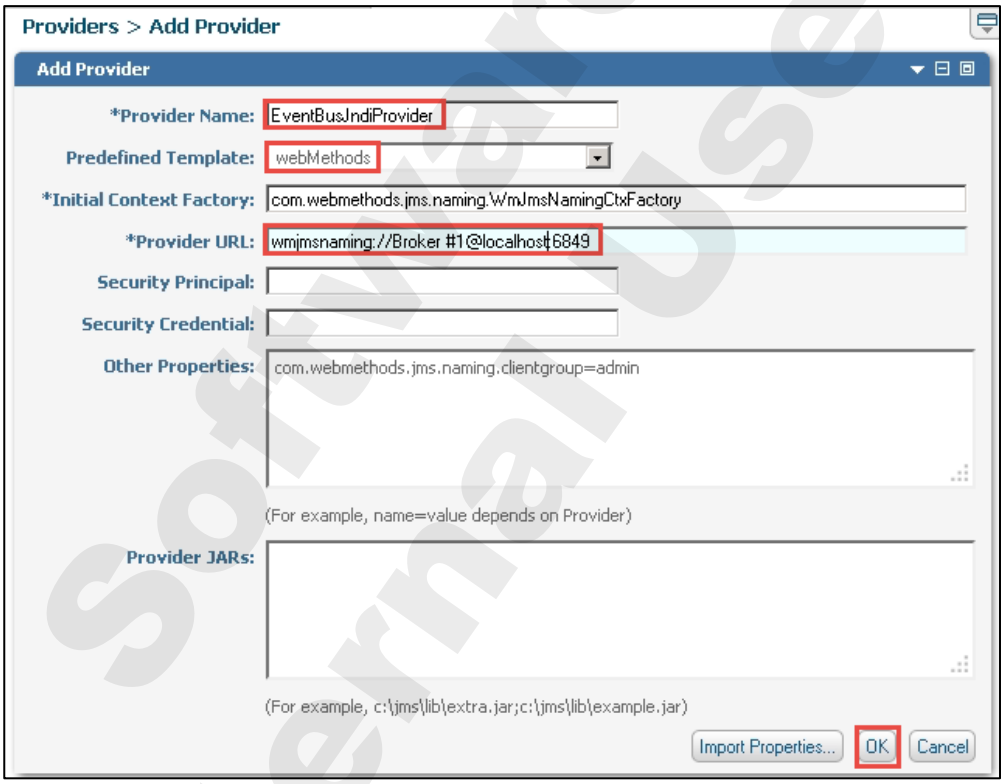
Software AG Internal Use only!

14. Finally modify your environment to use a different Provider (e.g. another Universal Messaging realm or webMethods Broker) as underlying Message Provider for your EDA-based process.
    In this exercise, we will use webMethods Broker as an alternative:

    a. Use Windows service control panel to ensure services **Software AG Broker Monitor** and **Software AG Broker Server** are up and running.



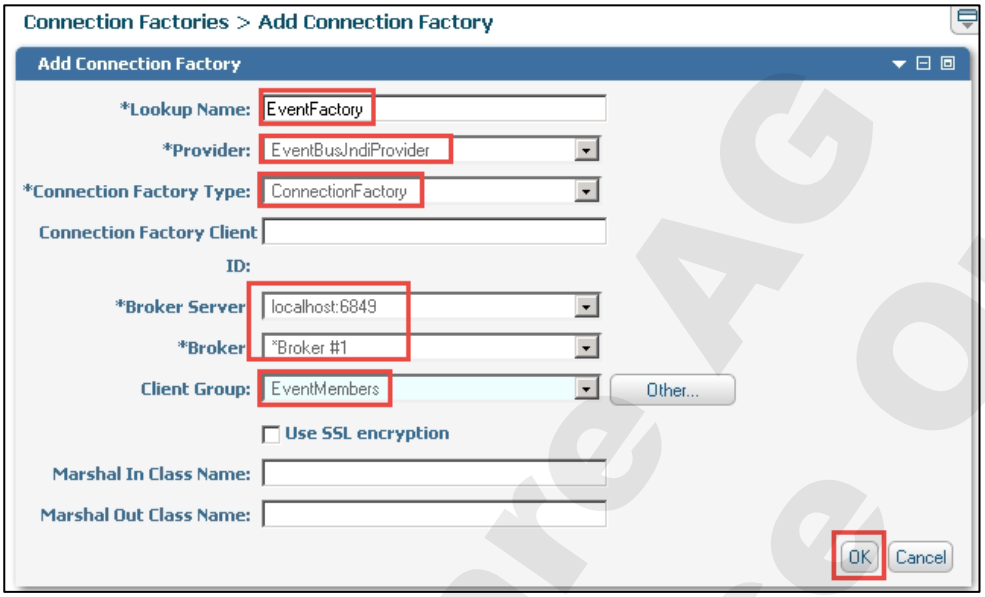
    b. Reuse the My webMethods browser tab (http://localhost:8585).
    Drill down to **Applications** > **Administration** > **Messaging** > **Naming Directories** > **Providers.** Add a new JNDI provider to the Provider list. Specify **EventBusJndiProvider** as Provider Name, select **webMethods** as Predefined Template (will set the initial context factory) and adapt the Provider URL to a value of **wmjmsnaming://Broker #1@localhost:6849**.
    Leave all other values unchanged and hit **OK**.



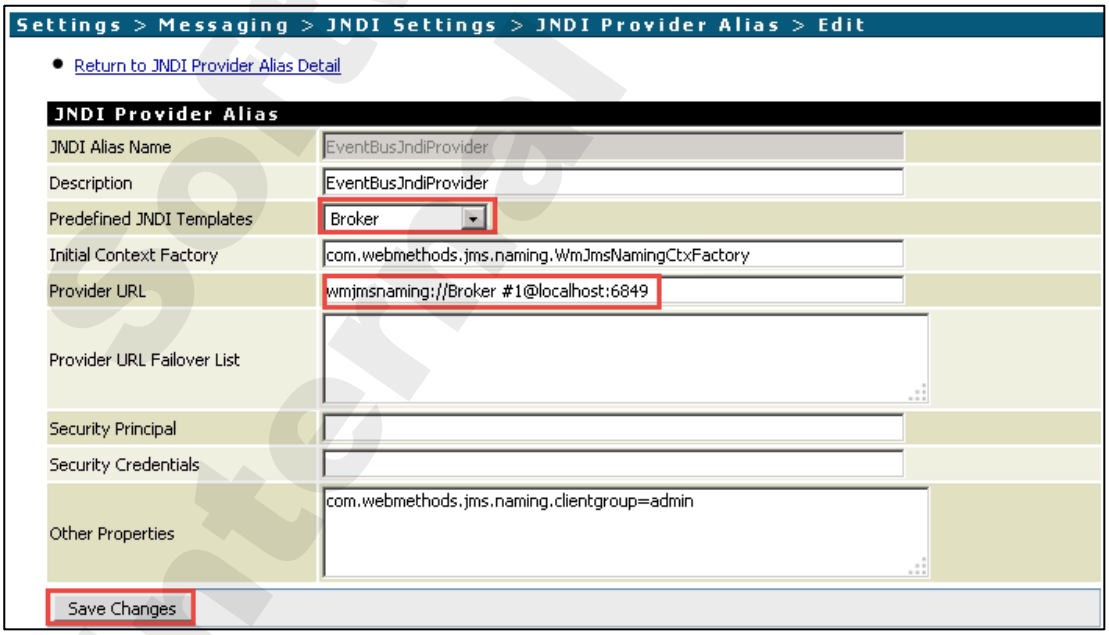
Internal Use Only Software AG

© 2000 – 2014 by Software AG

c. In My webMethods, navigate to **Applications** > **Administration** > **Messaging** > **Naming Directories** > **Connection Factories.** Add a new Connection Factory to the list of Connection Factories. Provide **EventFactory** as new Lookup Name, select **EventBusJndiProvider** from the list of existing providers, choose **Connection Factory** as factory type, and select **EventMembers** as Broker Client Group. **Broker Server** and **Broker** should be filled in automatically. Hit **OK** to create the Lookup Name.

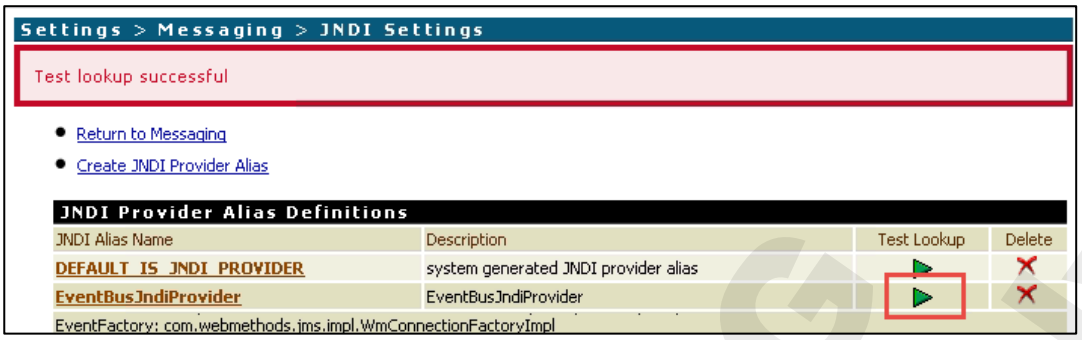


d. In another browser tab (http://localhost:5555), open the IS Administration console as **Administrator |manage**. Navigate to **Settings** > **Messaging** > **JNDI Settings.** Select and **edit** the existing JNDI Provider Alias **EventBusJndiProvider.**
Switch to **Broker** as predefined JNDI template and adjust the Provider URL to **wmjmsnaming://Broker #1@localhost:6849.** Save your changes.

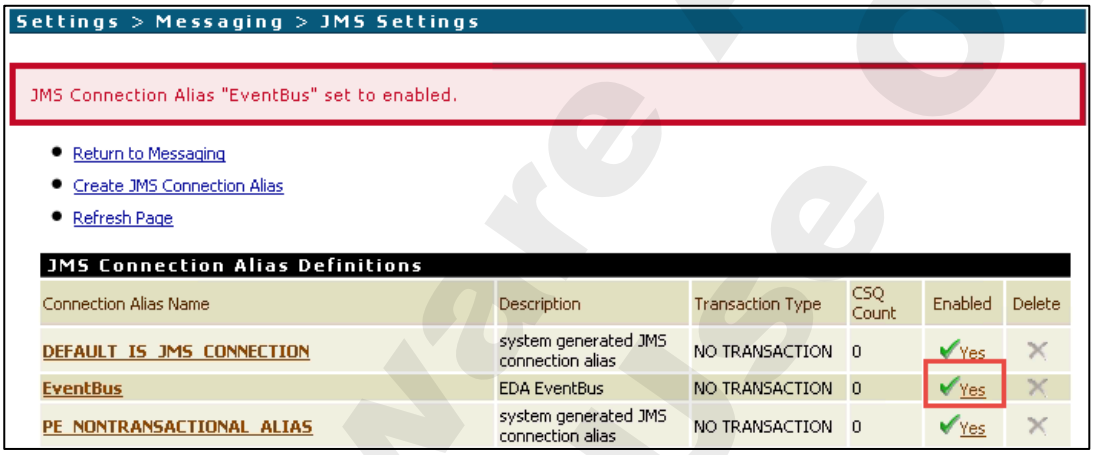


Software AG Internal Use Only!

e. Return to the JNDI Settings page. Perform a **Test Lookup** using your modified JNDI Provider Alias **EventBusJndiProvider.**

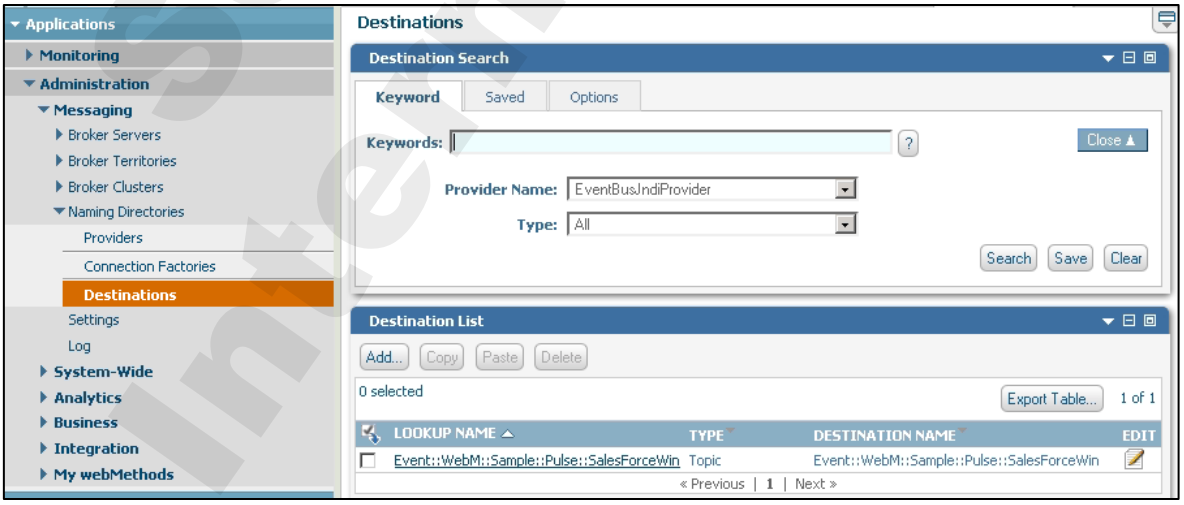


f. In the IS Administration Console, navigate to **Settings** > **Messaging > JMS Settings. Disable** and **enable** the pre-configured JMS Connection Alias **EventBus** which uses your modified JNDI Connection Alias **EventBusJndiProvider.**



15. Go back to Designer and re-build your process model **SalesForceWinProcess.**

16. Revisit My webMethods in a browser tab. Navigate to **Applications** > **Administration** > **Messaging** > **Naming Directories** > **Destinations.** The build process should have added a new destination (within webMethods Broker) named **Event::WebM::Sample::Pulse::SalesForceWin** of type **Topic** for the Start Message Event:



Software AG Internal Use Only!

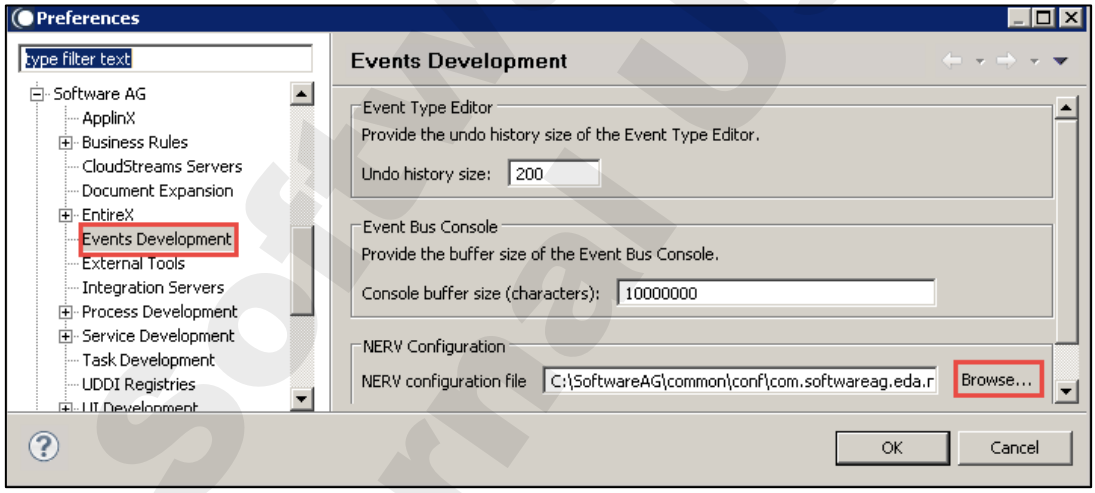
© 2000 – 2014 by Software AG

17. Because we are using the IS service **pub.event.nerv:send** within the service **EDAProcessesSupport.services:emitSalesForceWinViaNERV** to emit an event that starts our process, we have to adjust the IS NERV configuration also.
    Use a file explorer to locate file **com.softwareag.eda.nerv.properties** stored in folder **C:\SoftwareAG\profiles\IS_default\configuration\com.softwareag.platform.config.propsloader.**
    Use a text editor, e.g. Notepad++, to adjust the JMS provider URL used by NERV to access webMethods Broker:

```
#Wed Apr 23 18:04:58 UTC 2014
com.softwareag.eda.nerv.dead.letter.channel.uri=@url\:sag.install.area/common/nerv/dlc
com.softwareag.eda.nerv.in.memory.channel.size=1000
com.softwareag.eda.nerv.orchestrator.compatibility.mode=false
com.softwareag.eda.nerv.maximum.redelivery.attempts=100
enabled=true
com.softwareag.eda.nerv.cache.configuration.location=@path\:sag.install.area/common/conf/nerv/cache
com.softwareag.eda.nerv.default.jms.provider=wmjmsnaming\://Broker \#1@localhost\:6849
#com.softwareag.eda.nerv.default.jms.provider=nsp\://localhost\:9000
#com.softwareag.eda.nerv.default.jms.provider=nsp\://localhost\:19000
```

    Save your changes.

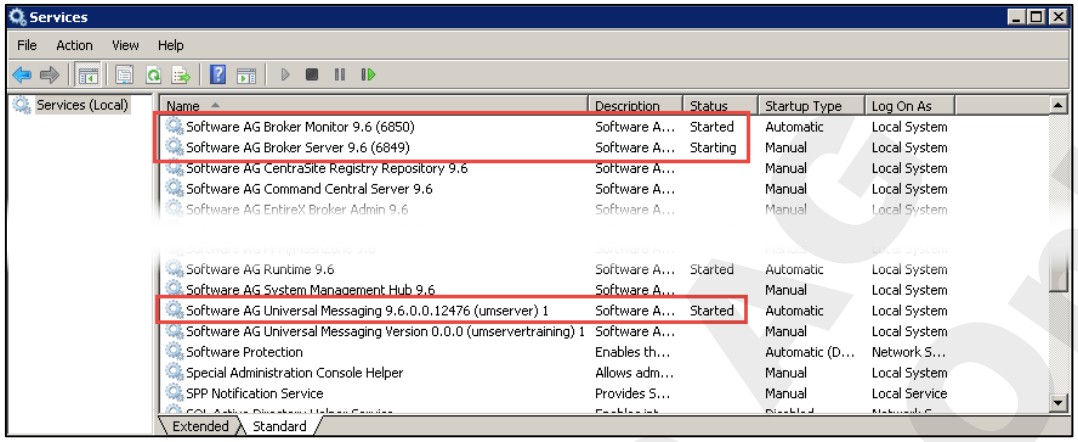
18. Re-open Designer and navigate to **Window** > **Preferences** > **Software AG** > **Events Development**. Click the Browse button next to Designer's NERV configuration file as defined here. In the upcoming file explorer, right-click the properties file to open it using a Text editor (e.g. Notepad++). Modify the JMS provider URL like done in the step above. Save the properties file.



    This change will affect the NERV settings of the Event Bus Console.

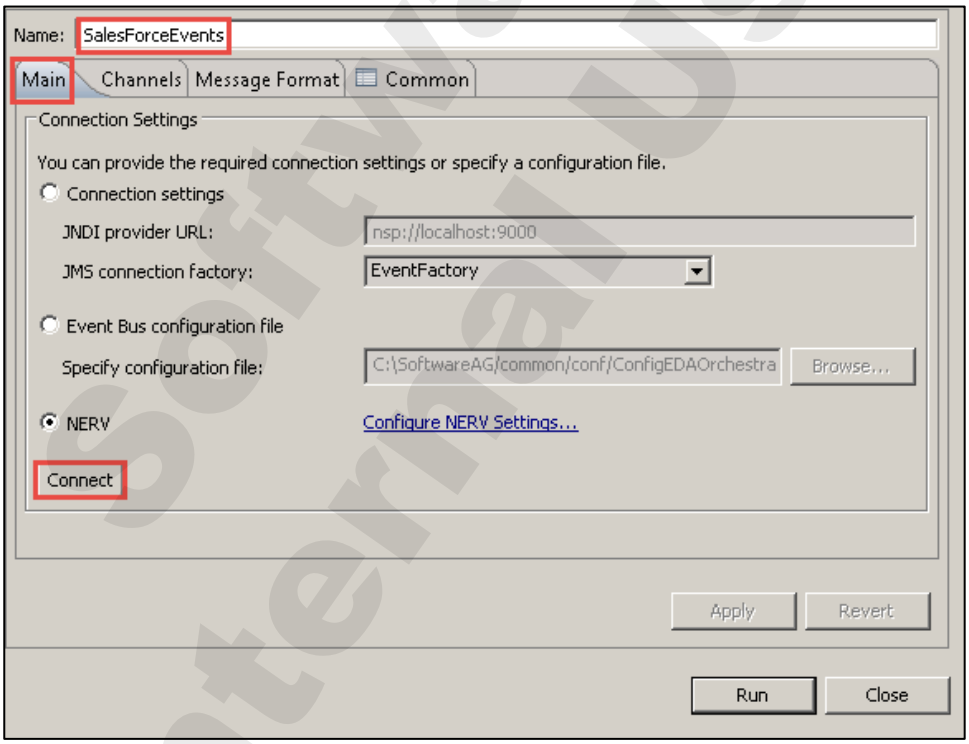
Internal Use Only! Software AG

19. Ensure both, Universal Messaging server and webMethods Broker are up and running. If not, start them.
    Note: Universal Messaging will be required also, because step transitions in your process will use JMS Connection Alias PE_NONTRANSACTIONAL_ALIAS which is still configured to access your Universal Messaging realm.



20. Test the process model **SalesForceWinProcess** once again, now using webMethods Broker for EDA communication:

    a. In Designer, switch to the **Events Development** perspective. Use the icon to open your existing Event Bus Console configuration **SalesForceEvents**.
    On the **Main tab,** hit **Connect** to reload the modified NERV configuration file.



    b. On the **Channels** tab, ensure both Event types (**SalesForceWin, SalesForceWinClassified)** are still checked.

Internal Use Only! Software AG

© 2000 – 2014 by Software AG

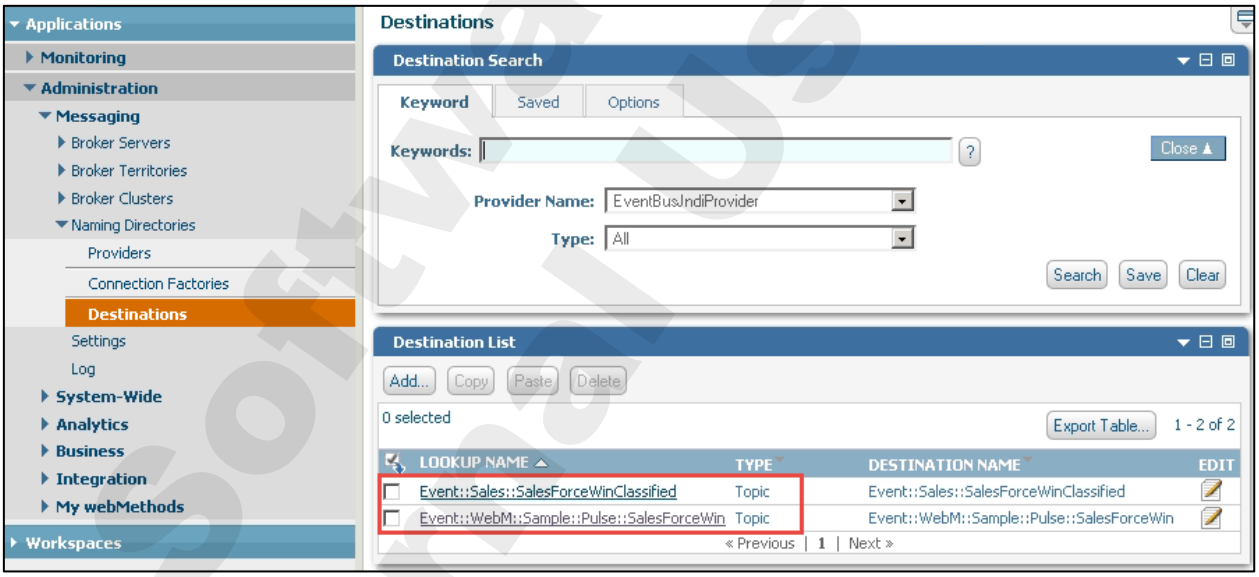
c. Hit **Apply** (if necessary) and **Run** to start the Event Bus Console listening via NERV for incoming events of both types, **SalesforceWin** and **SalesForceWinClassified.**

d. Switch to the **Process Development** perspective. From the Package Navigator, rerun the provided IS service **EDAProcessesSupport.services:emitSalesForceWinViaNERV** contained in the IS package **EDAProcessesSupport** (**Run As > Run Flow Service**). You can again use file **<*workshop-dir*>\Ex5\Resources\SalesForceWin_Input.txt** to provide the same input.

e. Switch back to the **Events Development** perspective. Use the **Console** view to look for thrown events of types **SalesForceWin** and **SalesForceWinClassified.**

21. Use a browser tab (http://localhost:8585) to login to My webMethods as **Administrator | manage**. Drill down to **Applications** > **Monitoring** > **Business** > **Process Instances** and ensure another process instance of type **SalesForceWinProcess** has been started end executed successfully.

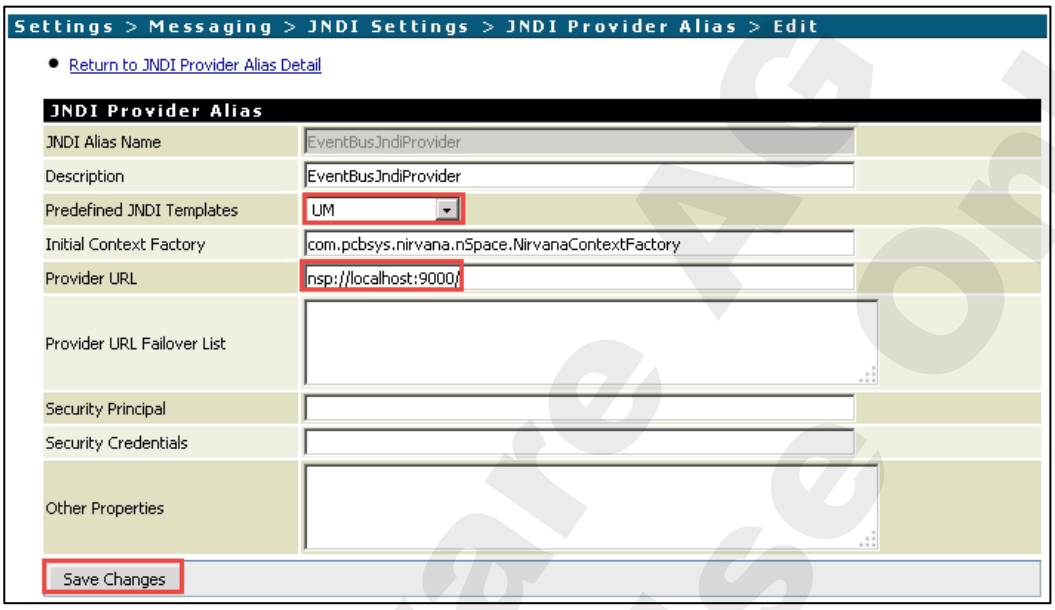
22. Switch back to My webMethods in your browser tab. Navigate to **Applications** > **Administration** > **Messaging** > **Naming Directories** > **Destinations.** The execution of your process should have added a another destination (within webMethods Broker) named **Event::Sales::SalesForceWinClassified** of type **Topic** for the throwing Intermediate Message Event:



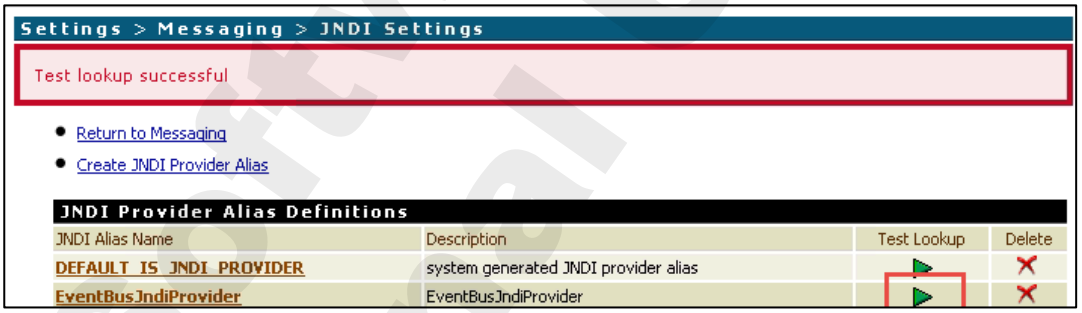
Internal Use Only! Software AG

23. Final housekeeping:
    Switch back to Universal Messaging for EDA communication. To do so:

    a. In a browser tab, login to IS Administration console (**http://localhost:5555**) as **Administrator | manage**. Navigate to **Settings** > **Messaging > JNDI Settings.** Select and **edit** the existing JNDI Provider Alias **EventBusJndiProvider.** Switch back to **UM** as predefined JNDI template and adjust the Provider URL to **nsp://localhost:9000.** Save your changes.



    b. Return to the JNDI Settings page. Perform a **Test Lookup** using your modified JNDI Provider Alias **EventBusJndiProvider.**



    c. In the IS Administration Console, navigate to **Settings** > **Messaging > JMS Settings. Disable** and **enable** the pre-configured JMS Connection Alias **EventBus** which uses your modified JNDI Connection Alias **EventBusJndiProvider**.

    d. Use a file explorer to locate file **com.softwareag.eda.nerv.properties** stored in folder **C:\SoftwareAG\profiles\IS_default\configuration\com.softwareag.platform.config.propsloader.** Use a text editor, e.g. Notepad++, to adjust the JMS provider URL used by NERV to access Universal Messaging:
    com.softwareag.eda.nerv.default.jms.provider=nsp\://localhost\:9000
    Save your changes.

    e. Repeat the previous step for file **com.softwareag.eda.nerv.properties** stored in folder **C:\SoftwareAG\common\conf** (that's the one used by the Event Bus Console in Designer).

Software AG Internal Use Only!

© 2000 – 2014 by Software AG

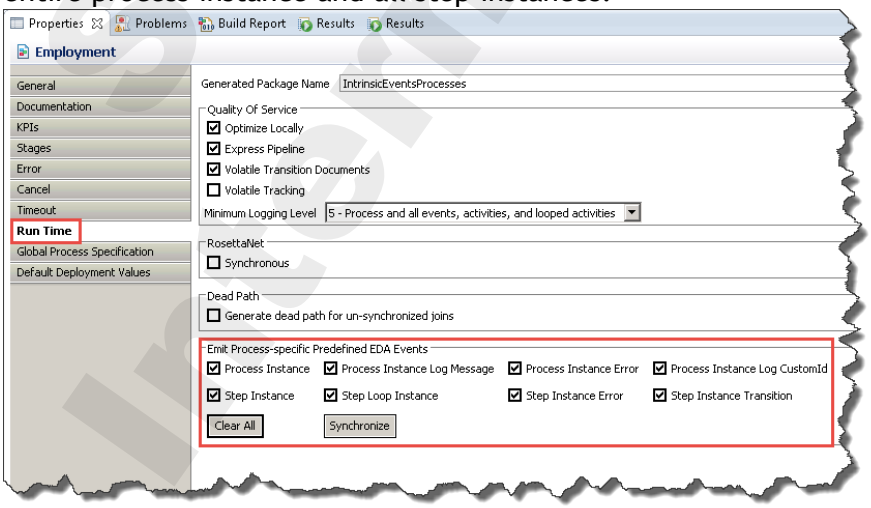
EXERCISE 6: PREDEFINED EDA EVENTS

Overview

In this exercise you configure an existing process model to emit Predefined EDA Events for the entire process as well as for every step. Moreover, Predefined EDA Events should contain business data which are Logged Fields for process steps in Designer.
Predefined EDA Events will be caught and displayed using the Event Bus Console in the Events Development perspective.
In this exercise, please replace:
<workshop_dir> by **C:\Training\602D-66E\BPM**

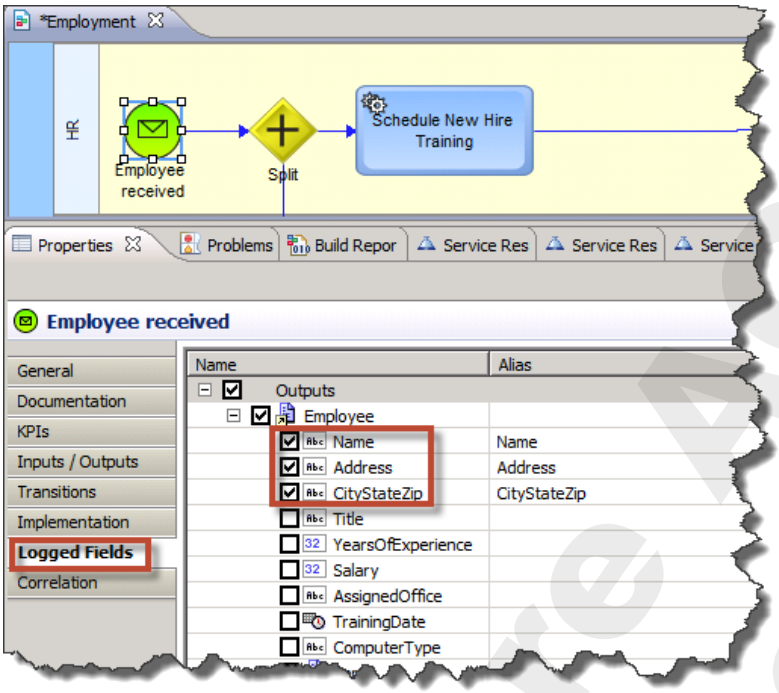
Steps

1. In a browser tab, login to the IS Administration Console (http://localhost:5555) as **Administrator** | **manage.** Navigate to **Packages** > **Management** to enable the already loaded IS package **IntrinsicEventsProcessesSupport**.
2. Start Designer using workspace **C:\Users\Administrator\workspace96**.
   Switch to the **Process Development** perspective.
   Use the Package Navigator view to push (**Sync Document Type**) the IS document structure of **intrinsicEventsProcessesSupport.docs:Employee** to the webMethods Messaging Provider (currently: webMethods Broker).
3. In Designer, use **File** > **Import** > **Existing Projects into Workspace** to import a process project from the archive file ***<workshop_dir>*\Ex6\Resources\Process Projects\IntrinsicEventsProcesses.zip** into your workspace.
4. Use the Solutions view to open the process model **Employment** contained in process project **IntrinsicEventsProcesses** in the process editor. Modify the process model as described by the following steps:
   a. Click some whitespace in the process editor to see the process properties in the Properties view. Enable the runtime properties to emit all Predefined EDA Events for an entire process instance and all step instances.



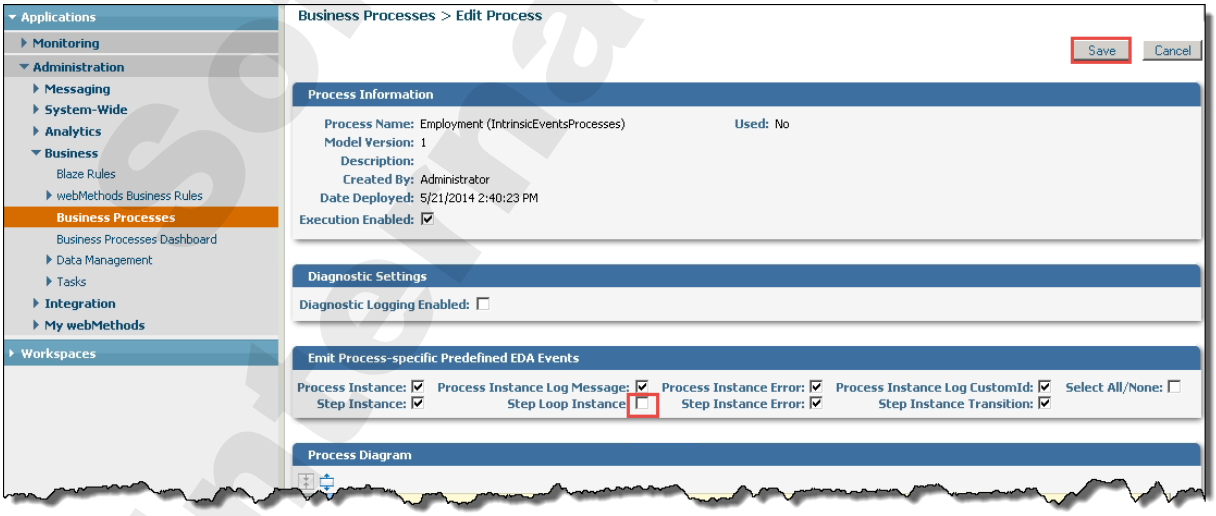
Software AG Internal Use Only!

b. Highlight Start Message Event **Employee received** in the Process editor and select the outgoing business data fields **Employee/Name**, **Employee/Address** and **Employee/CityStateZip** as **Logged Fields** in the Properties view.



c. Save, build and upload your process model. When get prompted, allow enabling the process model for execution.

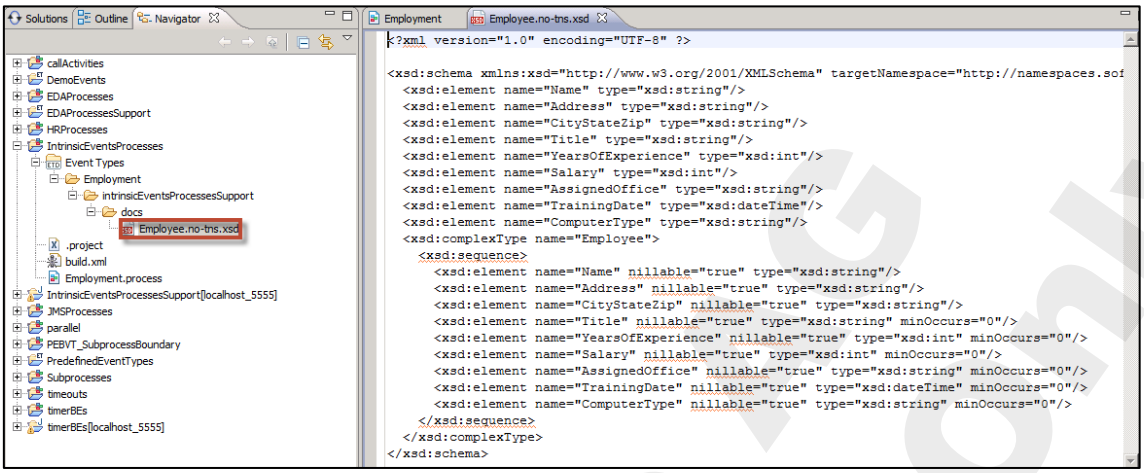
5. Use a browser tab (http://localhost:8585) to login to My webMethods as **Administrator | manage**. Drill down to **Applications** > **Administration** > **Business** > **Business Processes**. In the list of process models, click the link of process model **Employment** to see/edit its details page. Have a look for the Predefined EDA Events properties.
As your process does not contain loops, disable emitting Predefined EDA Events of type **Step Loop Instance**. Click **Save** to leave the page.



Internal use only! Software AG

© 2000 – 2014 by Software AG

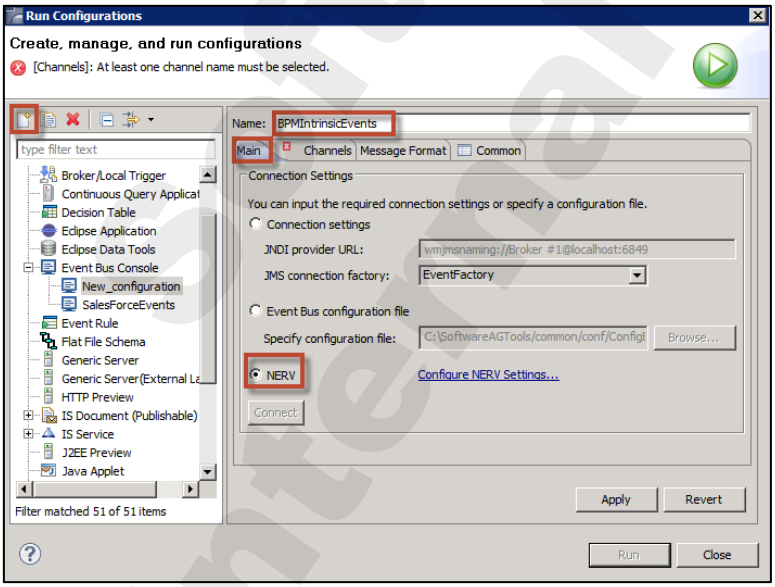
6. Switch back to Designer. Open the Navigator view. Using a text editor in Designer, open the Event Type schema **Employee.no-tns.xsd** which has been added to your process project **IntrinsicEventsProcesses** in the folder structure **Event Types/Employment/intrinsicEventsProcessesSupport/docs**.



Notes:

- The namespace/folder structure of the generated Event Type reflects the namespace/folder structure of the corresponding IS document type **intrinsicEventsProcessesSupport.docs:Employee**.
- Event Types are not displayed in the Solutions view, so the Navigator view has to be used instead.

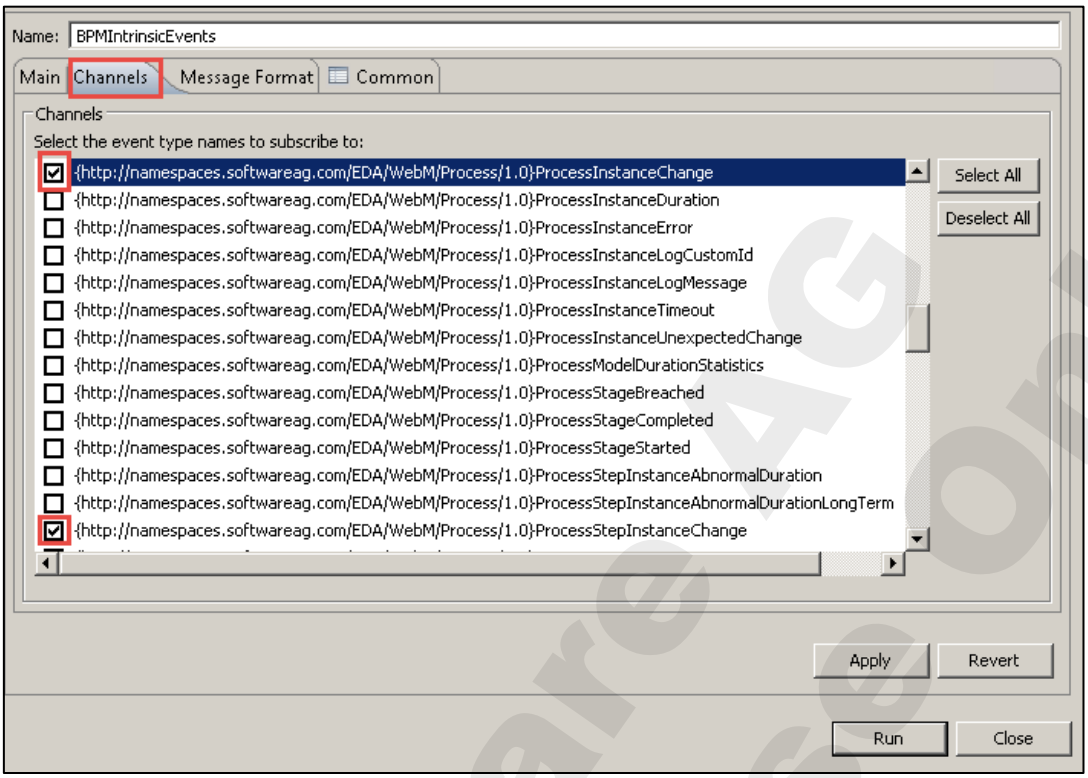
7. Switch to the **Events Development** perspective. Use the icon to open the Event Bus Console. Click to create a new Event Bus configuration. On the **Main** tab specify **BPMIntrinsicEvents** as configuration name, select NERV as access method to connect to the Event Bus and an Event Type Store. Then hit **Connect.**



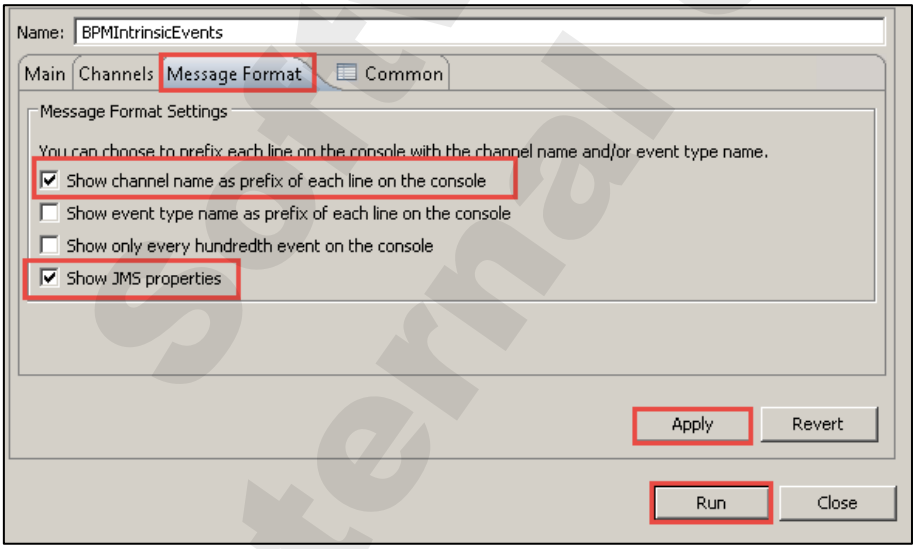
Note: Depending on the settings you made in exercise 5, NERV will access the Event Bus using Universal Messaging or webMethods Broker.

Software AG Internal Use Only!

On the **Channels** tab, check the Event Types **ProcessInstanceChange** and **ProcessStepInstanceChange** in the list of Event Types retrieved from the Event Type Store.



On the **Message Format** tab, only select **Show channel name as prefix…** and **Show JMS properties**. Hit **Apply** and **Run** to start the Event Bus Console listening via NERV for incoming events of types **ProcessInstanceChange** and **ProcessStepInstanceChange**.

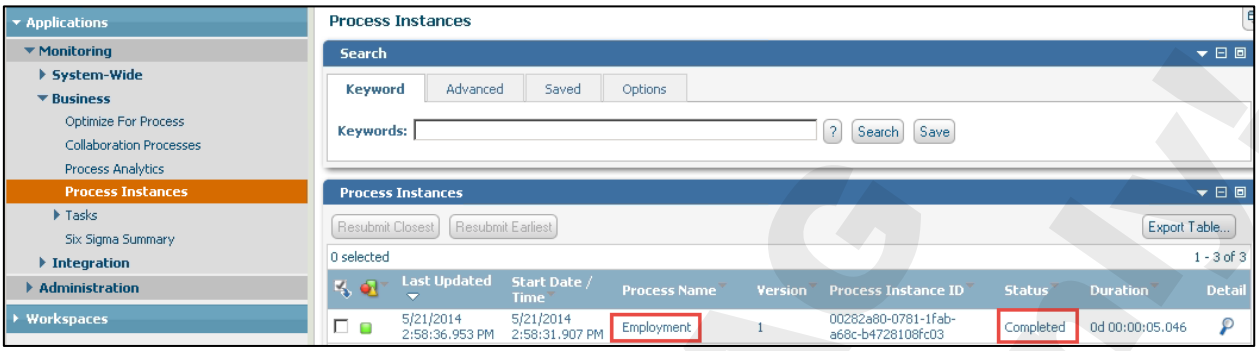


8. Switch back to the **Process Development** perspective.
   To start a process instance of your **Employment** process, run the provided service **intrinsicEventsProcessesSupport.services.util:publishEmployeeDoc** from the Package Navigator view (**Run As** > **Run Flow Service**).
   When you get prompted for the input data, load some input data from ***<workshop_dir>*\Ex6\Resources\Employee_Input.txt** and click **OK.**

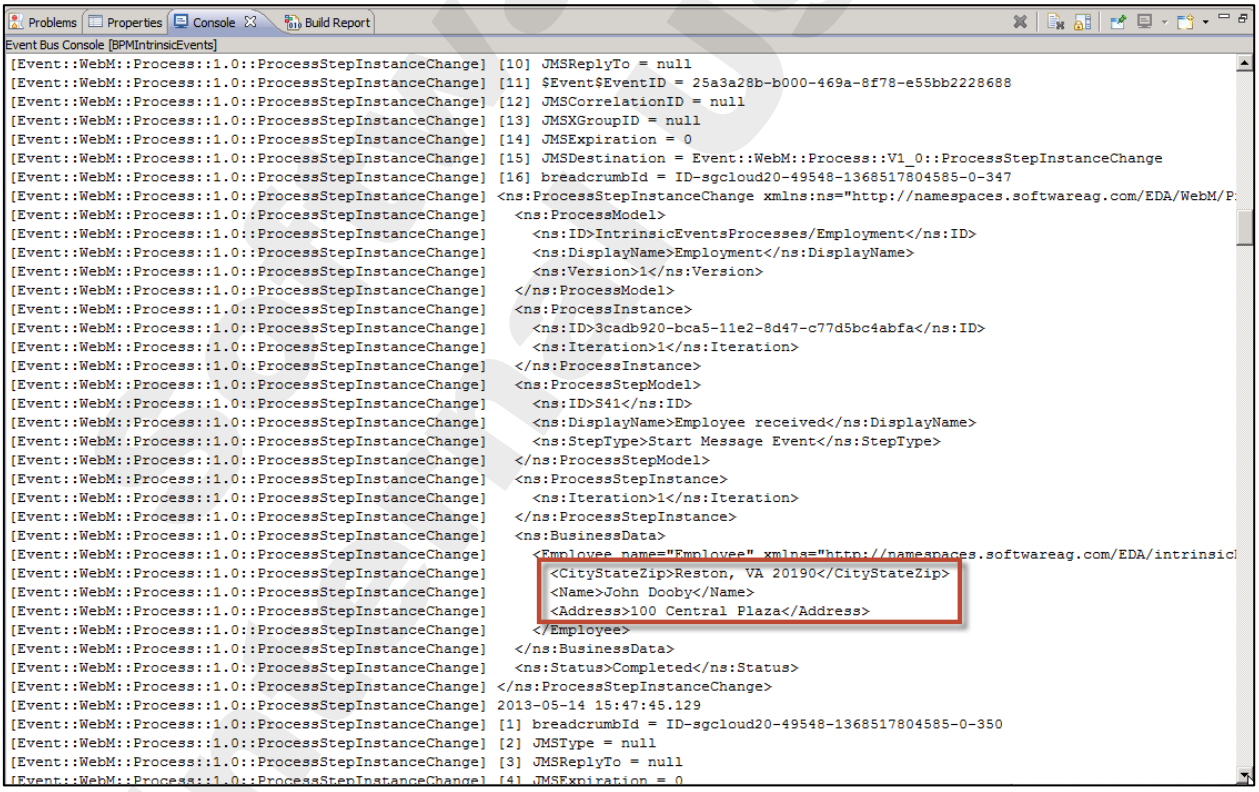
Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

9. Use the browser tab (http://localhost:8585) already connected My webMethods as **Administrator | manage**. Navigate to **Applications** > **Monitoring** > **Business** > **Process Instances** and ensure a process instance of type **Employment** has been executed successfully.

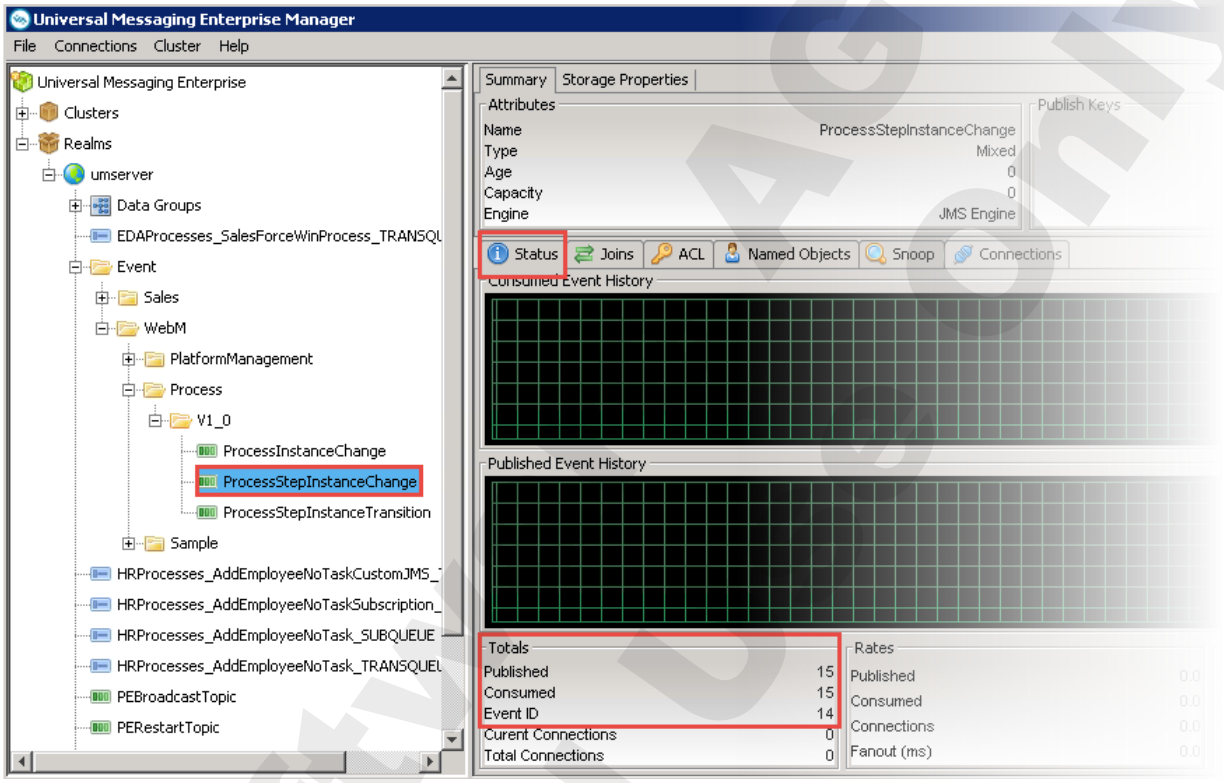


10. In Designer, go back to the **Events Development** perspective. Use the Console view and have a look for caught Predefined EDA Events. Ensure Console shows two Predefined EDA Events of type **ProcessInstanceChange** (started, completed) and additionally two Predefined EDA Events of type **ProcessStepInstanceChange** (started, completed) for each step in the process**.**
Moreover you should find the logged fields in the completed **ProcessStepInstanceChange** events of the corresponding steps that perform logging. You can right-click into the console view and choose **Find/Replace…** to search for e.g. business data of type **Address**.



Internal use only

11. *For extra credit:*
    If NERV and IS is configured to use Universal Messaging for Event Bus access (default), start Universal Messaging Enterprise Manager (**Start** > **All Programs** > **Software AG** > **Administration** > **Universal Messaging Administration 9.x** > **umserver** > **Enterprise Manager**).
    For realm **umserver**, locate topics corresponding to the Predefined EDA Event types (**Event** > **WebM** > **Process** > **V1.0**). Highlight topic **ProcessStepInstanceChange** and open the Status tab. You should see the number of Events emitted by the Process Engine via EDA and handled by Universal Messaging.



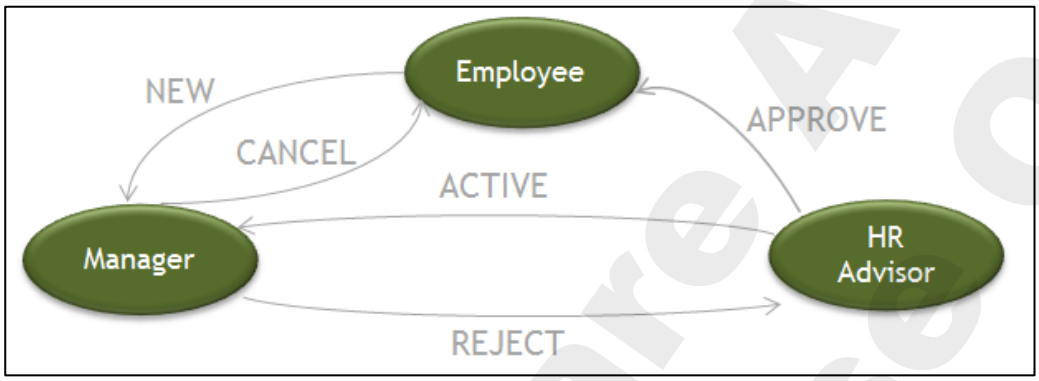
Software AG Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 7: OPEN PRESENTATION LAYER

Overview

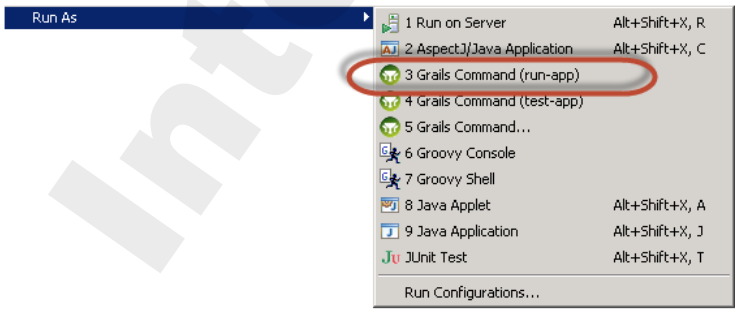
In this exercise you will use the RESTful Web services of the Task Engine API. A sample User Task Type "ApplyLeave" is used in MWS. By updating the "appstatus" business data field (inside TaskData), a User Task instance moves from the employee to the manager, to the HR advisor, and back to the employee.



A sample UI application called "taskapp" and implemented in Grails is available. This application communicates with the Task Engine through its RESTful Web services API. To learn how the API works, you will send REST requests from a simple REST client which is available as add-on in Firefox.
In this exercise, please replace: ***<workshop_dir>*** by **C:\Training\602D-66E\BPM**

Steps

1. Use the **taskapp** Grails application to create three new tasks of type **ApplyLeave**.
   To do so, perform the following steps:
   a. Ensure your Integration Server and MWS are up and running.
   b. Start the Groovy/Grails Tool Suite (GGTS) from the Start menu (**All Programs** > **Tools** > > **Groovy-Grails** > **Groovy-Grails Tool Suite**) . Accept the default workspace.
   c. Select the **taskapp** application in the Project Explorer and, from the context menu, use the command **Run As**> **Grails Command (run-app)** to start the application on the internal server.
   Note: Accept warnings about incompatible Groovy compiler versions in the Console view.

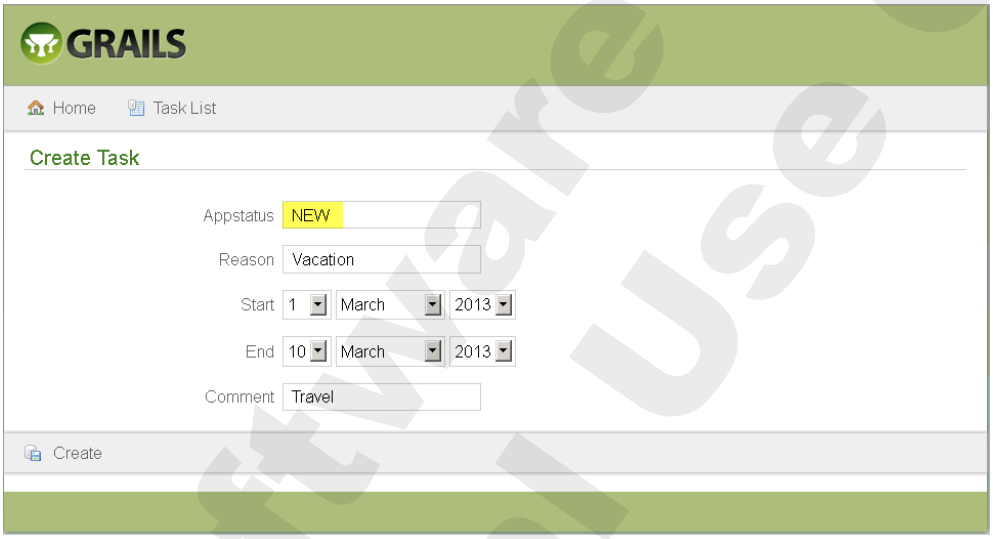


Software AG Internal Use Only!

d. When the internal server is up and running (Console view), use a browser tab to connect to the application using the URI http://localhost:8080/taskapp/user.

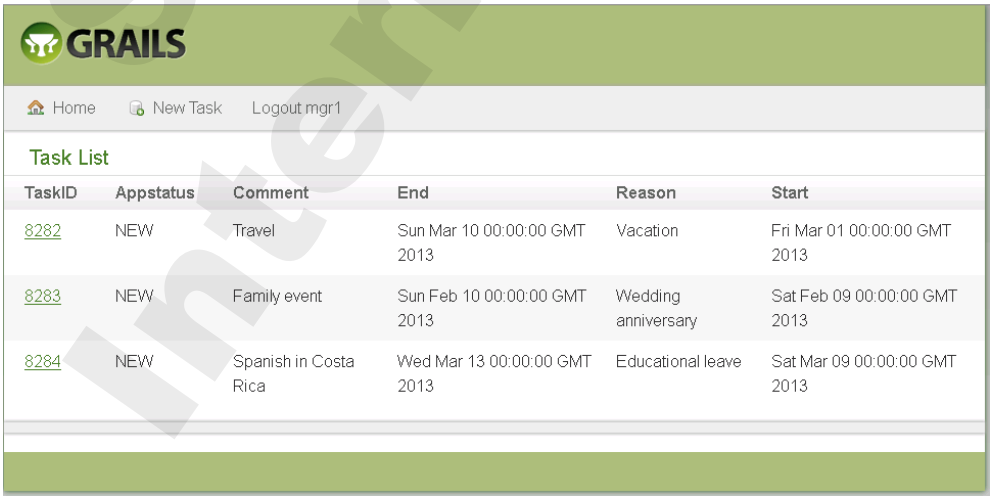
e. **Login** as employee (username **emp1**, password **manage**). An empty task list is displayed.

f. Use the **New Task** button to create a new User Task instance. Type in/select some task data as shown below. Field **Appstatus** will be filled/ overwritten automatically, so you don't have to provide a value. Finally hit the **Create** button to save your data.



g. Create two more new tasks before you **logout** as emp1.

h. Now, **login** as manager (username **mgr1**, password **manage**). All three tasks you created before should appear in the Task List.

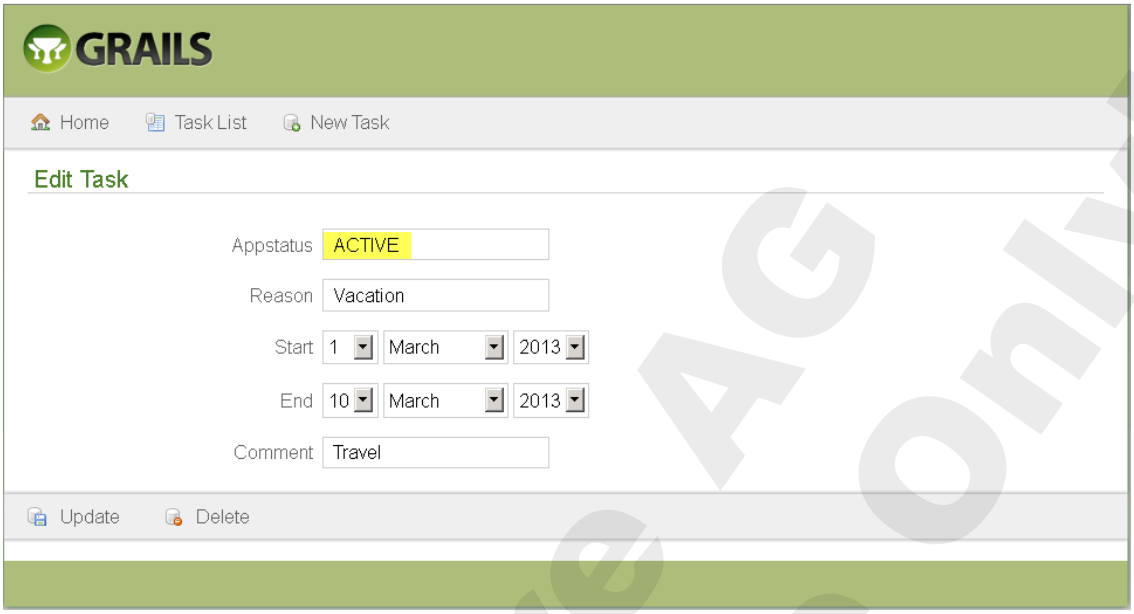


TaskID	Appstatus	Comment	End	Reason	Start
8282	NEW	Travel	Sun Mar 10 00:00:00 GMT 2013	Vacation	Fri Mar 01 00:00:00 GMT 2013
8283	NEW	Family event	Sun Feb 10 00:00:00 GMT 2013	Wedding anniversary	Sat Feb 09 00:00:00 GMT 2013
8284	NEW	Spanish in Costa Rica	Wed Mar 13 00:00:00 GMT 2013	Educational leave	Sat Mar 09 00:00:00 GMT 2013

Software AG Internal Use Only!

© 2000 – 2014 by Software AG

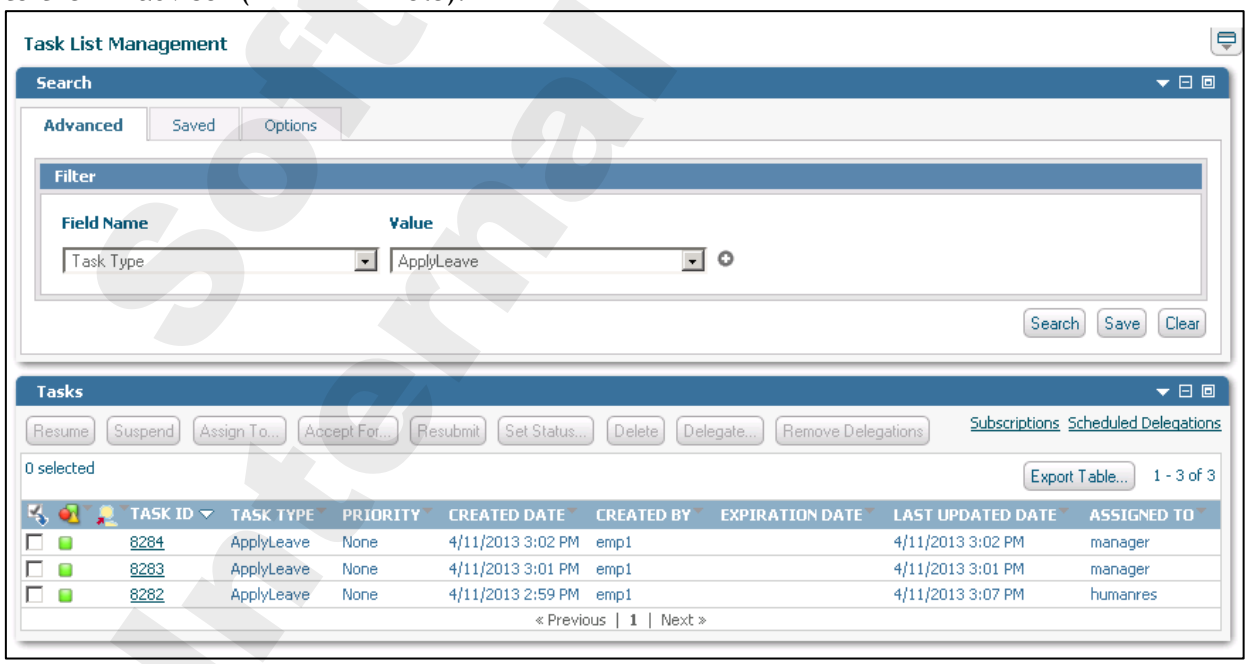
i. Open one of the tasks by clicking its Task ID link and click the **Edit** button at the end of the Show Task page. Forward the task to the HR advisor by setting the **appstatus** field to **ACTIVE**. Save your changes by clicking on the **Update** button of the Edit Task page.



j. The Task List for the manager should now contain one entry less (i.e. two entries).

k. **Logout** as manager and **login** as HR advisor (username **hr1**, password **manage**) and verify that the Task List contains the forwarded task.

2. In a browser tab, connect to My webMethods and login as **Administrator**| **manage**. Navigate to **Applications** > **Monitoring** > **Business** > **Tasks** to open the **Task List Management** page. **Search** for the three new tasks of type **ApplyLeave**. Verify that one of the tasks is assigned to the HR advisor (**humanres** role).



Software AG Internal Use Only!

3. Use the **RESTClient** for Firefox to test the Task Engine **RESTful** web services.
   To do so, perform the following steps:

   a. Start Firefox. In Firefox, click the icon to start the **RESTClient** plug-in.

   b. Send a REST service request to retrieve the task inbox of the manager. The response should list the two tasks assigned to the manager, including the TaskInfo part for each (but not the TaskData part). Make sure that you obtain a **JSON** response.

      Details:

      i. The request method is **GET**.

      ii. The URL to call the RESTful service is of the form:
      http://localhost:5555/rest/pub/opentask?<*input_parameter*>

      iii. Specify **searchUserTasks=true** and **user=mgr1** input parameters (separated by '&') to retrieve tasks assigned to the manager only.

      iv. Add a request header **Accept: application/json** to obtain a JSON response.

      v. Send the request.

      vi. After sending the request, you might be asked to authenticate to Integration Server. Provide **Administrator** | **manager** as user credentials.

      The JSON response should be of the following form (Response Body (Preview)):

```
{
 "Tasks": [
  {
   "TaskID": "8283",
   "TaskInfo": {
    "name": null,
    ...
    "assignedToList": [
     "manager"
    ],
    ...
   }
  },
  {
   "TaskID": "8284",
   "TaskInfo": {
    "name": null,
    ...
    "assignedToList": [
     "manager"
    ],
    ...
   }
  }
 ]
```

Note: If needed, use **File** > **Load request** to load the file **readTaskList.txt** from folder ***<workshop_dir>*\Ex7\Solutions** for this exercise into the RESTClient.

Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

c. Send another REST service request to retrieve the details for one of your tasks in the manager's inbox. In this text we use the example **TaskID=8284**. The response should include the TaskInfo as well as the TaskData parts for it. Make sure that you obtain a JSON response.

Details:

i. The request method is **GET**.

ii. The URL to call the RESTful service is of the form:
http://localhost:5555/rest/pub/opentask/8284?<*input_parameter*>
(where **8284** is the ID of the task to retrieve, replace with your own value)

iii. Add the input parameter **includeTaskData=true** to obtain the task's business data.

iv. Add a request header **Accept: application/json** to obtain a JSON response.

v. Send the request.

The JSON response should be of the following form (Response Body (Preview)):

```
{
  "TaskID": "8284",
  "TaskInfo": {
    "name": null,
    ...
    "assignedToList": [
      "manager"
    ],
    ...
  },
  "TaskData": {
    "end": "Wed Mar 13 00:00:00 GMT 2013",
    "appstatus": "NEW",
    "reason": "Educational leave",
    "start": "Sat Mar 09 00:00:00 GMT 2013",
    "comment": "Spanish in Costa Rica"
  }
}
```

Note: If needed, use **File** > **Load request** to load the file **readTaskDetails.txt** from folder **<*workshop_dir*>\Ex7\Solutions** for this exercise into the RESTClient. Of course you have to adapt the Task ID to a valid Task ID return by your first request.

d. Next, send a REST request to forward the task you retrieved in the previous step (for example **TaskID=8264**) to the HR advisor. Remember that, to do so, you have to update the **appstatus** field to **ACTIVE**.

Details:

i. The request method is **PUT**.

ii. The URL to call the RESTful service is of the form:
http://localhost:5555/rest/pub/opentask/8284?<*input_parameter*>
(where **8284** is the ID of the task to update)

iii. Add the input parameter **user=mgr1** to execute the service on behalf of the manager.

Software AG Internal Use Only!

iv. Copy and paste the entire JSON response from the raw data you obtained in the previous step c) into the request body field of the PUT request. Change the **appstatus** value to **ACTIVE**.

v. Add the request headers **Content-Type: application/json** and **Accept: application/json** to send and receive JSON.

vi. Send the request.

If the operation was successful, the JSON response is just the ID of the updated task, e.g.:

```
{"taskID": "8284"}
```

Note: If needed, use **File** > **Load request** to load the file **updateTask.txt** from folder **<*workshop_dir*>\Ex7\Solutions** for this exercise into the RESTClient. Of course you have to adapt the Task ID, paste the entire JSON response from the raw data you obtained in the previous step, and change the **appstatus** value to **ACTIVE**.

e. As you did before, use the Task List Management page in MWS to verify that the updated task is now assigned to the HR advisor, too (**humanres** role).

4. Refer to the documentation (**9-x_Task_Engine_API_and_Service_Reference.pdf**) for a complete list of the available Task Engine RESTful services.

For extra credit, try to do using RESTClient:

a. Complete the task you have updated in the previous step.
Hint: Set the **status** field (inside TaskInfo) to **completed**.

b. Delete the task.
Hint: The request method is **DELETE.**

c. Create (i.e. queue) a new task instance of type **ApplyLeave**.

Hint: The request method is **POST**. The request body should contain at least TaskTypeID and TaskData. A sample request body is:

```
{
  "taskTypeID":"E2DBE03B-46BA-C9BB-22E9-6C521C540CE9",
  "taskData":{
    "appstatus":"NEW",
    "class":"taskapp.TaskData",
    "comment":"Nothing",
    "end":"Mon Mar 11 00:00:00 GMT 2013",
    "reason":"My reason",
    "start":"Fri Mar 01 00:00:00 GMT 2013"
  }
}
```

d. POST a complex task search request.

Note: If needed, use **File** > **Load request** to load the files **completeTask.txt**, **deleteTask.txt** and **createTask** from folder **<*workshop_dir*>\Ex7\Solutions** for this exercise into the RESTClient.

Software AG Internal Use Only!

 © 2000 – 2014 by Software AG

Exercise 8a: Setup and test a BPM Process and User Task for Mobile Task Integration

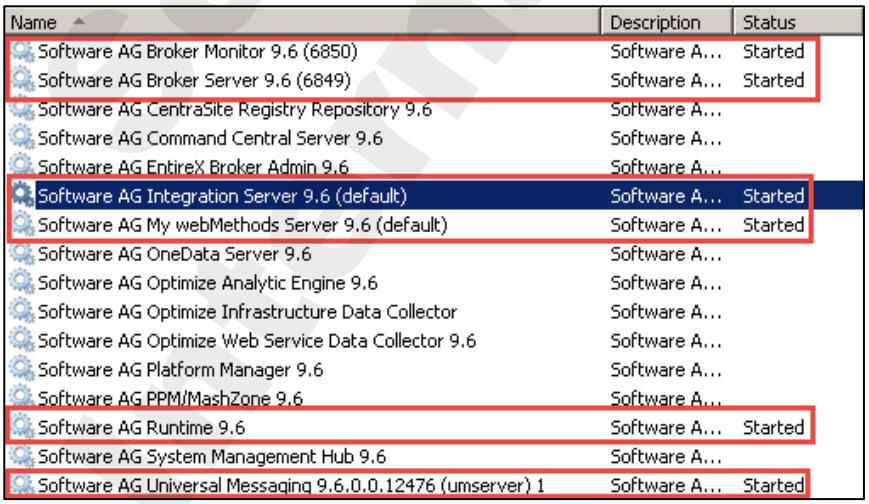
Objectives

In this exercise you will import a preconfigured process model that contains a User Task invocation. After performing some security definitions and small Task adaptions you will test the deployed project by running an IS service. You will work on User Tasks which are assigned to a specific role.

In this exercise, please replace:
<workshop_dir> by **C:\Training\602D-66E\BPM**

Steps

1. Use the Windows Service Control Panel and ensure the following Windows Services are up and running. If not, start them in the following sequence:
   - **Software AG Runtime 9.6**
   - **Software AG Broker Monitor 9.6**
   - **Software AG Broker Server 9.6**
   - **Software AG Universal Messaging 9.6 (umserver)**
   - **Software AG Integration Server**
   - **Software AG My webMethods Server 9.6 (default)**
   - **hMailServer**

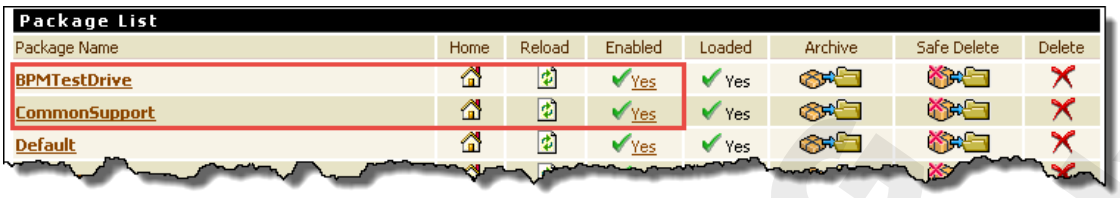


Name	Description	Status
Software AG Broker Monitor 9.6 (6850)	Software A...	Started
Software AG Broker Server 9.6 (6849)	Software A...	Started
Software AG CentraSite Registry Repository 9.6	Software A...	
Software AG Command Central Server 9.6	Software A...	
Software AG EntireX Broker Admin 9.6	Software A...	
Software AG Integration Server 9.6 (default)	Software A...	Started
Software AG My webMethods Server 9.6 (default)	Software A...	Started
Software AG OneData Server 9.6	Software A...	
Software AG Optimize Analytic Engine 9.6	Software A...	
Software AG Optimize Infrastructure Data Collector	Software A...	
Software AG Optimize Web Service Data Collector 9.6	Software A...	
Software AG Platform Manager 9.6	Software A...	
Software AG PPM/MashZone 9.6	Software A...	
Software AG Runtime 9.6	Software A...	Started
Software AG System Management Hub 9.6	Software A...	
Software AG Universal Messaging 9.6.0.0.12476 (umserver) 1	Software A...	Started

Important: These services should be up and running for all subsequent exercises.

Software AG Internal Use Only!

2. Use a browser tab and URL http://localhost:5555 to open the Integration Server Administration console. Use **Administrator** | **manage** for authentication.
   In the IS Administration console visit **Packages** > **Management**. Ensure that packages **BPMTestDrive** and **CommonSupport** are enabled. If disabled, click the appropriate link in the **Enabled** column to enable.

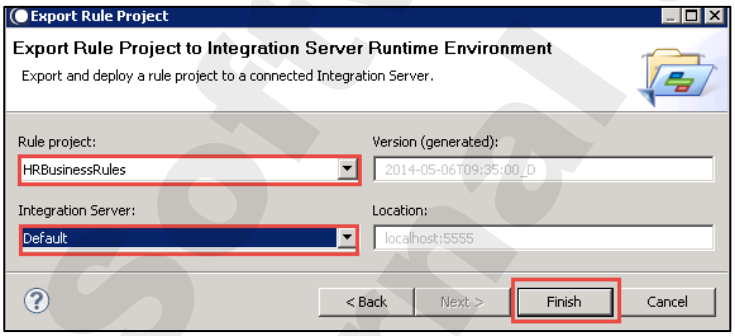


3. Launch Software AG Designer (**Start** > **All Programs** > **Software AG Tools** > **Tools** > **Software AG Designer**). Accept the default workspace. Import prepared assets into your workspace and export/publish them to their corresponding runtime environment:

   a. Right-click the existing process project **HRProcesses** in the Solution view and select **Import**. In the Import window, select **Software AG** > **Process File** and click **Next**. Browse for file **<*workshop_dir*>\Ex8a\Resources\AddEmployee.process** and use the selected project **HRProcesses** as target. Click **Finish**.

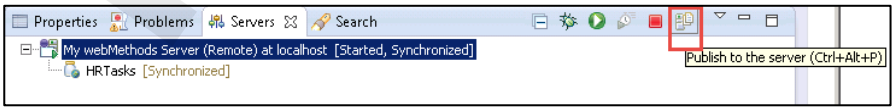
   b. In the **File** menu, click **Import**. In the Import window, click **General** > **Existing Projects into workspace**. Click **Next**. Choose option **Select archive file** and browse to the zipped Business Rules project **<*workshop_dir*>\Ex8a\Resources\HRBusinessRules.zip**. Select the archived file and click **Open**. Click **Finish**.

   c. In the **File** menu, click **Export**. In the Export window, select **Software AG** > **Rule Projects to Integration Server runtime**. On the Export Rule Project panel, select (imported) Rule project **HRBusinessRules** and Integration Server **Default**. Hit **Finish**.



   d. In the **File** menu, click **Import**. In the Import window, click **Software AG** > **Existing CAF Projects into workspace**. Hit **Next**. Choose option **Select archive file** and browse to the zipped User Task project **<*workshop_dir*>\Ex8a\Resources\HRTasks.zip**. Select the archived file and click **Open**. Click **Finish**.

   e. In Designer, switch to the **UI Development** perspective. In the **Servers** view, right-click **My webMethods Server** and add User Task project **HRTasks** to your running MWS. Use the **Publish** icon to publish your User Task project to MWS. If asked for authentication, provide **Sysadmin** | **manage**.

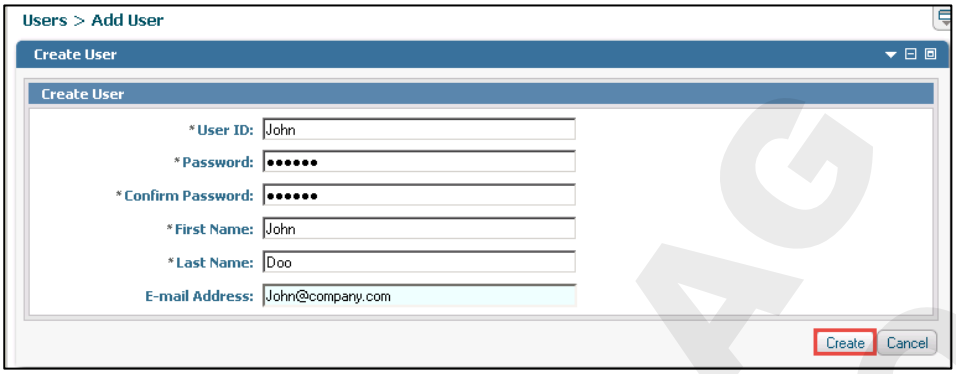


Software AG Internal use only!

© 2000 – 2014 by Software AG

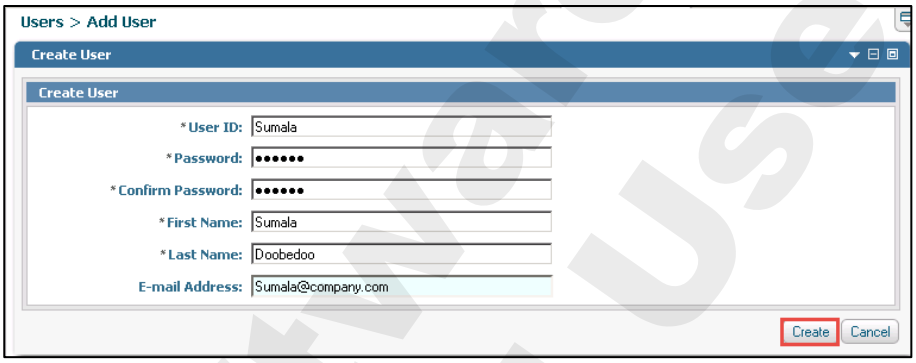
4. Use a browser tab to login to My webMethods (http://localhost:8585) as Administrator |manage.

   a. In My webMethods, navigate to **Administration** > **System-Wide** > **User Management** > **Users**. Add a user with User ID **John**, password **manage** and the following details:



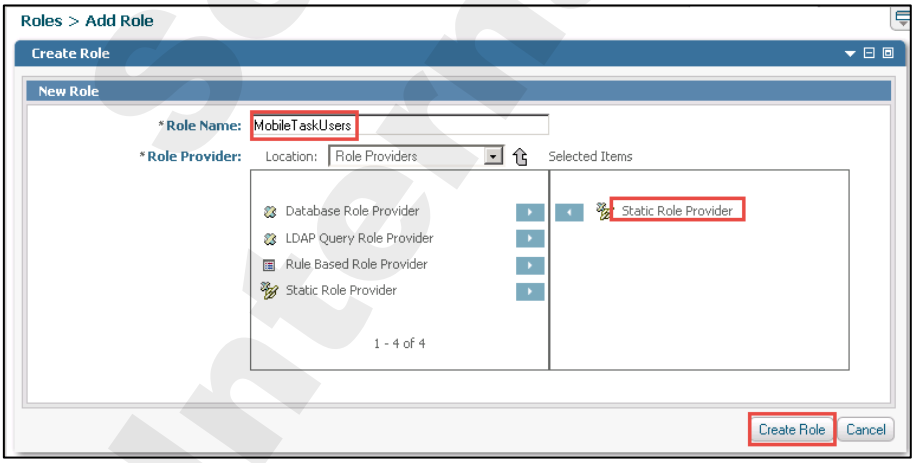
   Click **Create** and **Save**.

   b. Add another user with User ID **Sumala**, password **manage** and the following details:



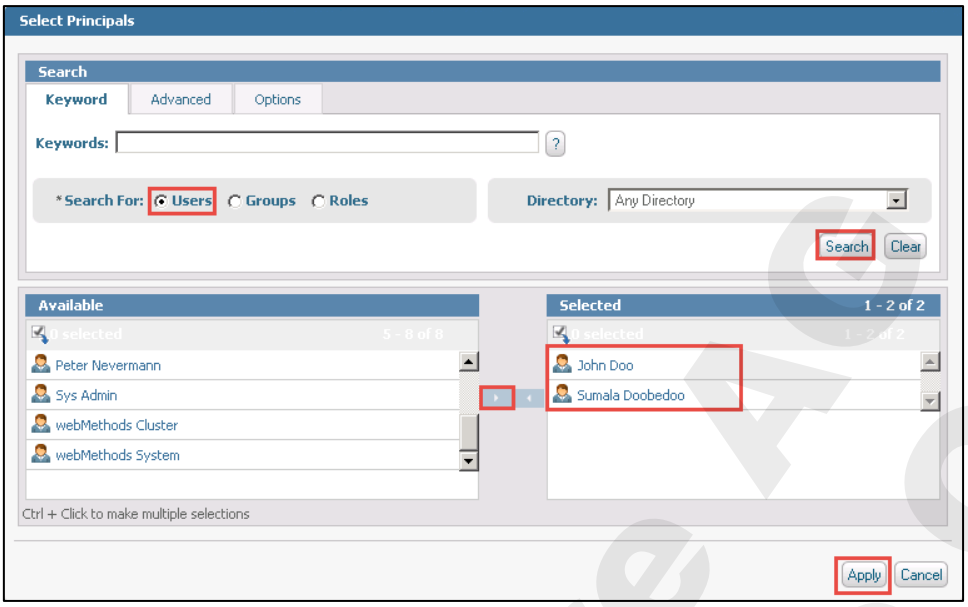
   Click **Create** and **Save**.

5. In My webMethods, navigate to **Administration** > **System-Wide** > **User Management** > **Roles**. Add a role named **MobileTaskUsers** with a **Static Role Provider**. Click **Create Role**.



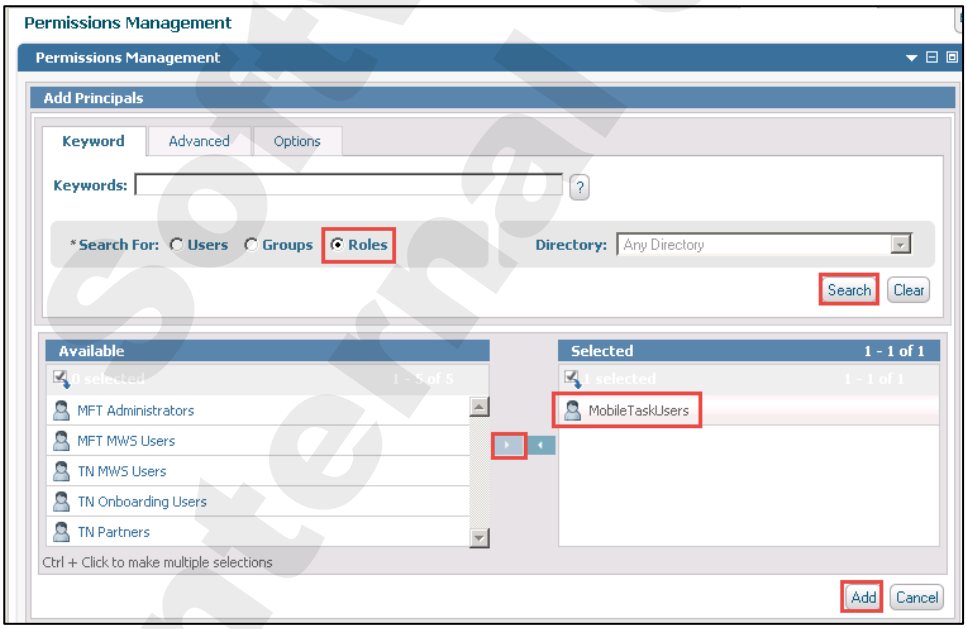
Software AG Internal Use Only!

On the Role Attributes portlet, select the **Members** tab. Click **Edit Members…** , search for users and pick users **John Doo** and **Sumala Doobedoo** as principals belonging to that role. Hit **Apply** and **Save**.



6. In My webMethods, navigate to **Administration** > **System-Wide** > **Permissions Management**.

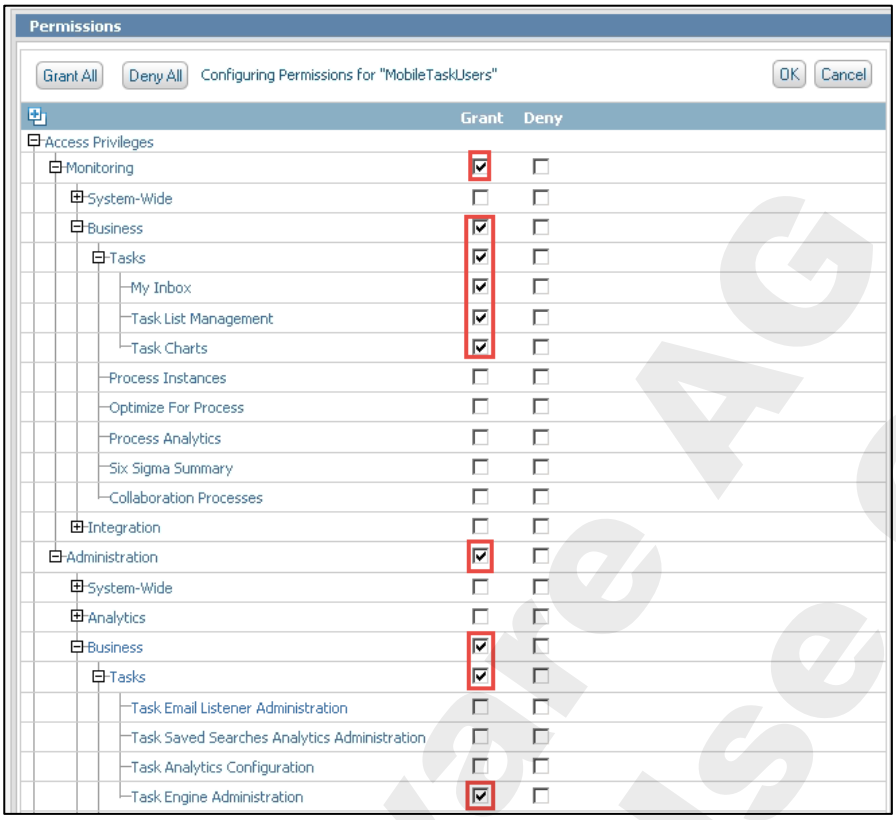
   a. On the Permissions Management page, select Resource Type **webMethods Applications**, search and select **My webMethods Applications**, then hit **Next**.
   On the next panel click **Add**. Search and select role **MobileTaskUsers** as principal and click **Add**.



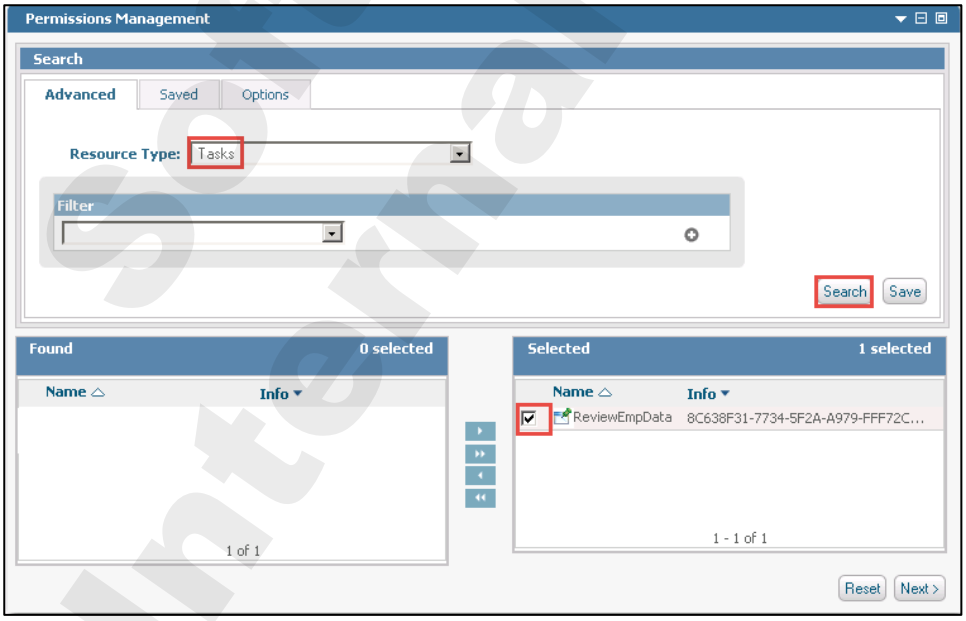
Software AG Internal Use Only!

© 2000 – 2014 by Software AG

On the next panel, grant the following Task-related access privileges and hit **OK**. Then click **Apply**.



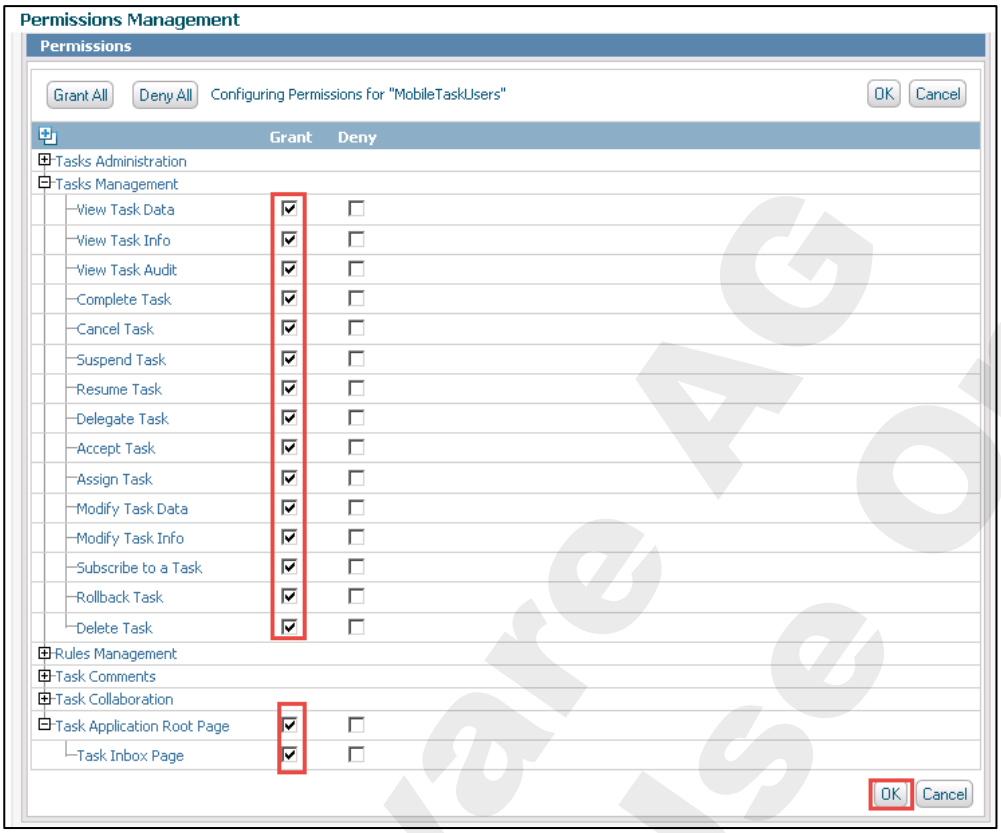
b. On the initial Permissions Management page, select Resource Type **Tasks**, hit **Search**, select and check User Task type **ReviewEmpdata**. Hit **Next**.



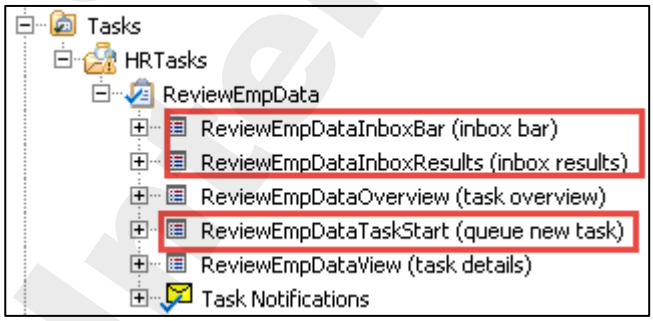
On the next panel click **Add**. Search and select role **MobileTaskUsers** as principal and click **Add**.

Software AG Internal Use Only!

On last panel, grant all Task Management- and Task Application Root Page-related permissions. Hit **OK**, then click **Apply**.



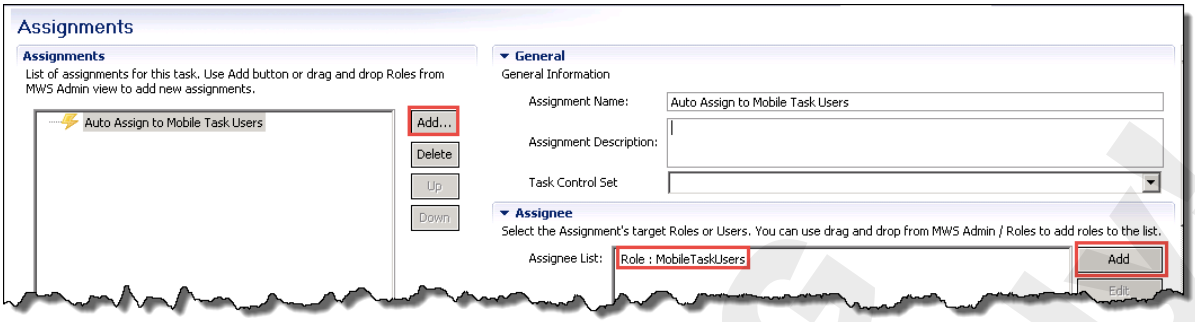
7. Enhance your imported User Task in Designer:
   a. In Designer, switch to the **UI Development** perspective.
   b. Use the Solutions view to ensure your User Task type additionally contains the following Task type-specific portlets:
      - **<*Task Type*>InboxBar**
      - **<*Task Type*>InboxResults**
      - **<*Task Type*>TaskStart**



Software AG Internal Use Only!

© 2000 – 2014 by Software AG

c. In the Task editor, open the **Assignments** tab. Add an assignment named **Auto Assign to Mobile Task Users**. For the assignment, add MWS role **MobileTaskUsers** as assignee.

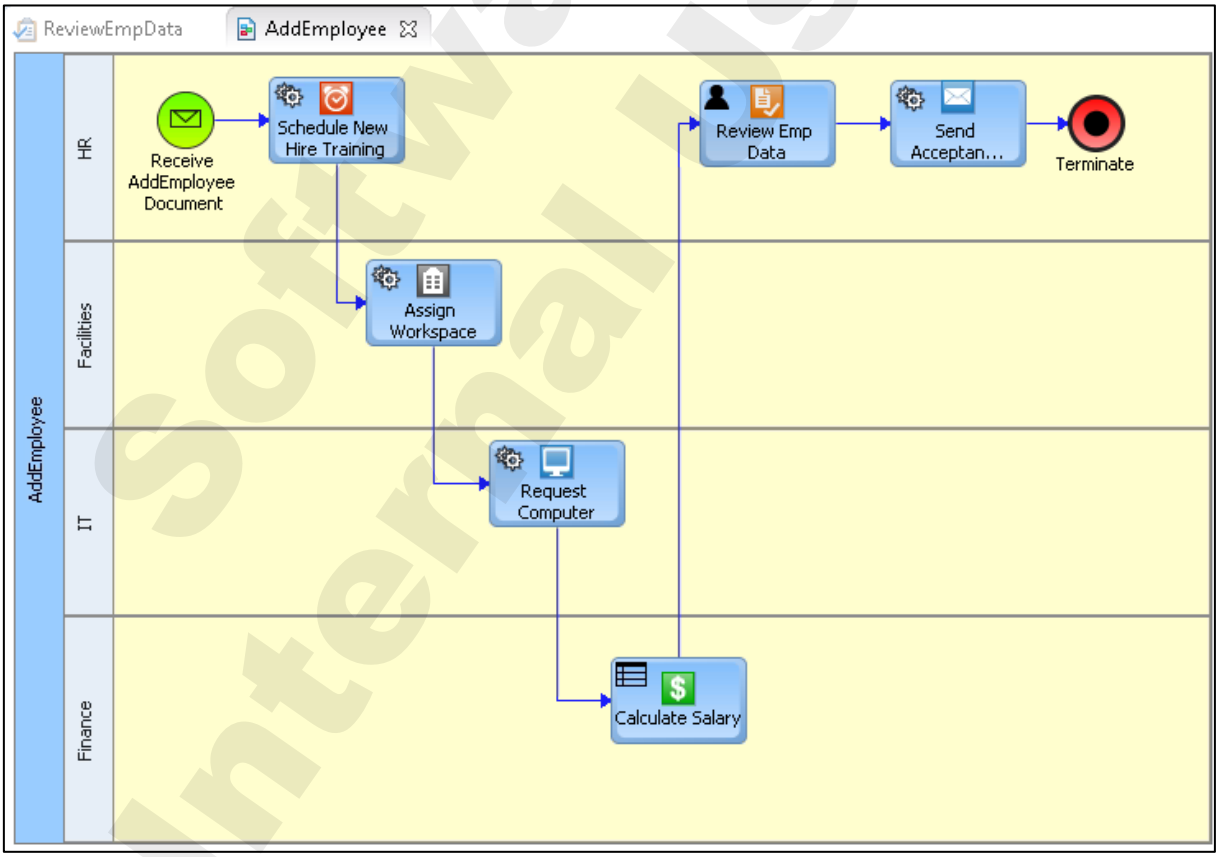


d. For later test purposes, additionally add My webMethods user **Administrator** to the list of assignees, too (this is to allow Administrator to see User Tasks of this type in his (Mobile) Task Inbox also). Hint: You can eiter key the name in or drag it from the MWS Admin view.

e. Leave all other assignment settings unchanged and save your definitions.

8. Use the Servers view in Designer to re-publish your modified Task project to MWS.
   If asked for authentication, provide **Sysadmin** | **manage**.

9. Switch to the **Process Development** perspective and open process model **AddEmployee** as contained in process project **HRProcesses** from the Solutions view in the process editor.



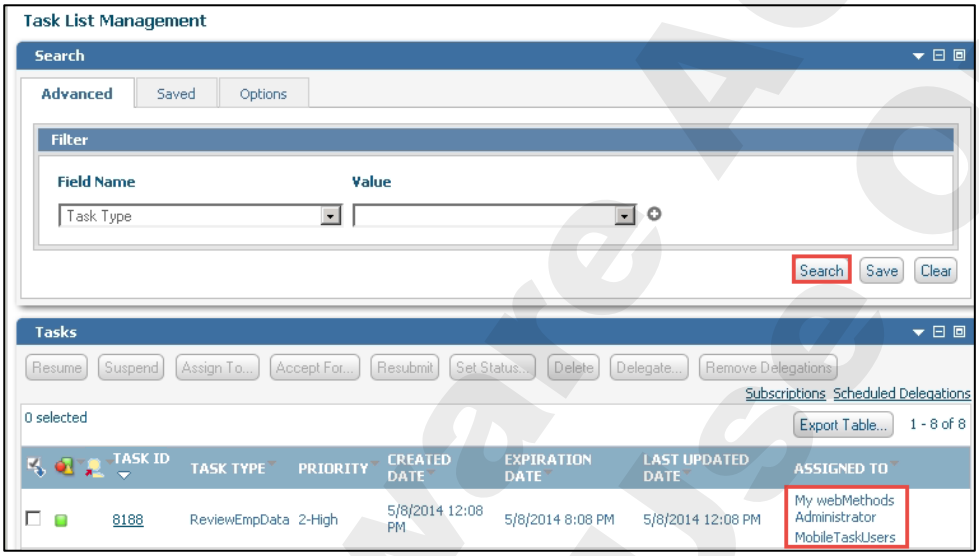
10. Build and upload ( ) process model **AddEmployee**. When get prompted, allow enabling the process model for execution.

Software AG Internal Use Only

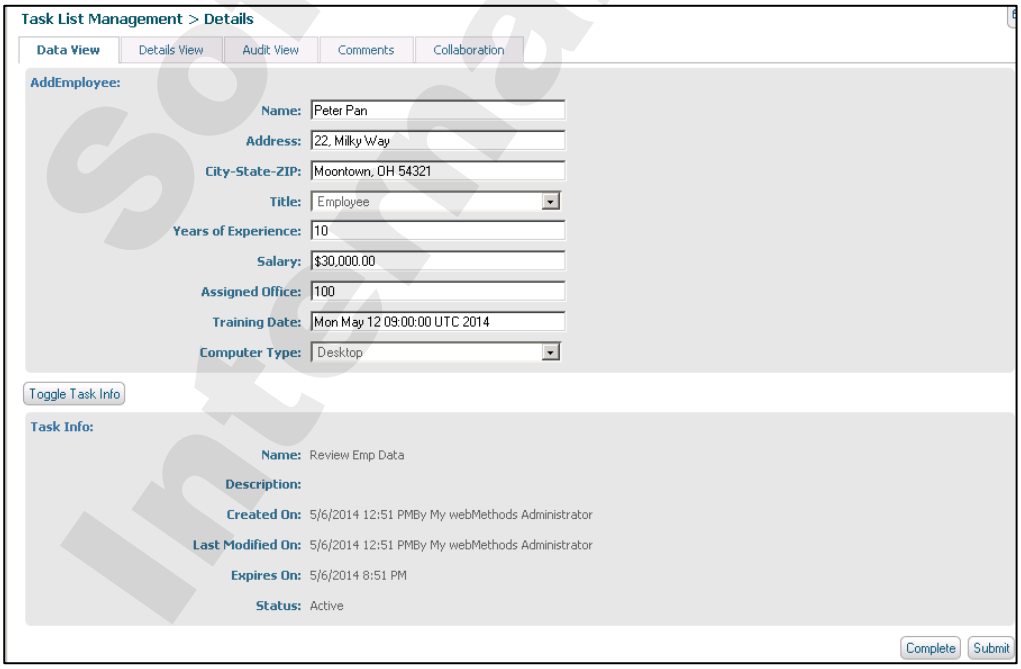
11. Test your process:

    a. In Designer, right-click IS service **bpmTestDrive.services.util:sendAddEmployeeDoc** in the Package Navigator view and select **Run As** > **Run Flow Service**.
    When get prompted for employee input data, load data from file ***<workshop_dir>*\Ex8a\Resources\ProcessInput.txt** and hit **OK**.

    b. Use a browser tab to login to My webMethods (**http://localhost:8585**) as **Administrator | manage**. Drill down to **Applications** > **Monitoring** > **Business** > **Tasks** > **Task List Management**. Click **Search** in the Search portlet. You should see a new User Task of Task type **ReviewEmpData** in the Tasks portlet. Ensure it is assigned to role **Mobile Task Users** and user **My webMethods Administrator**:



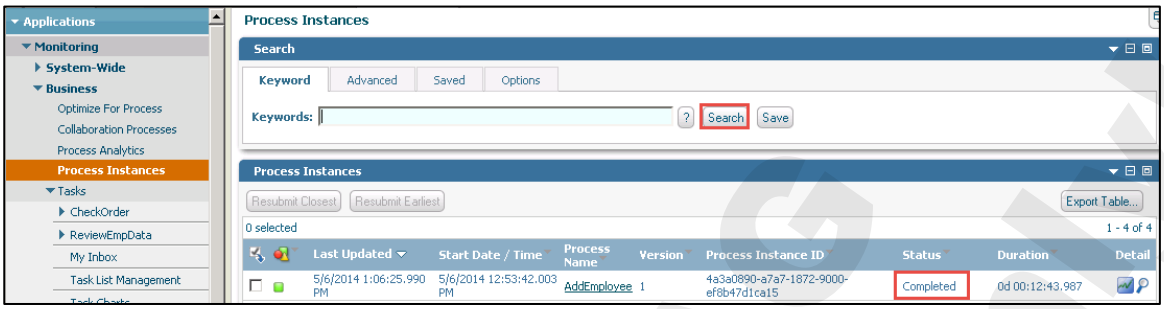
    c. Click on the Task ID link to open the User Task's data view. Verify all business data are displayed. Fields Salary, Assigned Office Training Date and Computer Type should contain data created by the invoked IS services and the business rule. **Accept** and **Complete** the User Task.



Software AG Internal use only!

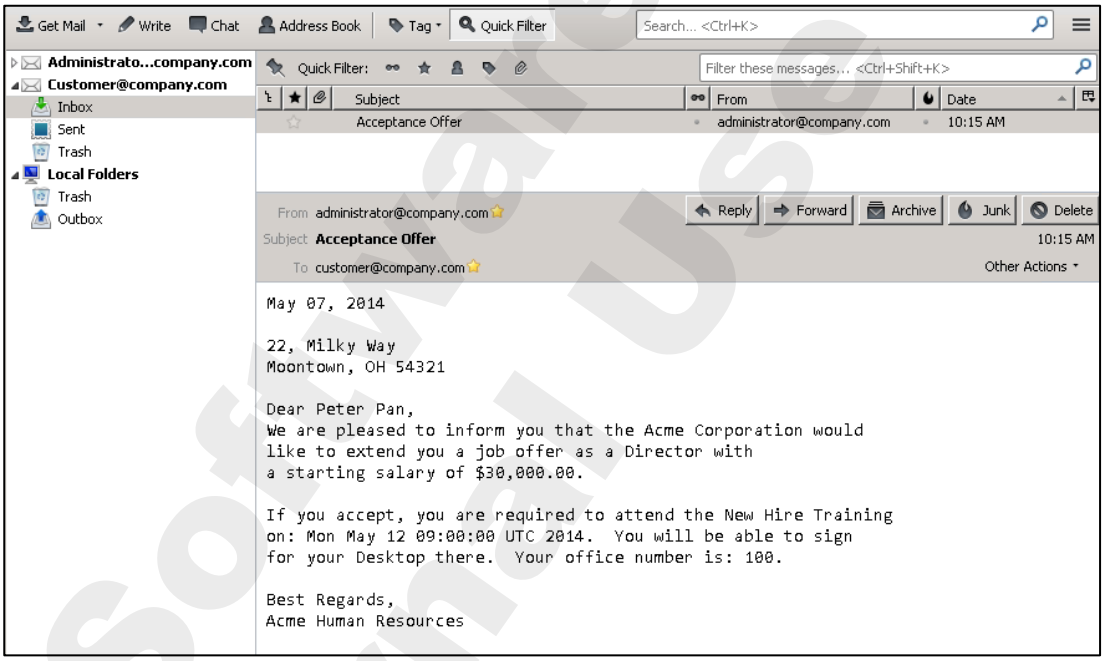
© 2000 – 2014 by Software AG

d. In My webMethods, navigate to **Applications** > **Monitoring** > **Business** > **Process Instances**. Click **Search** and verify a process instance of process model type **AddEmployee** has been completed successfully.



e. Logout from My webMethods.

f. *Optional*: Start Mozilla Thunderbird. Have a look for a new email with subject **Acceptance Offer** sent to email account Customer@company.com (password: **manage**):

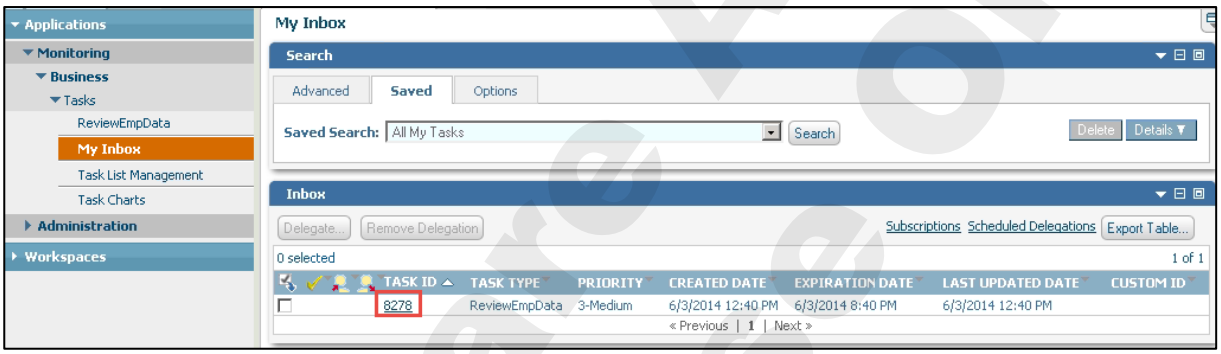


Software AG Internal Use Only!

12. Test your process again acting as a Mobile Task User:

    a. In Designer, right-click IS service **bpmTestDrive.services.util:sendAddEmployeeDoc** in the Package Navigator view and select **Run As** > **Run Flow Service.**
    When get prompted for employee input data, load the same data from file ***<workshop_dir>*\Ex8a\Resources\ProcessInput.txt** and hit **OK.**

    b. Use browser tab to login to My webMethods (http://localhost:8585) as **John** | **manage** or **Sumala** | **manage**. You should see a quite restricted navigation panel due to the permission settings applied above. Drill down to **Applications** > **Monitoring** > **Business** > **Tasks** > My Inbox. Click **Search** in the Search portlet. You should see a new User Task of Task type **ReviewEmpData** in the Inbox portlet. Click on the Task ID link to open the User Task's data view.



    c. Verify all business data are displayed. **Accept** and **Complete** the User Task.

    d. Logout from My webMethods.

Software AG Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 8B:
DEFINING AN MTI APP IN MOBILE ADMINISTRATOR

Objectives

In this exercise you will define the Mobile Task Integration (MTI) app in advance to Mobile Administrator. Mobile Administrator offers an environment that allows you remotely build the app for various device types and to offer it in an Enterprise app store.
To avoid naming conflicts in the common used Mobile Administrator, identifying resources have to be suffixed with your user ID.

In this exercise, please replace:
<workshop_dir> by **C:\Training\602D-66E\BPM**

Steps

1. Open a browser tab on your VM (or computer) and specify http://sag-training.appfresh.com to connect to Mobile Administrator. Login as **tDeveloper |mobile**.
2. In Mobile Administrator, define your Mobile Task Integration (MTI) app in advance. Specify name and other required information — such as an app description and an app icon — needed to present the app in the Enterprise App Store for iOS and Android devices. To do so:

   a. Select the "Manage Apps" tab and switch to the grid view

   .

   b. Click on the "+ App" toolbar button to add a new app.

   c. Select link **I will enter the application details manually** and enter the following input:

      - Name: **HR Tasks** <***YOUR_USERID***> (e.g. HR Tasks OBR)
      - Short Description: **The HR Tasks App**
      - Description: **HR Tasks app allows you to work on HR approval tasks**
      - Identifier: **HR_Tasks_<*your_userid*>**
        (one word without blanks, must be unique, e.g. HR_Tasks_obr)
      - Contact name: <***your_userid***>
      - Contact email: **<*your_mail*>**
      - Copyright: < ***your_userid*** >
      - Product category: select **Extended Training 9.6**
      - Keywords: **MTI,Training**
      - Downloads are public: ***leave unchecked***
      - Available on managed devices only: ***leave unchecked***

Software AG Internal Use Only!

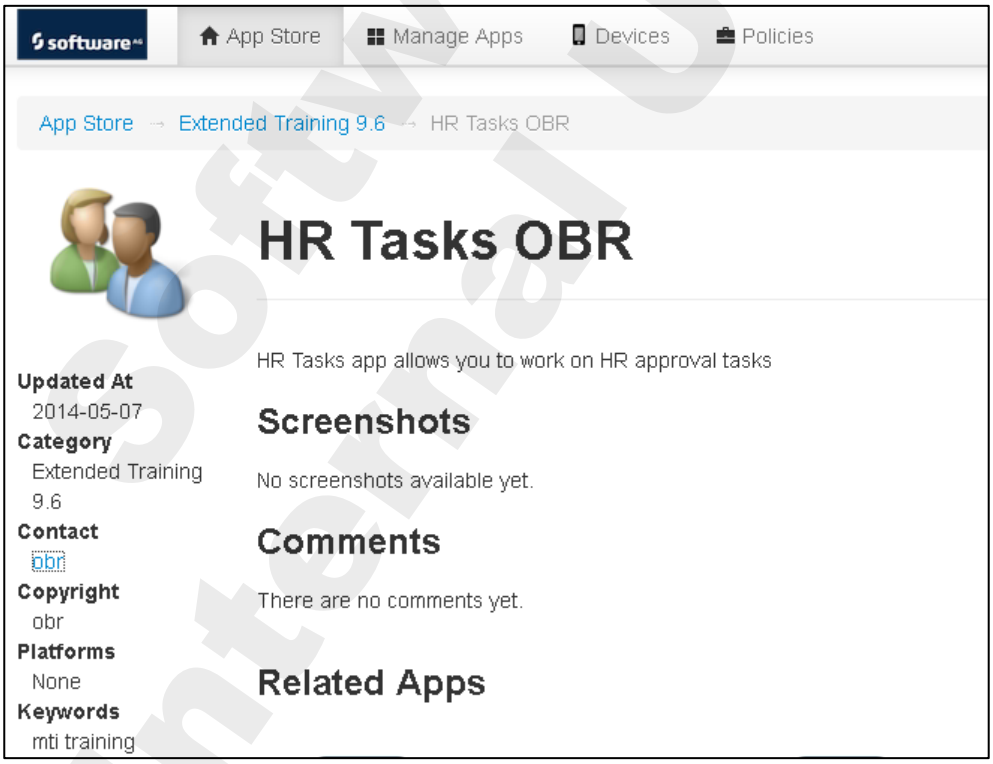
- Hidden from App Store listing: ***leave unchecked***
- Icon: browse for **<*workshop_dir*>\Ex8b\Resources\HRAppIcon.png**
- iOS Bundle ID: **com.softwareag.mobiledev.hrtasksmobile<*your_userid*>** (Package name must be all in lower case letters and must take the form of a fully qualified Java package name, e.g. com.softwareag.mobiledev.hrtasksmobileobr)
- iOS App Store ID: ***leave empty***
- Android Bundle ID: **com.softwareag.mobiledev.hrtasksmobile< *your_userid* >** (Package name must be all in lower case letters and must take the form of a fully qualified Java package name, e.g. com.softwareag.mobiledev.hrtasksmobileobr)
- Android Is in Play Store: ***leave unchecked***

d. Click on the ✔ Create App button to save your data.

e. Select **Permissions** from the left-hand side navigation and click + Add User Group to grant the **View and Download Stable Versions** permission to group **All Users**.

Confirm by clicking the + Add User Group button.

Finally, click the Show in App Store button to see how the MTI app will appear for end users in the Enterprise App Store. Of course, the MTI app is not installable as no binaries have been provided yet.



Software AG Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 8C: CREATING A MOBILE TASK INTEGRATION APPLICATION

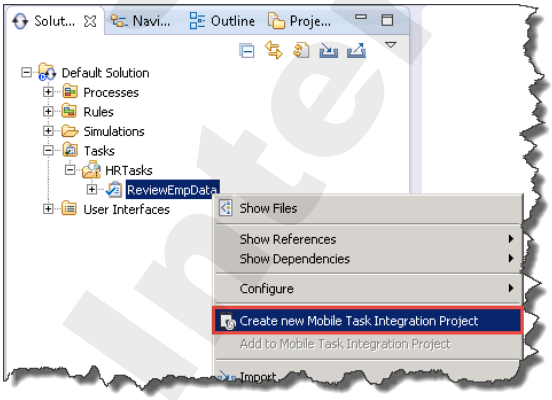
Objectives

In this exercise you will use Designer to generate a local Mobile Task Integration app based on your existing User Task Type in Designer. As a first approach, the generated app will be tested locally using Phoney using the administrator account.

In this exercise, please replace:
<workshop_dir> by **C C:\Training\602D-66E\BPM**

Steps

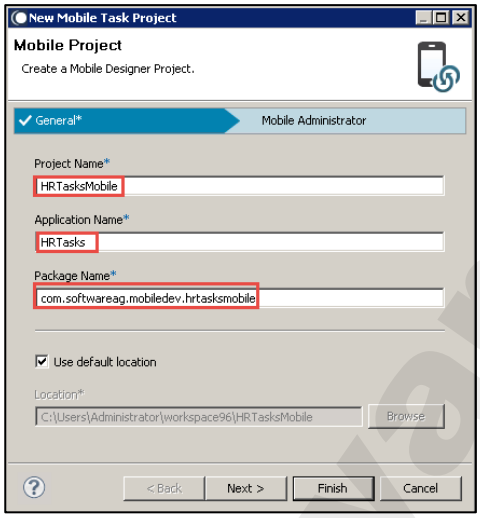
1. Launch Designer and use the default workspace.
   a. Switch to the **Process Development** perspective.
   b. To create some new sample User Task instances, use the Package Navigator view to run IS service **bpmTestDrive.services.util:sendAddEmployeeDoc** four more times (**Run As** > **Run Flow Service).** When get prompted for employee input data, load input from the following four input files:
      - ***<workshop_dir>*\Ex8c\Resources\ProcessInput1.txt**
      - ***<workshop_dir>*\Ex8c\Resources\ProcessInput2.txt**
      - ***<workshop_dir>*\Ex8c\Resources\ProcessInput3.txt**
      - ***<workshop_dir>*\Ex8c\Resources\ProcessInput4.txt**
2. Perform the following steps to generate an MTI project based on your existing User Task type in Designer:
   a. Open the **UI Development** perspective.
   b. In the Solutions view, right-click your User Task type **ReviewEmpData** within Task project **HRTasks** and choose **Create new Mobile Task Integration Project.**



Software AG Internal Use Only!

c. The New Mobile Development Project wizard comes up. Provide the following input on the **General** tab before you click **Next:**

- Project Name: **HRTasksMobile** (will become project name in Designer)
- Application Name: **HRTasks**
  (one word, no blanks, can be different from Application Name in Mobile Admin.)
- Package Name: **com.softwareag.mobiledev.hrtasksmobile**
  (Package name must be all in lower case letters and must take the form of a fully qualified Java package name, name can be different from chosen Bundle ID in central Mobile Administrator, *recommendation*: package name is prefix of chosen Bundle ID in MA)



d. On the **Mobile Administrator** tab, check **Use Mobile Administrator** and provide the following input:

- URL: https://sag-training.appfresh.com (Mobile Administrator home page)
- Access Token: **IOGYJKXXDIUOLODUDFGWGWCGUWWQVFQTP**

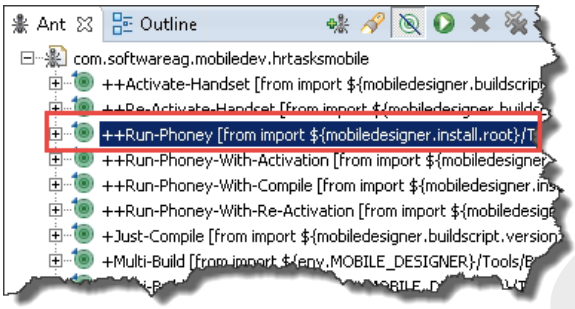
Then click the **Login** button. Check **Existing** and select your existing application identifier **HR_ Tasks_<*your_userid*>** to create an association. Click **Finish**.

Internal Use Only! Software AG

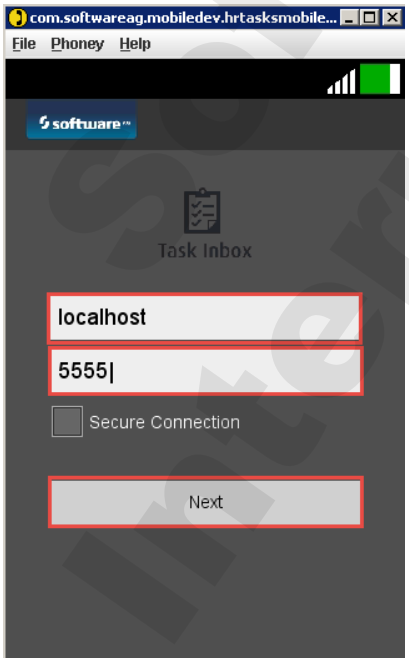
© 2000 – 2014 by Software AG

3. In Designer, switch to the **Mobile Development** perspective.
   a. Expand the generated **HRTasksMobile** project in the Package Explorer view.
   b. Add the **build.xml** file of the generated **HRTasksMobile** project to the **Ant** view. To do so, simply drag **build.xml** from the Package Explorer view and drop it to the Ant view. (As an alternative, right-click in the Ant view and select **Add Buildfiles…**).

      To suppress showing internal targets click the icon within the toolbar of the ant view.
4. You will now test the generated MTI app in Designer using the Phoney Emulator:
   a. Expand the tree in the Ant view and double-click the target **++Run-Phoney** to run it.



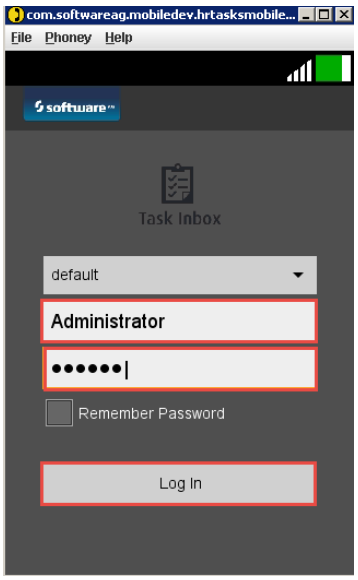
   b. A Phoney window opens automatically. By default, an Android 2.3 handset (AND_generic_Android23xAPI) has been activated because it is the first one in the list of targets (in the next exercise we will choose a different device type emulation instead). You should see the first MTI panel asking for connectivity data. Provide **localhost** and **5555** as input and hit **Next** to establish the connection.
      Hint: When you click into an input field once, you will see a virtual keyboard for input popping up. If you click into any space outside the keyboard a second time, keyboard vanishes and you can provide input by using the “real” keyboard to the field selected first.



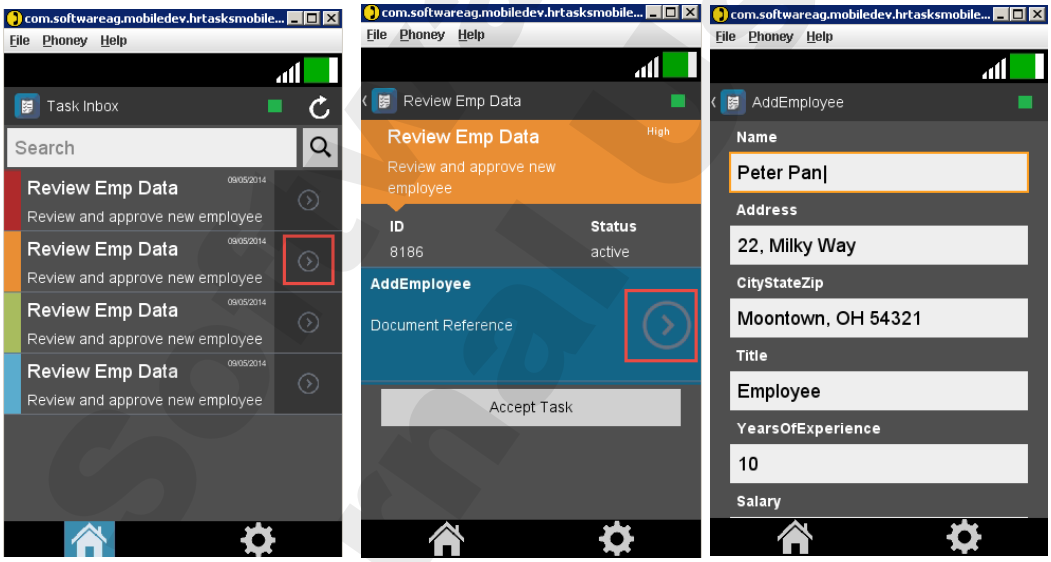
   Your connectivity data will be automatically stored by the app as a **default** connection.

Software AG Internal Use Only!

c. On the next panel, confirm the default connection and authenticate as **Administrator |manage**. Click **Log In** to connect.



d. Because Administrator is a Task assignee also, you should see at least four User Tasks with different priorities in the Mobile Task Inbox for user Administrator. Click on one Task instance to drill down into its details and business data:



e. Do NOT accept the User Task yet. Click to **log out** from webMethods.

f. Close Phoney.

Internal Use Only! Software AG

© 2000 – 2014 by Software AG

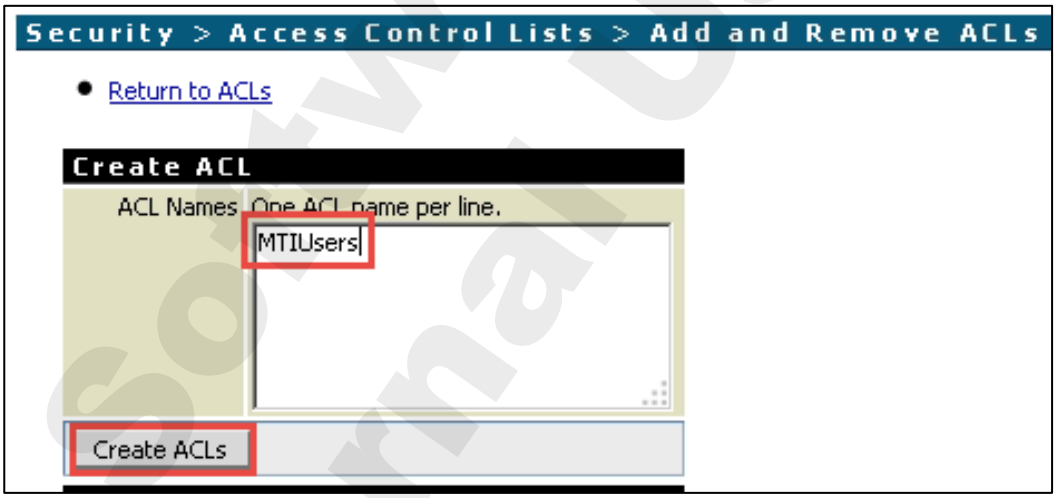
Exercise 8d: Working on Tasks using your MTI App in Phoney

Objectives

In this exercise you will use the generated Mobile Task Integration app as a business user (John or Sumala). Running the MTI again will be performed using Phoney.
To allow business users to access User Tasks from an MTI, you have to define additional privileges and access control settings in Integration Server in advance. These settings will allow a business user like John or Sumala to invoke predefined services in IS package WmTaskClient via REST calls. The invoked REST services will invoke the corresponding services in the Task Engine using SOAP communication between IS and MWS.

Steps

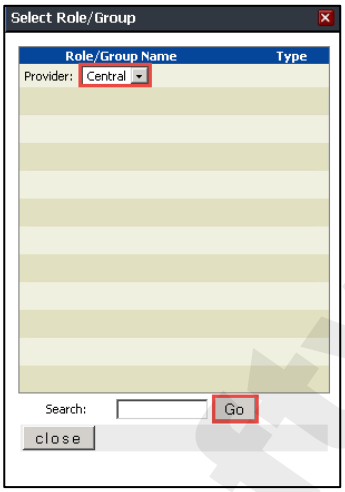
1. Use a browser tab to login to IS Administration (http://localhost:5555) as user **Administrator | manage**. Define the following privileges and access control settings to allow business users to connect from an MTI application to the Task Engine via Integration Server:
   a. In the IS Administration console, navigate to **Security** > **ACLs** and click the link **Add and Remove ACLs**. Create an ACL named **MTIUsers:**

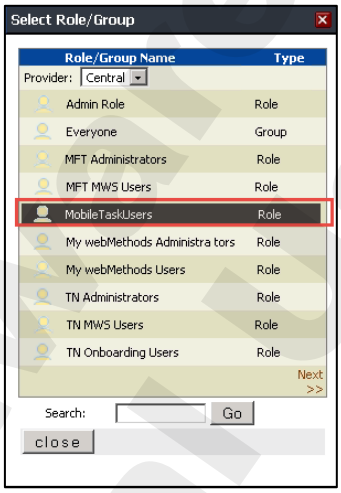


Software AG Internal Use Only!

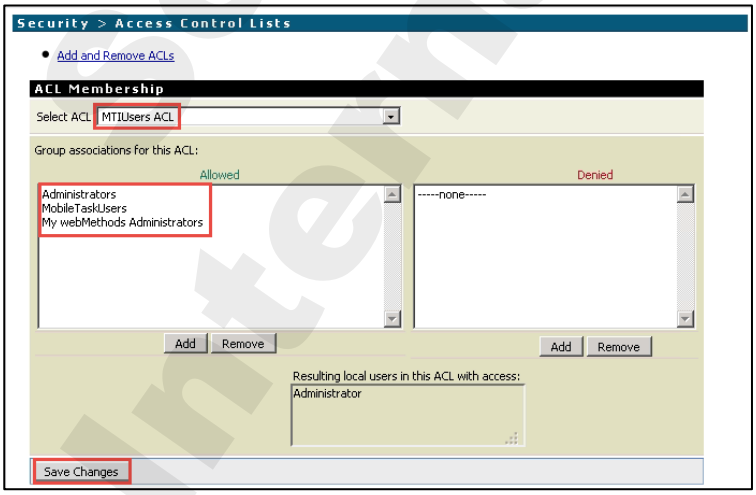
b. On page **Security** > **ACLs** select your new (and empty) ACL **MTIUsers ACL** from the drop down (suffix ACL has been added automatically).
Click **Add** to add the **local** IS group **Administrators** to your ACL.

Click **Add** again. This time select provider type **Central** and hit **Go** to browse for existing MWS principals. Select MWS role **MobileTaskUsers**.





Click Add for a third time. Select provider type **Central** and add MWS role **My webMethods Administrators.** Your MTIUsers ACL should look now like this:



Save your changes.

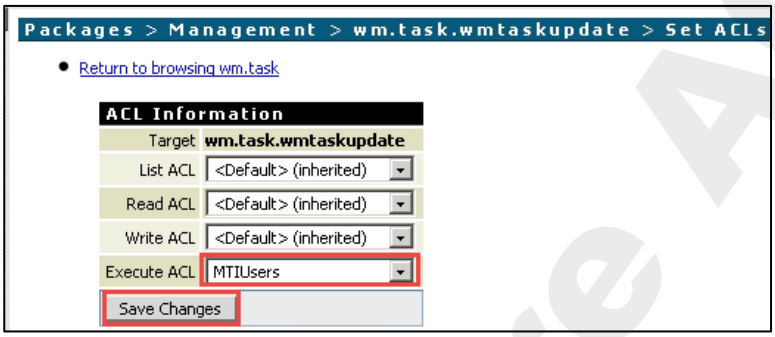
Software AG Internal Use Only!

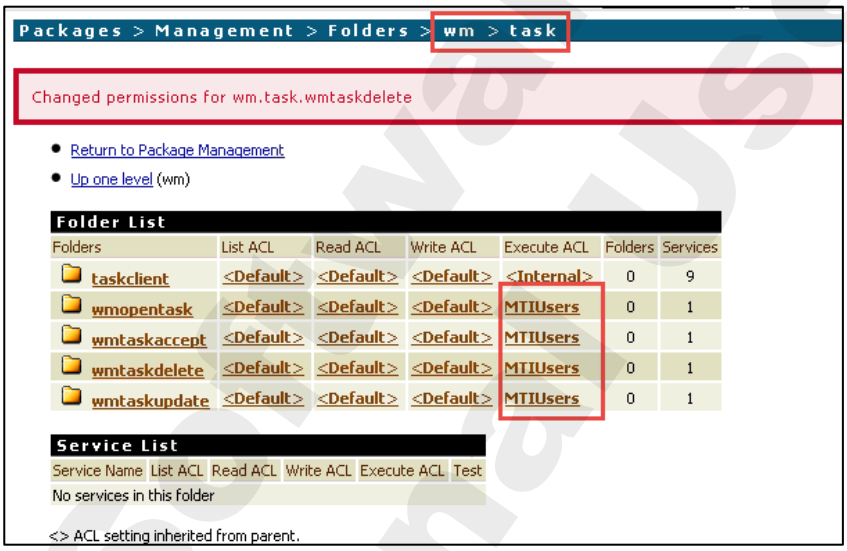
© 2000 – 2014 by Software AG

c. In the IS Administrations console, navigate to **Packages** > **Management**. Click link **Browser Folders**. Drill down to folders **wm** > **task**.
For each of the subfolders

- **wmopentask**
- **wmtaskaccept**
- **wmtaskdelete**
- **wmtaskupdate**

set the Execute ACL to your **MTIUsers** ACL and hit **Save Changes.**



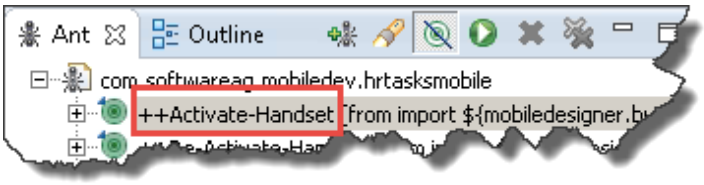


Software AG Internal Use Only!

2. Launch Designer and use the default workspace. Switch to the **Mobile Development** Perspective.

3. Acting as business users John and Sumala, execute the generated Mobile Task Integration application using Phoney:

   a. First activate a device simulation of your choice. To do so, double-click to run **++Activate-Handset** in the Ant view:

   and choose your preferred handset

   - **AND_generic_Android4xAPI** *xor*
   - **IOS_Apple_Universal)**

   xor

   Click **Activate Handset** to start the local build for the chosen device type.

   b. After a successful build, execute **++Run-Phoney** from the Ant view to start the emulator. *Note*: The following screen shots are taken using the Android 4.x emulator (because the class developer could not afford an expensive iOS device ☺).

   c. On the login screen, use the **default** connection and login as **john |manage** (all characters in lower case).

   d. On the Task Inbox panel you should see (at least) four Task instances, all of them with different priority indicators (colors):

      i. Red = critical

      ii. Orange= high

      iii. Green = medium

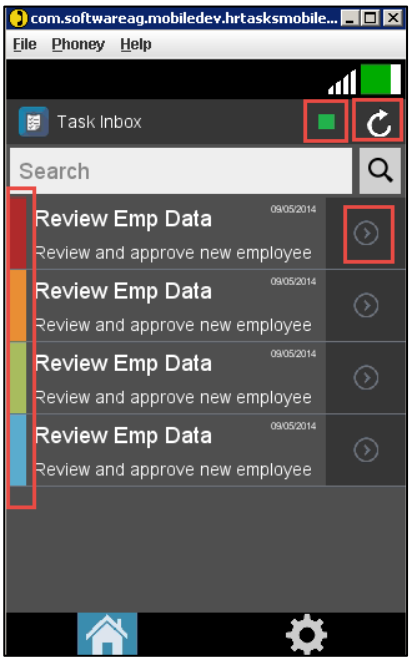
      iv. Blue = low

      v. Purple = none

Software AG Internal Use Only!

© 2000 – 2014 by Software AG

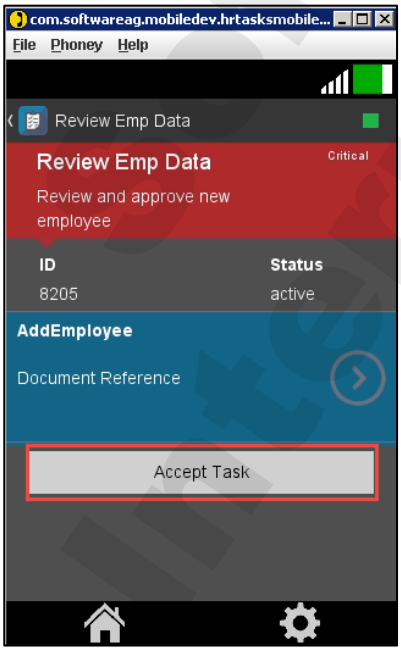
A green icon in the upper right corner indicates the server connection status. A Refresh icon is available to refresh the list at any time. Select the critical User Task instance (red) to open its User Task details.



e. On the Task details panel, the following Task metadata will be displayed:

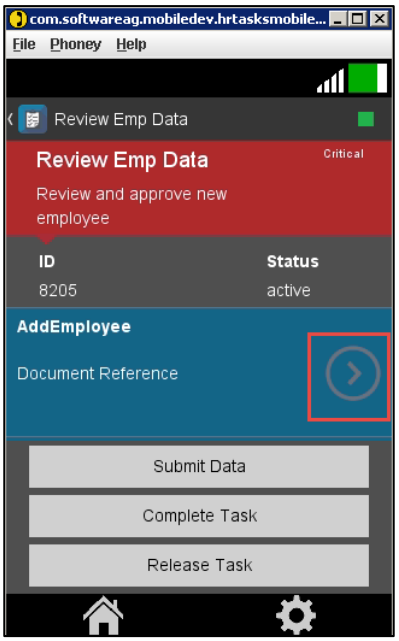
- Task ID
- Status
- Description
- Priority (text and color).

Accept the User Task instance.

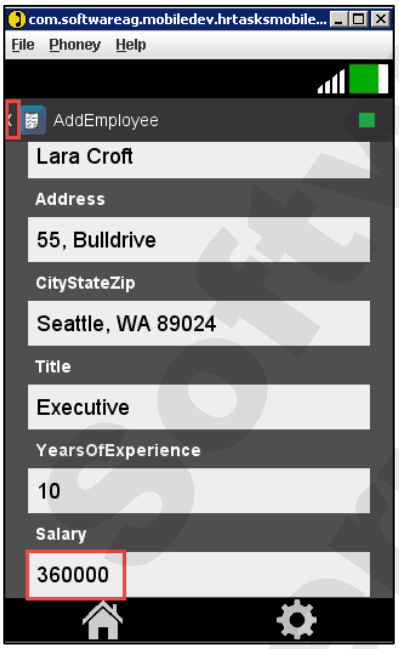


Software AG Internal Use Only!

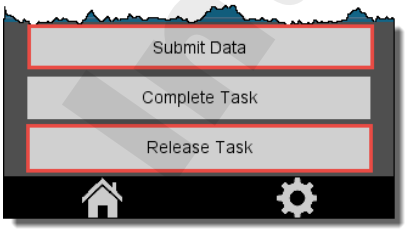
f. Because the task business data consists of a structured IS document, you don't see them (together with the Task metadata) on the Task details panel. To see the business data on a separate panel, clicking the > icon on the document reference.



g. On the Business Data panel, change the **Salary** of the employee to a value of your choice. Use the < icon to go back to the Task details panel.



h. On the Task details panel first click **Submit Data**. Select the same Task again from the Task list and click **Release Task** to release the task.



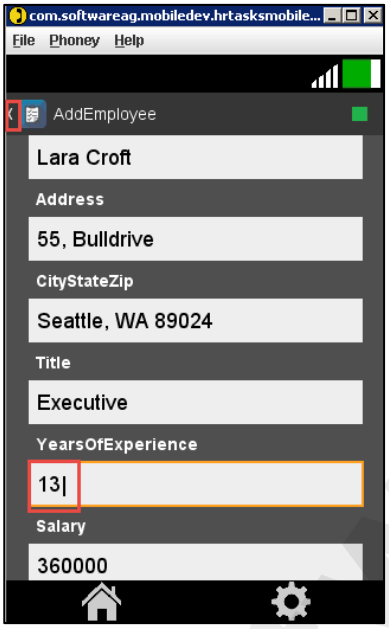
Internal Use Only! Software AG

© 2000 – 2014 by Software AG

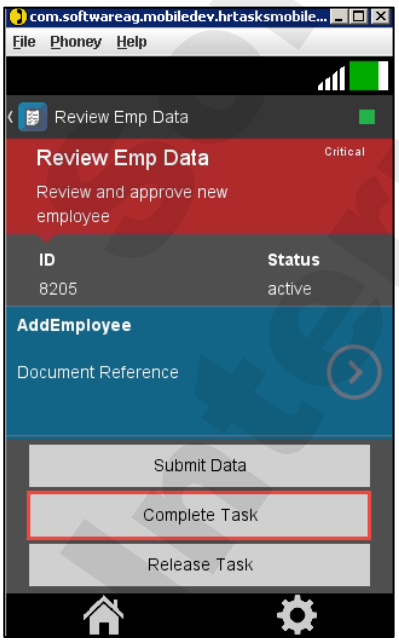
i. Leave Phoney up and running (don't close the windows).
Re-visit the Ant view in Designer. Execute **++Run-Phoney** (or **++Run-Phoney-with-Activation** – allows you to change the handset type for this second Phoney session in advance) to start a second Phoney session.
For the second session, use the default connection and login as **sumala** | **manage** (all in lower case).

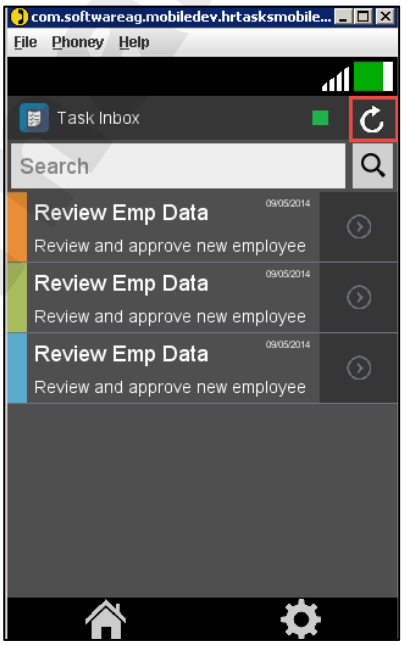
j. On the Task Inbox panel, open the same (critical) User Task instance changed above by John. On the Task details panel, **Accept** the task. Open the business data and ensure you see the salary data changed by John.

k. Update business data field **Years of Experience** and go back to the Task details panel.



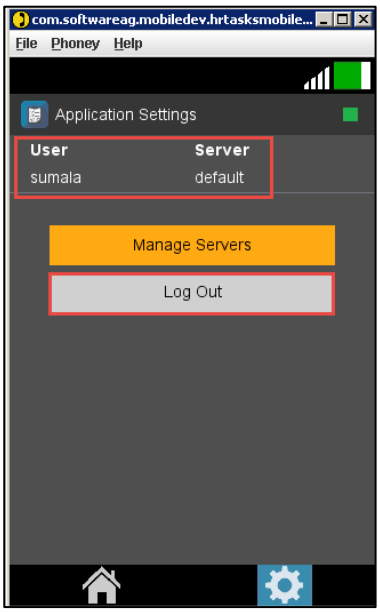
l. **Complete** the User Task. On the Task Inbox panel, the completed (critical) task should have vanished. If necessary click on the **Refresh** icon to reload the task list.





Software AG Internal Use Only!

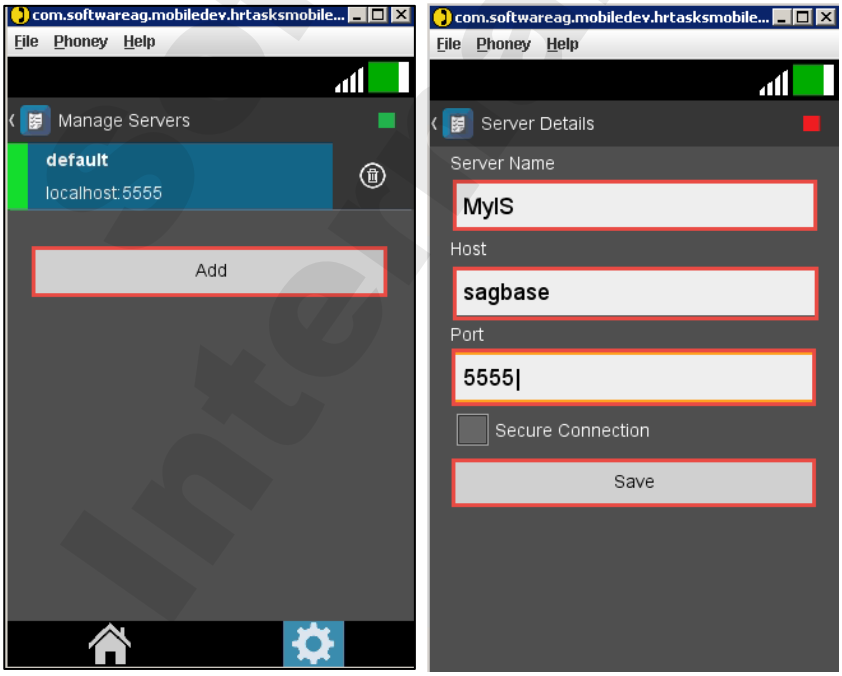
m. Click on the icon to open the Application Settings panel. Verify that you are still acting as user **sumala** connected using connection **default**. **Log Out** from here.



4. Switch to the first Phoney session still opened from Designer (that's John's session):

a. If you don't see a panel titled Task Inbox, click the home icon to open (and reload) it. If you are already seeing the Task Inbox panel, click on the Refresh icon to reload its content. In any case, the critical task should not be displayed in the task list anymore because the task has already been completed by Sumala.

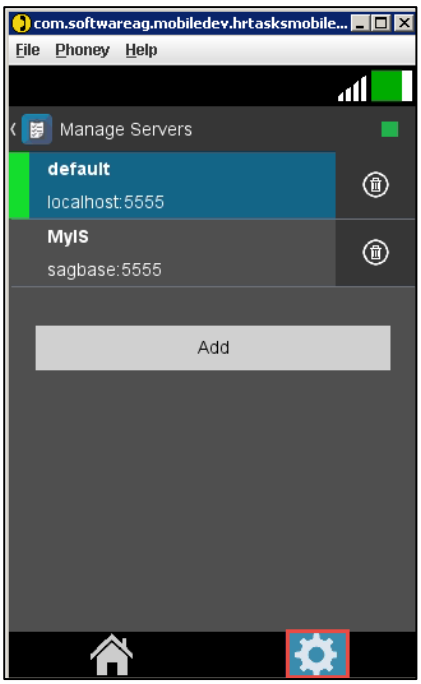
b. Click on the icon to open the Application Settings panel. Verify that you are still acting as user **john** connected using connection **default.**
Click **Manage Servers** to **add** another connection named **MyIS** to an Integration server running on host **sagbase** using port **5555**. Hit **Save**.

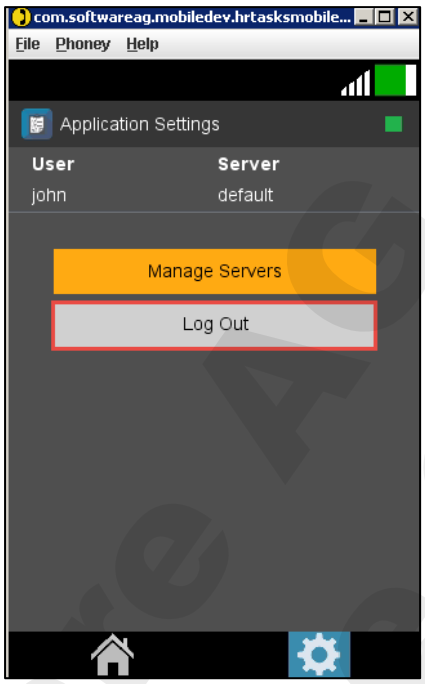


Internal Use Only! Software AG

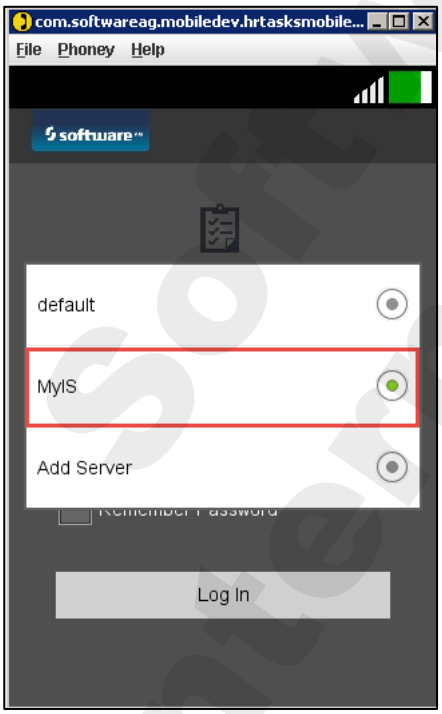
© 2000 – 2014 by Software AG

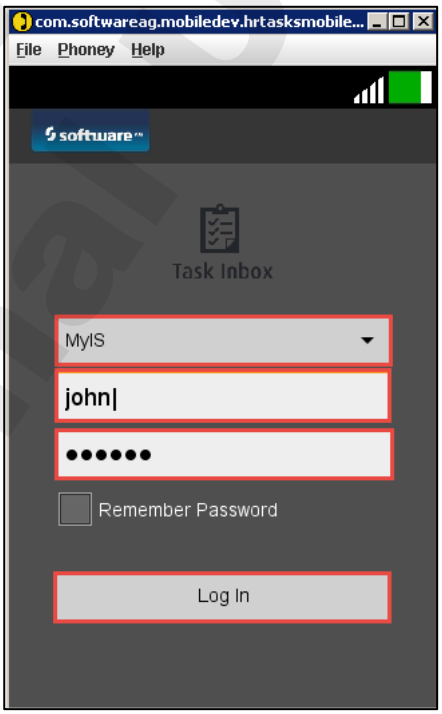
c. On the Manage Servers panel, click on the icon again and **Log Out** from server default.





d. On the Welcome panel, select server **MyIS**, authenticate as **john** | **manage** (all in lower case) and login again.
Note: You could have also added your server during login from the server drop-down.

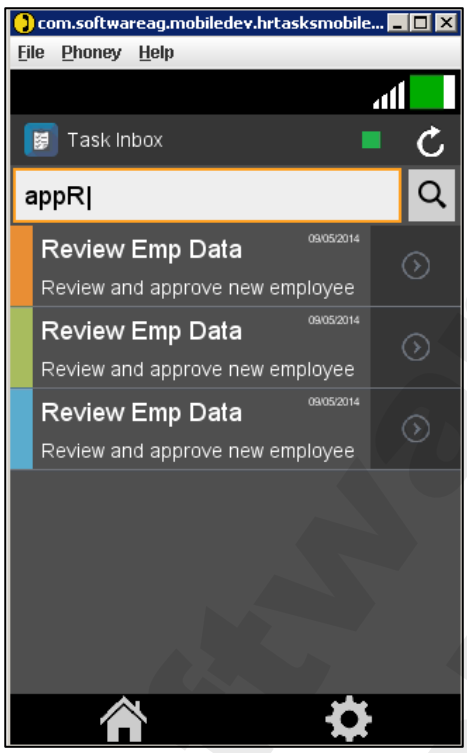


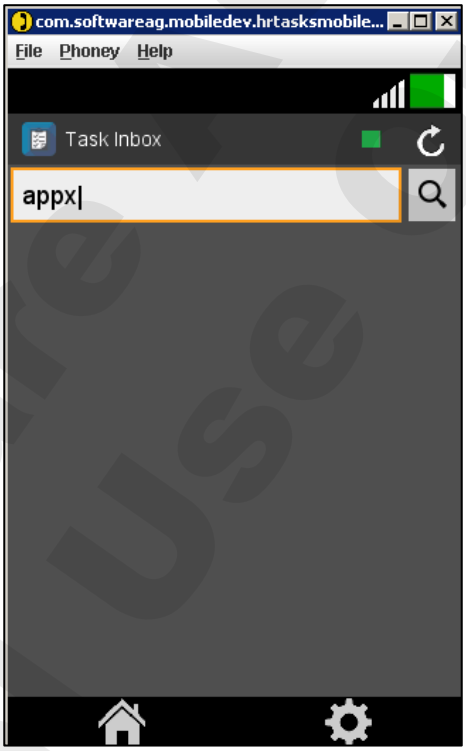


Software AG Internal Use Only!

e. After successful login, you should see the Task inbox panel showing John's Task Inbox retrieved using server connection MyIS. Server status is indicated by a green square. Tab on the magnifying glass or click into the **Search** field to specify a search filter value. Type in the following search (filter) patterns:

- **rev**
- **DATA**
- **appR**
- **Appx**





Each pattern immediately filters the task list. The pattern is compared to the task name and task description of all Inbox tasks. Lower and upper case is not taken into account.

f. On the Manager Servers panel, click on the icon again and **Log Out** from server MyIS.

5. Close both Phoney sessions in Designer.

Software AG Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 8E (*OPTIONAL*): BUILD AND PUBLISH MOBILE TASK INTEGRATION APP

Objectives

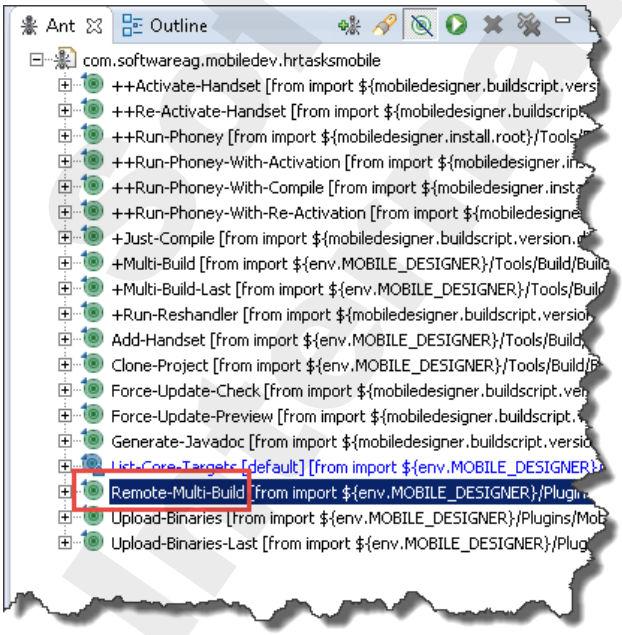
In this exercise you will use initiate a remote build of your Mobile Task Integration app on Mobile Administrator for iOS and Android devices. Because of the definitions and associations made in exercise 8b and 8c, the app will become available in the enterprise app store of Mobile Administrator (appfresh). Additionally, you can download and install your MTI app on your own mobile device (requires for an internet connection to your mobile device).

***Important note*: This optional exercise accesses internal Software AG resources, so its feasibility depends on your classroom settings and the availability of those internal resources. Ask your trainer for details.**

In this exercise, please replace:
<workshop_dir> by **C:\Training\602D-66E\BPM**

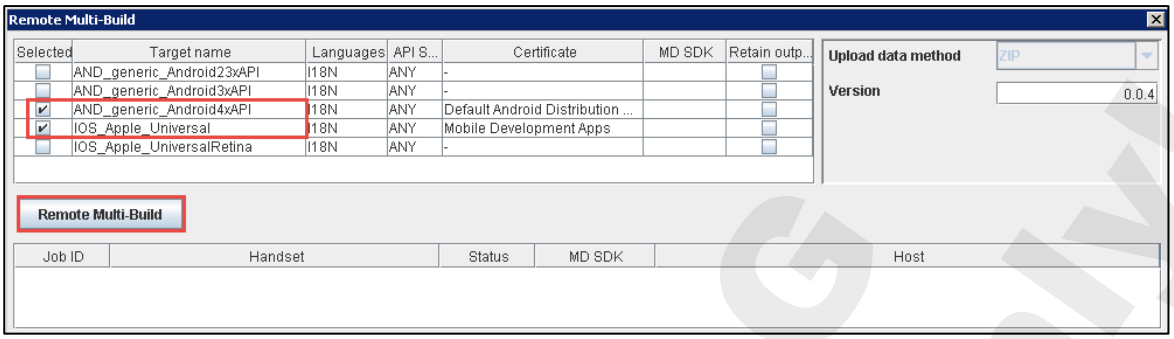
Steps

1. Build and publish your MTI project on Mobile Administrator from **Designer** (remote build):
   a. Launch Designer and use the default workspace.
   b. Switch to the **Mobile Development** perspective. Ensure your MTI project named **HRTasksMobile** is opened in the Package Explorer view, its build.xml file has been added to the Ant view and internal targets are hidden ( ).
   c. Execute **Remote-Multi-Build** from the Ant view.



Software AG Internal Use only!

d. Select Target names **AND_generic_Android4xAPI**, **IOS_Apple_Universal**, and –if necessary - another Target name that fits to the OS version of your own device and click **Remote Multi-Build.**



Hint: In the background, Designer will upload your MTI project as a zip file to Mobile Administrator as configured in exercise 8c. On Mobile Administrator, a build configuration will be created. Based on the configuration settings, Mobile Administrator cross-compiles, and builds binaries of your MTI project for each chosen target name (platform).
The progress status will be indicated in Designer:

Job ID	Handset	Status	MD SDK	Host
2366	AND_generic_Android4xAPI	RUNNING	9.6.1.0.390	AppFreshs-Mac-mini.local
2367	IOS_Apple_Universal	PENDING		

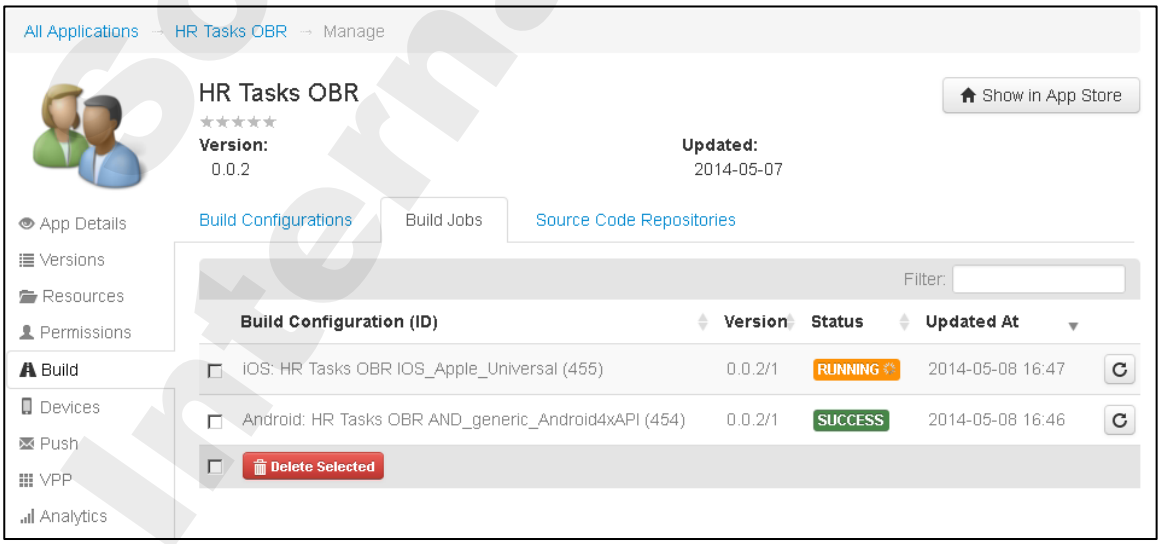
2. During cross-compiling and build, open a browser tab to login to **Mobile Administrator** (http://sag-training.appfresh.com):

   a. Login as **tDeveloper** | **mobile**.

   b. Select the "Manage Apps" tab.

   c. Find your personal HR Tasks app **HR Tasks** <***YOUR_USERID***> (e.g. HR Tasks OBR) and click on the icon to open its settings as defined in exercise 8b.

   d. Select **Build** from the left-hand navigation bar and open tab **Build Jobs**. Depending on the time already passed by, you should also see the build jobs here and their current status:

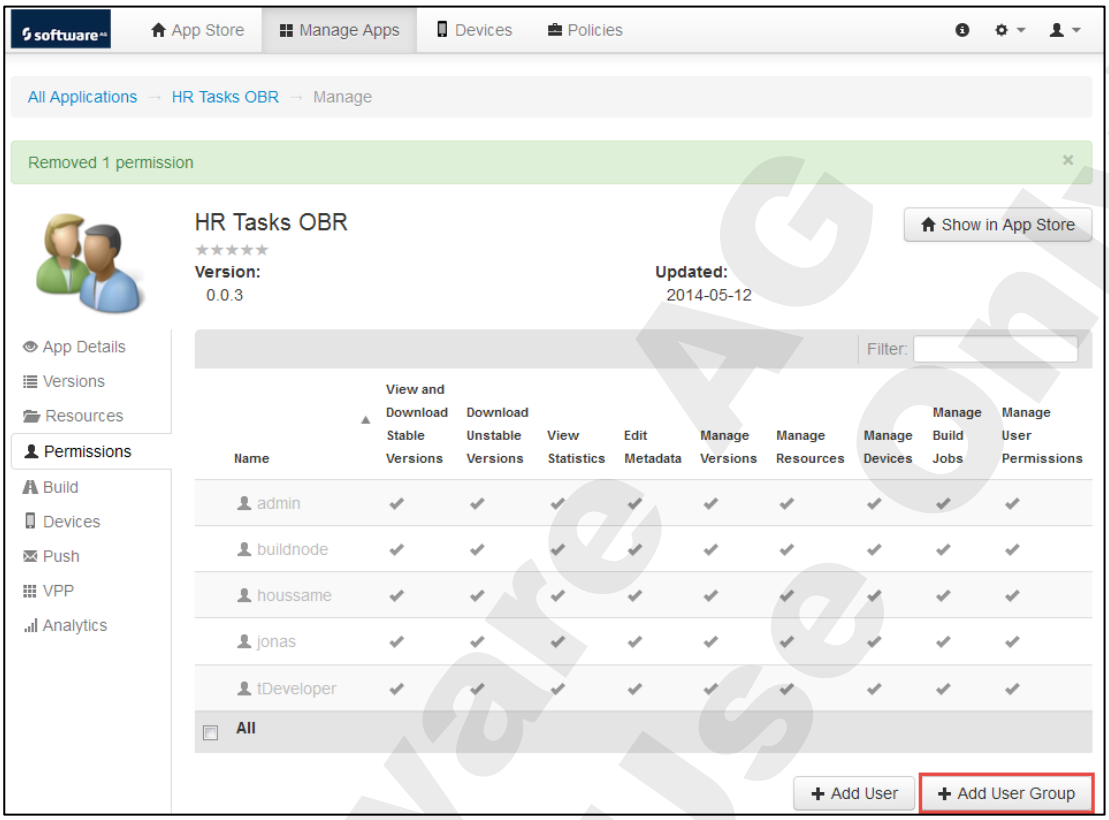


Ensure all build jobs complete successfully.

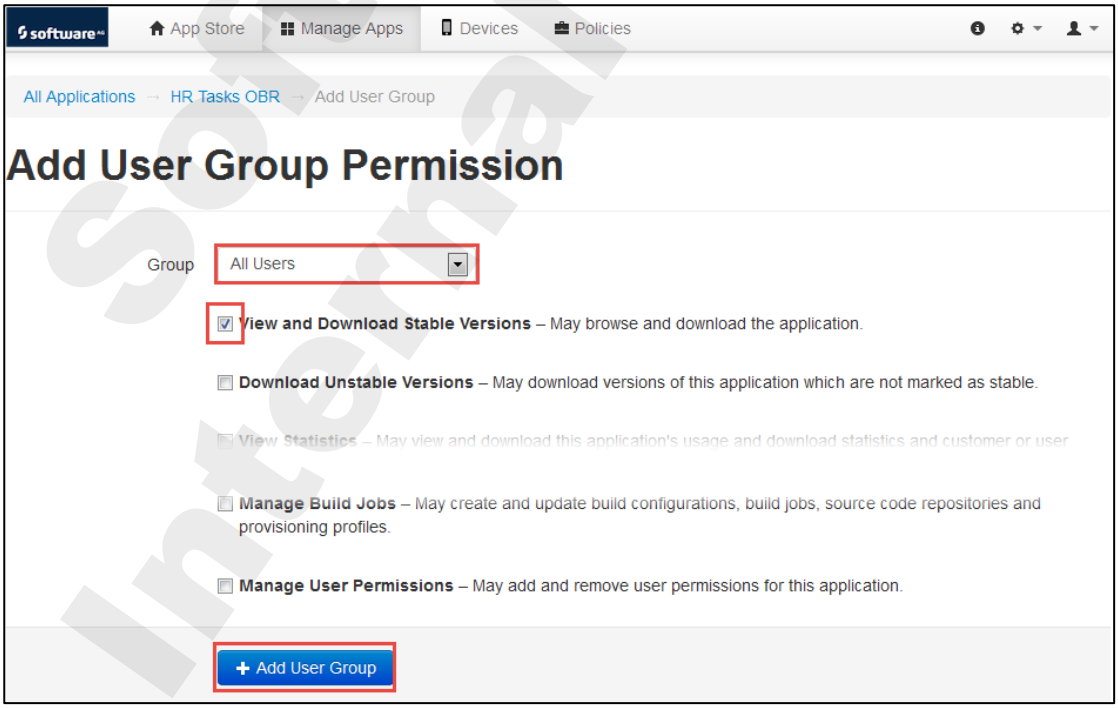
Software AG Internal Use Only!

© 2000 – 2014 by Software AG

e. To allow MTI users to download the app from Mobile Administrator, click on **Permissions** in the left-hand navigation bar. Only if **All Users** are not listed yet, hit **+ Add User Group**.

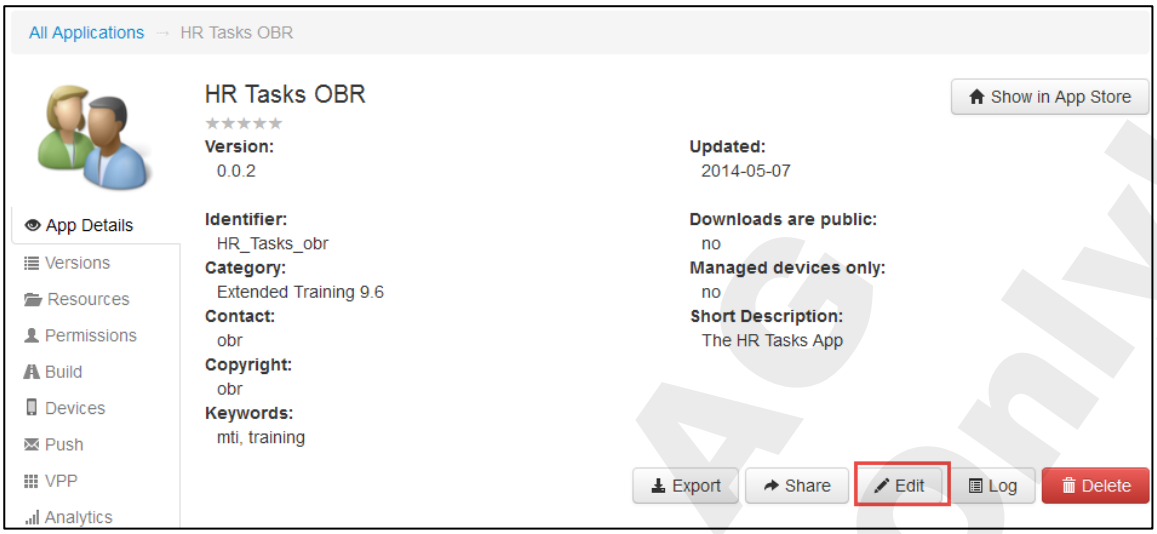


Ensure, **All Users** are allowed to **View and Download Stable Versions**.

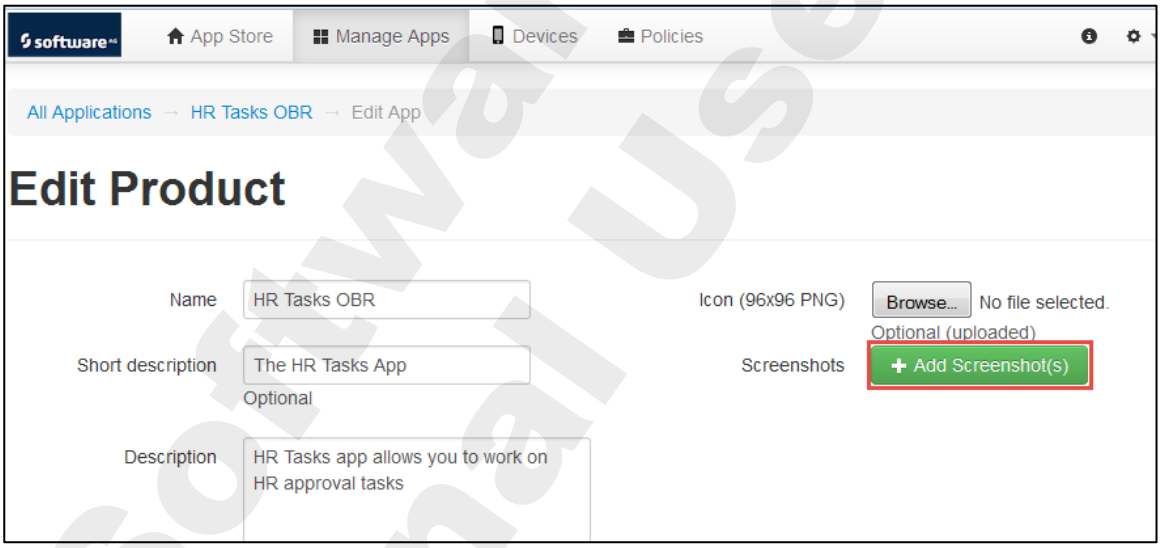


Software AG Internal Use Only!

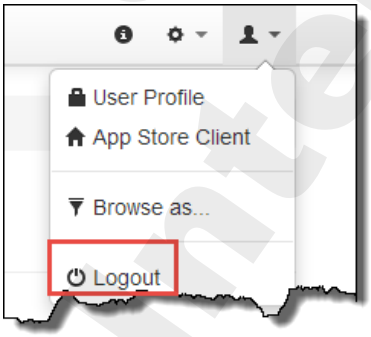
f. Finally click on **App Details** in the left-hand navigation bar. Hit **Edit**.



Add screenshot from file <***workshop_dir***>**\Ex8e\Resources\HRAppScreenshot.jpg** and hit **Update App** at the very bottom of the Edit Product page.



g. **Logout** from Mobile Administrator as tDeveloper.



Software AG Internal Use Only!

© 2000 – 2014 by Software AG

3. Use a browser tab to login to **Mobile Administrator** (http://sag-training.appfresh.com) as end user (MTI user) to browse the Enterprise App Store:

   a. Login as **tUser** | **mobile**.

   b. On the App Store page, type **HR Tasks** as search pattern press **Enter**. You should see all tasks having this value as part of their name.

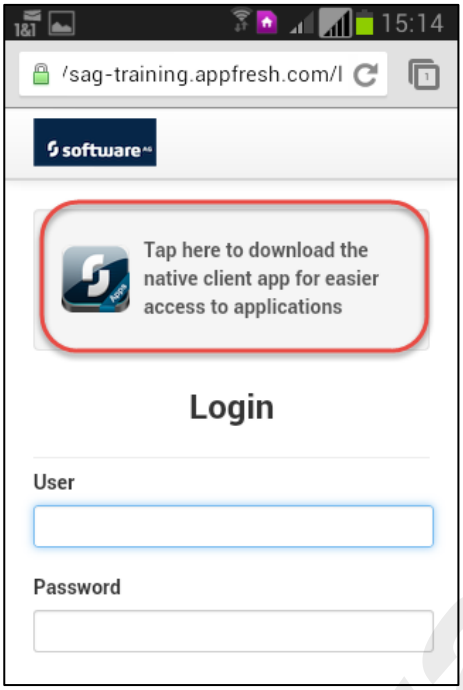
   c. Click the icon of your app to view the details:

   d. **Logout** from Mobile Administrator as tUser.

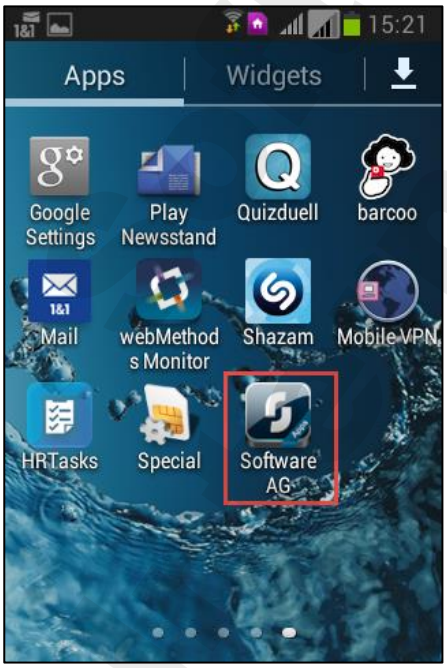
Software AG Internal Use Only!

4. *Optional*: Download and install your MTI app from Mobile Administrator on your own mobile device (requires for an internet connection to your mobile device):

   a. On your mobile device, open a browser tab and connect to **Mobile Administrator** (http://sag-training.appfresh.com). Tap on the marked area to download and install the **App Store Client** on your device.



   b. On an iOS device confirm that you want to install the app. On an Android device find the downloaded **.apk** file in folder **Downloads** and tap on it to start its installation.
   Finally you will find the App Store Client as **Software AG** app on your mobile device:

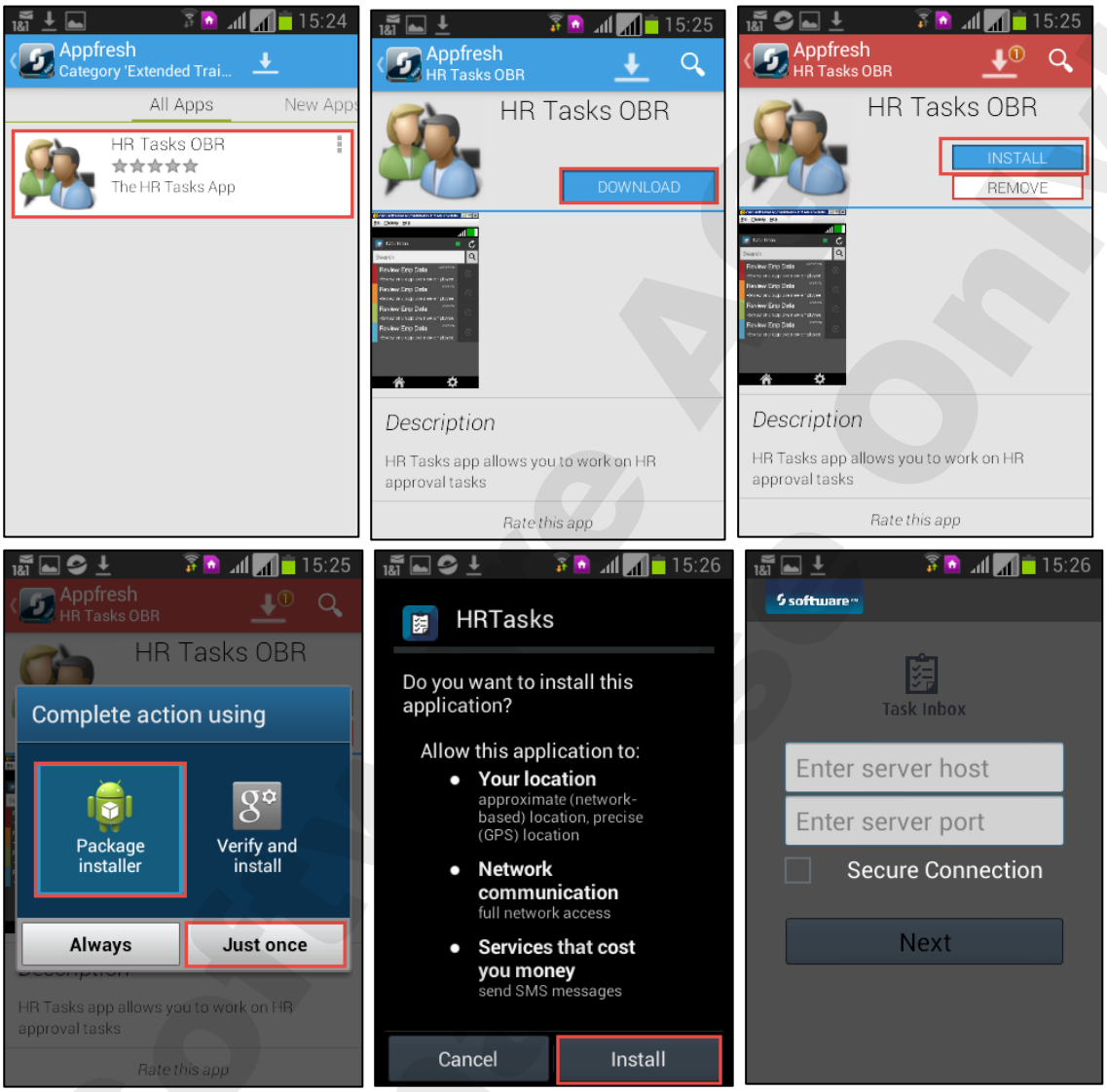


Software AG Internal Use Only!

© 2000 – 2014 by Software AG

c. Run the **Software AG** app (App Store Client) and login as **tUser | mobile.**

d. Find the **HR Tasks** <***YOUR_USERID***> app in the **Extended Training 9.6** category, download, install, and launch it on your mobile device.



5. Final housekeeping:

a. On your VM, use a browser tab to login to **Mobile Administrator** (http://sag-training.appfresh.com) as **tDeveloper | mobile**.

b. Select the "Manage Apps" tab.

c. Find your personal HR Tasks app **HR Tasks** <***YOUR_USERID***> (e.g. HR Tasks OBR) and click on the icon to open its settings.

d. Click **Delete** to remove your app from the app store.

e. Logout from Mobile Administrator.

Internal use only!

Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

TABLE OF CONTENTS – ESB, TERRACOTTA AND UNIVERSAL MESSAGING

Exercise 01: Service Results Caching 105
Exercise 02: Caching with overflow to off-heap (e.g. BigMemory) 109
Exercise 03: Using Cache Search 115
Exercise 04: Using the Enhanced XML Parser with BigMemory 121
Exercise 05: JSON support in Integration Server 129
Exercise 06: Using Global Variables 131
Exercise 07: Use IDataMap in a Java Service 133
Exercise 08: Publish / Subscribe with Integration Server & Universal Messaging Provider for JMS 135

Software AG Internal Use Only!

This page intentionally left blank

Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 01: SERVICE RESULTS CACHING

Objectives

In this exercise you will test a service WITHOUT service caching enabled and then view the results. You will then enable caching and view the results.

Since this is not an exercise in Flow or Java service developing, we have some pre-built services you can use. They are:

utilityTestService in namespace: **V96KnowledgeUpdateESB.Caching**
This service calls **utilitySvcToGetResult** (this will be the service you will EVENTUALLY enable caching).

utilitySvcToGetResult calls a Java service **calculateFactorial** in namespace: **V96TrainingESBSupport.support.caching**. When this service is called WITHOUT caching enabled it will require multiple calls to get the result. After you enable caching on this service you will see the number of calls as **1** (because the result will be pulled from the serviceResults cache).

calculateFactorial service uses an input value of N to calculate and return a factorial number. Think of this service as a backend location for the data. In reality this could be a database table, 3rd party app, file system, etc….

Steps

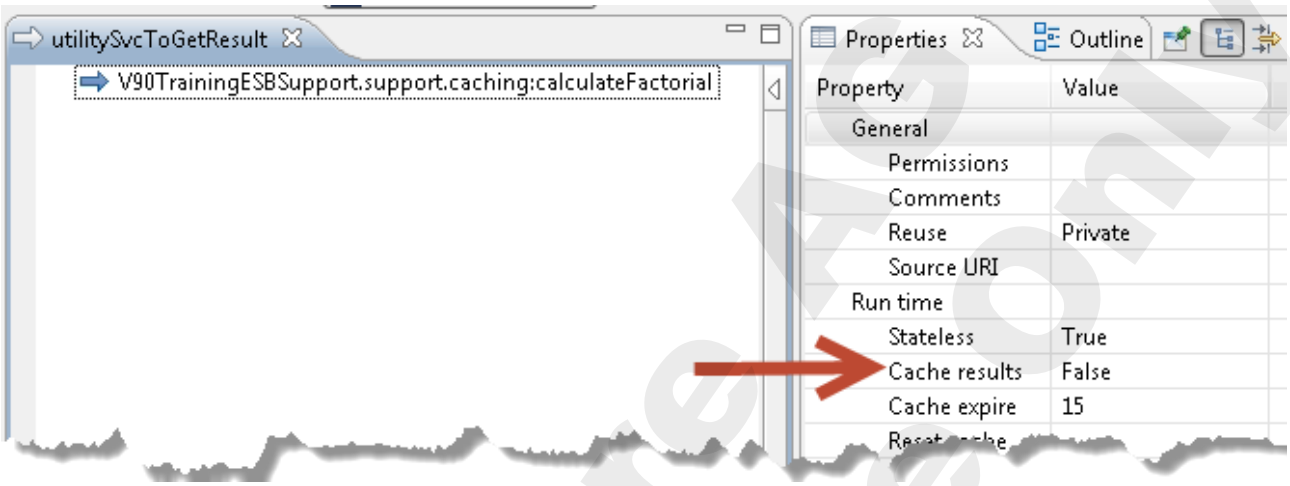
1. Start the following webMethods services:
   a. Broker Monitor / Server: Software AG Broker Monitor 9.6 (6850) Software AG Broker Server 9.6 (6849)
   b. Universal Messaging realm: Software AG Universal Messaging 9.6.0.0.12476 (umserver) 1
   c. Integration Server: Software AG Integration Server 9.6 (default)
   (do not start the server with "(2)" at the end)

   Note: The Integration Server we are using is located in **C:\SoftwareAG\IntegrationServer** and has an **HTTP port** of **5555**.
2. Enable the packages **V96TrainingESBSupport and V96KnowledgeUpdateESB** using the Integration Server package management on your VM.
   Use IS Administration Web UI at http://localhost:5555
   Username=Administrator Password=manage
3. Start Software AG Designer and make sure you are in the Service Development perspective.
4. You will notice a package called **V96KnowledgeUpdateESB** with various subfolders. This will be your namespace for the upcoming exercises.

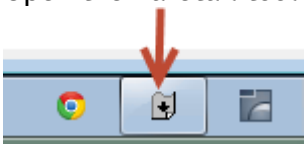
Software AG Internal Use Only!

5. Navigate to **V96KnowledgeUpdateESB.Caching** and review the services **utilityTestService** and **utilitySvcToGetResult**.
   You will notice that utilityTestService loops through a StringList called N, constructs an argument array, and then for each occurrence in the array it calls utilitySvcToGetResult. Then calculateFactorial is called to get the factorial result.
   Various debugLog services have been inserted to display the results.

6. Look at the service properties for **utilitySvcToGetResult** and make sure **Cache results** is **False**.



7. Open the Baretail tool using the shortcut from the Task bar.



   Open the tab to view server.log.

   Go back into Designer and run the service **utilityTestService.**

   View the results in Baretail. You should see the service results for looping through the value N from 1 to 9. The important line to look at starts with:

   2013-07-18 17:06:22 MESZ [ISP.0090.0003C] ***** TOTAL # OF CALLS to get N ….**

```
] *** Starting first LOOP - the value of N is: 3
] *** Response from utilitySvcToGetResult: input (factoralLookup) was 1     - Factorial result received (factorialResult) is: 1
] *** Response from utilitySvcToGetResult: input (factoralLookup) was 2     - Factorial result received (factorialResult) is: 2
] *** Response from utilitySvcToGetResult: input (factoralLookup) was 3     - Factorial result received (factorialResult) is: 6
] *** Response from utilitySvcToGetResult: input (factoralLookup) was 3     - Factorial result received (factorialResult) is: 6
] *** Response from utilitySvcToGetResult: input (factoralLookup) was 2     - Factorial result received (factorialResult) is: 2
] *** Response from utilitySvcToGetResult: input (factoralLookup) was 1     - Factorial result received (factorialResult) is: 1
] *** TOTAL # OF CALLS to get N (3) = 6  (If > 1 then value not in cache)
] *** Starting first LOOP - the value of N is: 4
```

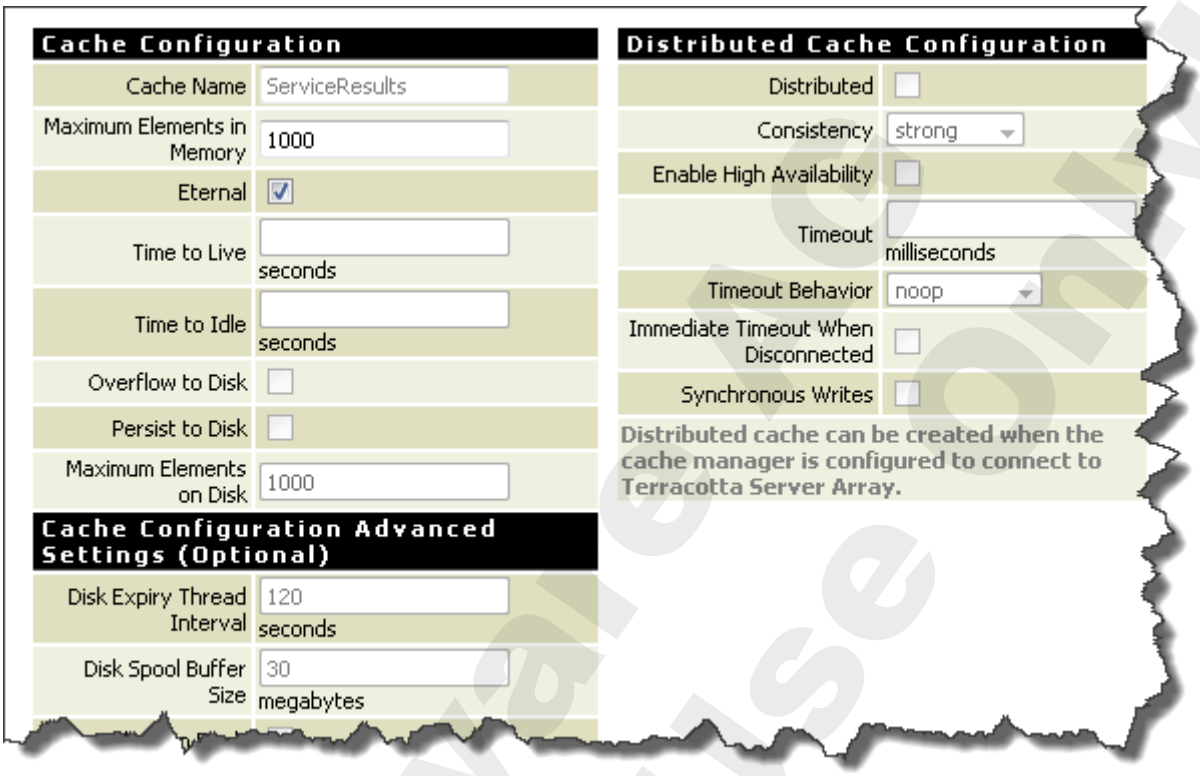
   If the total number of calls value is greater than 1 then that means the result is NOT in a cache and had to be calculated from calculateFactorial.

Internal Use Only! Software AG

© 2000 – 2014 by Software AG

8. Open the ServiceResults cache in the IS Administration Web UI (http://localhost:5555). This is where the cache can be configured.
   Remember, the ServiceResults cache cannot be distributed.

   Make sure Eternal is checked and leave Time to Live and Time to Idle blank.



9. Change the Cache results property in **utilitySvcToGetResult** to True and then re-run **utilityTestService**.

   View the results in Baretail. You should see the service results for looping through the value N from 1 to 9. The important line to look at starts with:

   2013-07-18 17:06:22 MESZ [ISP.0090.0003C] ***** TOTAL # OF CALLS to get N ….**

```
*** Starting first LOOP - the value of N is: 3
*** Response from utilitySvcToGetResult: input (factoralLookup) was 1     - Factorial result received (factorialResult) is: 1
*** Response from utilitySvcToGetResult: input (factoralLookup) was 2     - Factorial result received (factorialResult) is: 2
*** Response from utilitySvcToGetResult: input (factoralLookup) was 3     - Factorial result received (factorialResult) is: 6
*** Response from utilitySvcToGetResult: input (factoralLookup) was 3     - Factorial result received (factorialResult) is: 6
*** Response from utilitySvcToGetResult: input (factoralLookup) was 2     - Factorial result received (factorialResult) is: 2
*** Response from utilitySvcToGetResult: input (factoralLookup) was 1     - Factorial result received (factorialResult) is: 1
*** TOTAL # OF CALLS to get N (1) = 1  (If > 1 then value not in cache)
```

   HOWEVER, this time you will notice that the total # of calls is = 1 all the time. This means that the result was found in the serviceResults cache.

Internal Use only! Software AG

10. You can get some cache statistics using the IS Administration Web UI.
    Navigate to **Server > Service Usage**. Scroll down to the bottom of the page:

Cache and Prefetch Information

Name	Last Cache Hit	Cache Hits	Cache Hits/ Total Count	Cache Entries	C
¥96KnowledgeUpdateESB.Caching:utilitySvcToGetResult	2014-06-13 18:50:05 UTC	100	45%	10	

Softwware AG
Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 02: CACHING WITH OVERFLOW TO OFF-HEAP (E.G. BIGMEMORY)

Objectives

In this exercise you will create a new Cache Manager and Cache to contain 20,000 elements on-heap. Initially you will not enable off-heap for the cache, and will try to load 50,000 elements into the cache. You can then observe the behavior based on the cache settings.

Then you will enable overflow to off-heap (BigMemory) and run the load test again. You will again observe the results.

Note: the amount of entries that can be stored in the cache (on-heap and off-heap) depend on:

- how many entries will be stored
- size of the key and value stored
- size of your Java heap
- optionally: size of your off-heap (e.g. BigMemory)

Steps

1. Make sure your Integration Server is started (Windows service "Software AG Integration Server 9.6 (default)"
2. Set the amount of memory that you want to allocate to BigMemory. To do this, edit C:\SoftwareAG\IntegrationServer\instances\default\bin\setenv.bat

   Then uncomment the line that starts with "rem set JAVA_MAX_DIRECT_SIZE…." and then change **MaxDirectMemorySize** to 500M in

```
rem ##########################################################################
rem
rem Uncomment the following line and specify the amount of memory that you want
rem to allocate to BigMemory.
rem
rem ##########################################################################

set JAVA_MAX_DIRECT_SIZE=-XX:MaxDirectMemorySize=500M
```

Software AG Internal Use Only!

3. For the purposes of demonstrating how off heap (BigMemory) can be used to extend the cache, you will modify the Integration Server Java Min and Max settings. These will not be recommended server settings but they will make it easier for the exercise to work properly in our VM that has limited resources.

   In setenv.bat, copy the lines for Java Min and Max and then comment out the first set of lines. On the uncommented settings change Java Min, Max and Perm Size to 512M. You should now have:

```
rem set JAVA_MIN_MEM=256M
rem set JAVA_MAX_MEM=1024M
rem set JAVA_MAX_PERM_SIZE=256M
set JAVA_MIN_MEM=512M
set JAVA_MAX_MEM=512M
set JAVA_MAX_PERM_SIZE=512M
```

   Keep in mind that the settings in your real environment will be higher. These settings are being used because of the available resources on the training server AND to demonstrate caching behavior along with BigMemory.

   Note: Later on you will restore the original settings.

   Save your changes.

4. Re-register the IS Windows Service

   Open a command prompt window. Change your working directory to `C:\SoftwareAG\IntegrationServer\instances\default\support\win32`.

   In the window run the installsvc command twice:

   `installsvc unreg` (this will shutdown the Integration Server and unregister the service)

   `installsvc reg` (this will register the Windows service again, but will not automatically start it)

Software AG Internal Use Only!

© 2000 – 2014 by Software AG

```
Administrator: Command Prompt
S:\SoftwareAG\IntegrationServer\support\win32>installSvc.bat unreg
Un-installing
STATUS | wrapper  | Found #include file in S:\SoftwareAG\profiles\IS\configurati
on\wrapper.conf: S:\SoftwareAG\profiles\IS/configuration/wrapper-license.conf
STATUS | wrapper  |   Loading included configuration file, S:\SoftwareAG\profile
s\IS\configuration\wrapper-license.conf
STATUS | wrapper  | Found #include file in S:\SoftwareAG\profiles\IS\configurati
on\wrapper.conf: S:\SoftwareAG\profiles\IS/configuration/custom_wrapper.conf
STATUS | wrapper  |   Loading included configuration file, S:\SoftwareAG\profile
s\IS\configuration\custom_wrapper.conf
wrapper  | Service is running.  Stopping it...
wrapper  | Waiting to stop...
wrapper  | Waiting to stop...
wrapper  | Waiting to stop...
wrapper  | Waiting to stop...
wrapper  | Software AG Integration Server stopped.
wrapper  | Software AG Integration Server service removed.

S:\SoftwareAG\IntegrationServer\support\win32>installSvc.bat
Installing
STATUS | wrapper  | Found #include file in S:\SoftwareAG\profiles\IS\configurati
on\wrapper.conf: S:\SoftwareAG\profiles\IS/configuration/wrapper-license.conf
STATUS | wrapper  |   Loading included configuration file, S:\SoftwareAG\profile
s\IS\configuration\wrapper-license.conf
STATUS | wrapper  | Found #include file in S:\SoftwareAG\profiles\IS\configurati
on\wrapper.conf: S:\SoftwareAG\profiles\IS/configuration/custom_wrapper.conf
STATUS | wrapper  |   Loading included configuration file, S:\SoftwareAG\profile
s\IS\configuration\custom_wrapper.conf
wrapper  | Software AG Integration Server service installed.

S:\SoftwareAG\IntegrationServer\support\win32>
```

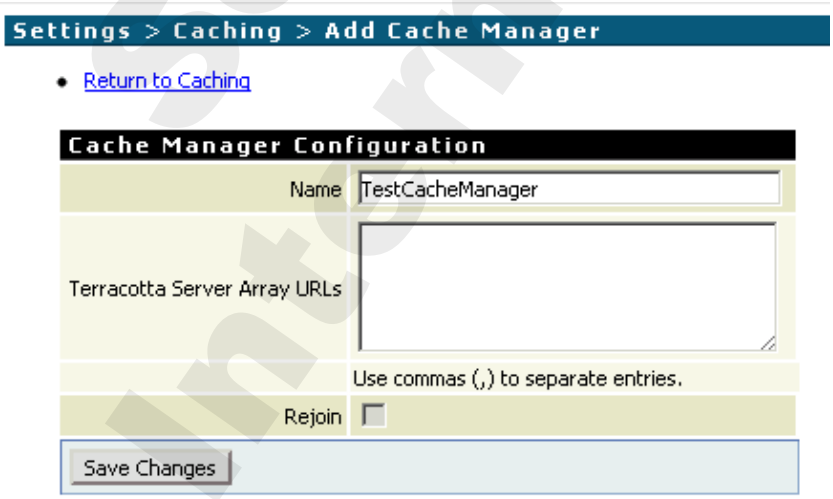
Note: when the service is re-registered, it will set it with a startup type of “Automatic”. You may wish to change this to manual startup.

5. Start the Integration server
6. After the Integration Server starts, create a new Cache Manager called **TestCacheManager** and a cache called **Data.**
   It and should hold 20000 elements in on-heap memory (Maximum Elements).

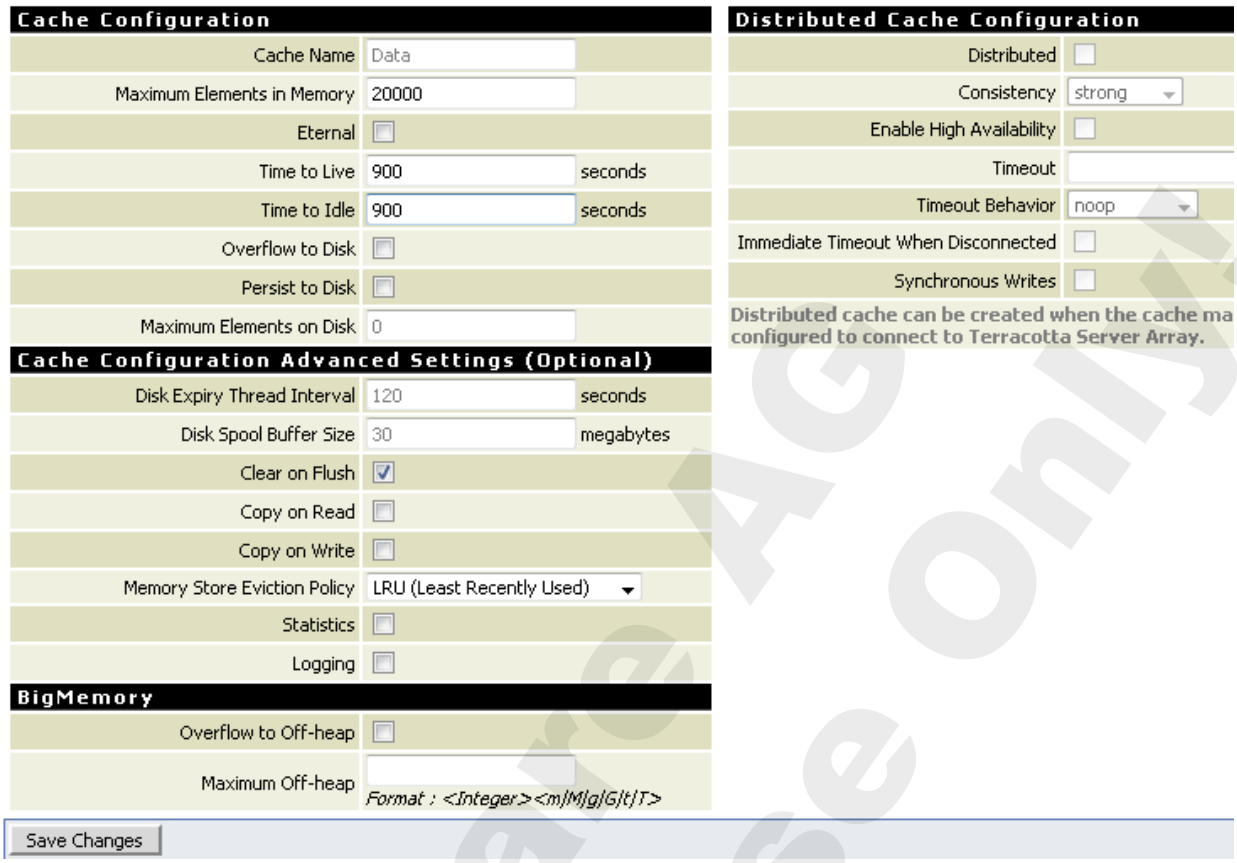
   Do not check Eternal. Set the TTL and TTI to 900 seconds.

   DO NOT Enable off heap caching to BigMemory yet.

   Use IS Administration Web UI and the following screen shots to do this:
   (save your changes when you are done)



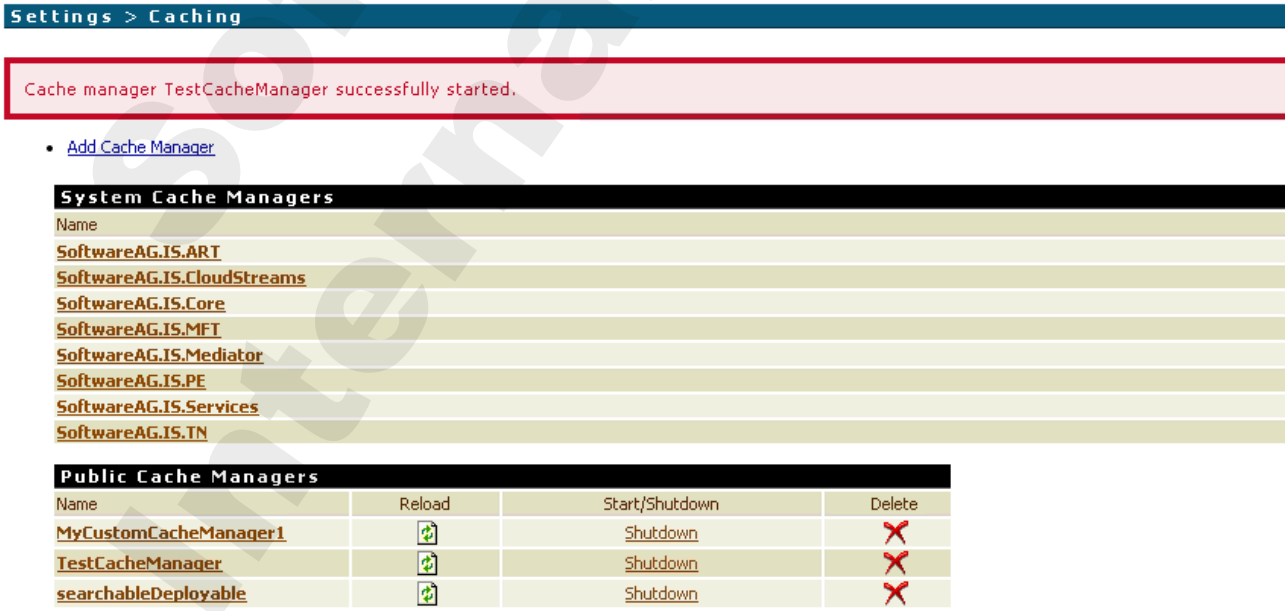
Software AG Internal Use Only!



7. After completing the above step, you should have a cache called “Data” managed by a cache manager called “TestCacheManager” that can hold 20000 elements in JVM on-heap memory (keep in mind the value stored for each key will impact how many elements can actually be stored).

   Since overflow is not enabled (to disk or BigMemory) the cache cannot overflow and cannot exceed 20000 entries.

8. Start the Cache Manager (this will enable the cache).



Software AG
Internal Use Only

© 2000 – 2014 by Software AG

9. Now you will run a service that will fill the cache with 50,000 entries (or will it ?). The value size for each entry is 25 bytes.
   Open cacheFiller in V96KnowledgeUpdateESB.BigMemory. You will notice it will generate a random value with a size of 25 bytes. This value will be stored in the cache using a generated key.

   Run cacheFiller.

   After the service completes, to prove that the cache only contains 20,000 entries, run the service listAllKeysInCache in V96TrainingESBSupport.support.caching. Provide the following input:

Name	Value
cacheManagerName	TestCacheManager
cacheName	Data

   Look in the Service Result tab, and select the keys field. Notice the last value for the keys is 19999 ?? If more entries cannot fit in the cache, the older ones will be dropped off (this is because overflow to off-heap is not enabled)

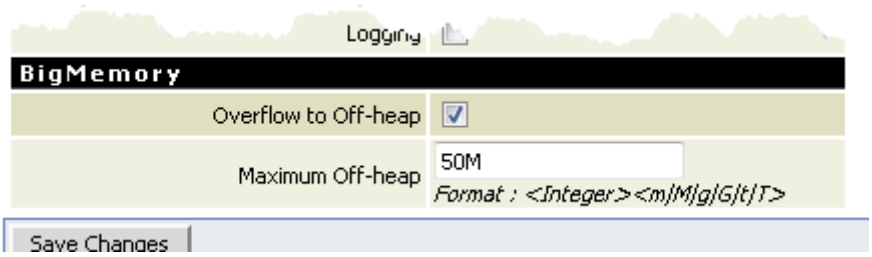
keys

Index	Value
19989	0:14:20.64-d6c4cb66-5c4d-481b-9ac5-fb2663465dda
19990	0:14:17.723-6a582761-51ca-4a0e-af9c-57dc8302c28a
19991	0:14:17.930-ab80d27c-df6c-470a-8688-a20c27d3731d
19992	0:14:21.290-e53ac656-23af-4ac9-aeed-f84a7465bbc7
19993	0:14:24.291-563679eb-d1c4-4a18-a0d7-fae0975b9d0a
19994	0:14:20.687-7428922a-1669-4cd7-8519-1c030310bb0a
19995	0:14:19.971-1116ff74-a149-4a6e-b5c2-b221cb2caf6b
19996	0:14:33.218-d1721283-1322-41dd-87cb-fd090631cb54
19997	0:14:31.193-7903d8cc-d3e3-425b-9c32-a355df5b22cf
19998	0:14:29.914-0f21cc35-7c9e-43b9-87c3-ffc6dc2a9e05
19999	0:14:24.595-2f046c37-70cc-47b4-9af7-12c38cd0d85d

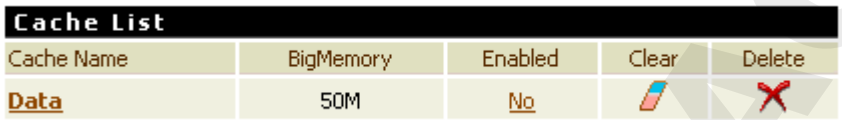
10. Next you will extend your cache to allow overflow to off-heap.
    You will set overflow to use BigMemory instead of disk.

    Shutdown the TestCacheManager so that you can make changes to the cache settings.
    Edit the **Data** cache and select **Overflow to Off-heap** under BigMemory.
    Set **Maximum Off-heap** to 50 M. This is not a realistic value for BigMemory but it is fine for this exercise. (**BigMemory can be used to have off heap storage in hundreds of GIGAbytes!)**

Software AG Internal Use Only!



Keep the max elements at 20,000.
Save your changes.
Clear the cache contents by clicking the icon under Clear.

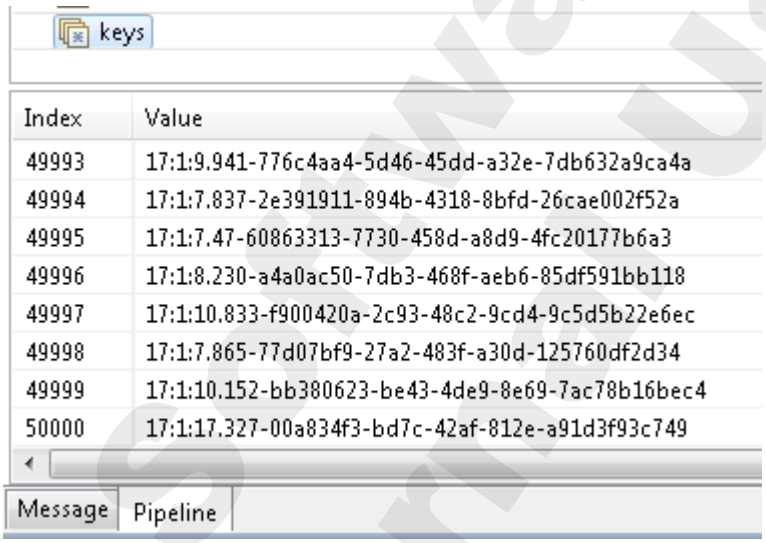


Start TestCacheManager.

Run cacheFiller service again.

After the service completes, to prove that the cache contains 50,000 entries, run the service listAllKeysInCache.

Look in the Service Result tab, and select the keys field. Notice the last value for the keys is 50000.



Internal Use Only! Software AG

© 2000 – 2014 by Software AG

EXERCISE 03: USING CACHE SEARCH

Objectives

In this exercise you will create a new cache within your custom Cache Manager to hold Customer information. The value stored in the cache will be based on a Java object called Person (as covered in the topic slides).
The person.jar file has already been copied to
C:\SoftwareAG\IntegrationServer\instances\default\lib\jars\custom
but in this exercise you will copy it into your Designer workspace.

After making the cache searchable you will review and then run some pre-built sample (Java and Flow) services to search the cache.

Steps

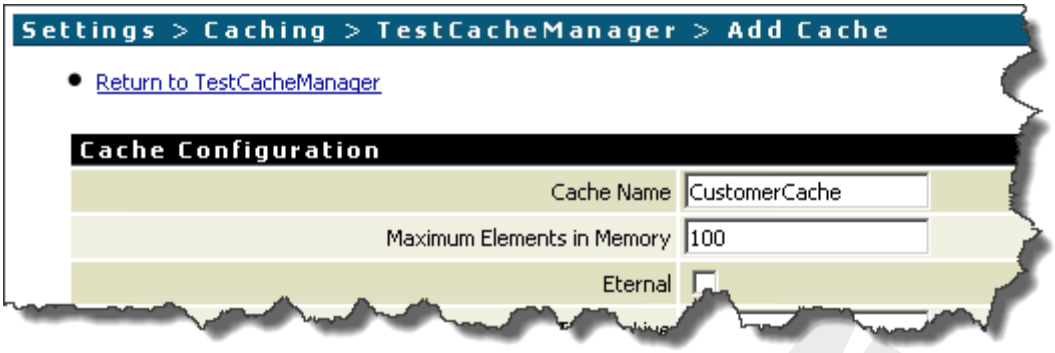
1. If your Integration Server is started (Windows service "Software AG Integration Server 9.6 (default)" shut it down.
   Now start it from the command line by opening a command prompt and changing directory to:
   C:\SoftwareAG\IntegrationServer\instances\default\bin

   To start the Integration Server so that the logs output in the console, type:
   `startup.bat –log none`

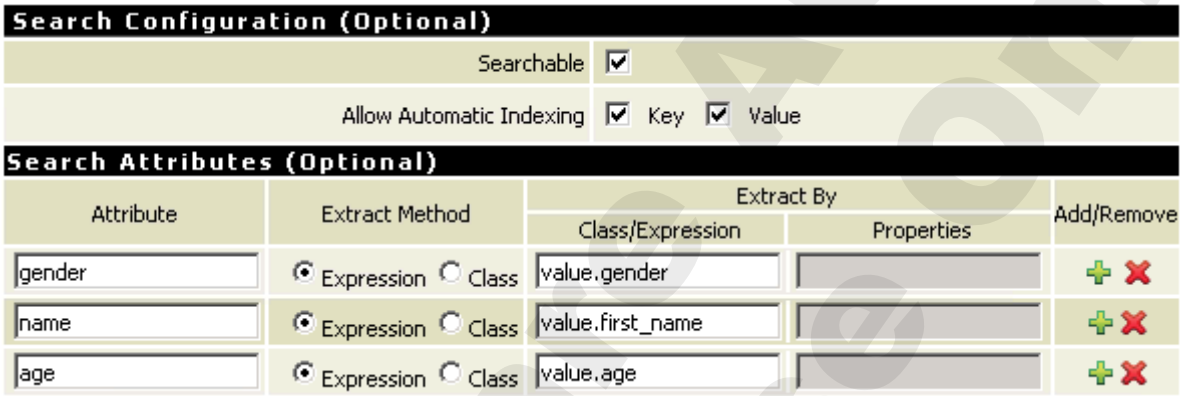
   Note: we are doing this because some of the Java samples for the Cache search using System.out.println. This output does NOT get written to the IS log (server.log). The output will only appear in the Java application console. At the end of this exercise you will switch the output back to the server.log.
2. Copy the Person.jar file from
   C:\SoftwareAG\IntegrationServer\instances\default\lib\jars\custom
   to
   C:\Users\Administrator\workspace96\V96KnowledgeUpdateESB[localhost_5555]\lib

   In Designer, open a Project Explorer view and open the project called V96KnowledgeUpdateESB[localhost_5555]. Look in the lib folder and you should see Person.jar (if you don't see it, right click on the project and select Refresh.
3. Open **FireFox browser** to the IS Administration UI and create a new cache called CustomerCache within the Cache Manager TestCacheManager.
   This cache will be used to hold customer information stored in a Java object called Person.
   Set the following values:
   Cache Name = CustomerCache
   Maximum Elements in Memory = 100 (this small number is only used for testing this exercise)

Software AG Internal Use Only!



Searchable = check this box
Set the Search Configuration and Attributes as follows:



Leave the rest of the values at their defaults and save your changes.

4. Next you will run a sample Java service to load the cache.
   In Designer open the service fillCache in namespace V96KnowledgeUpdateESB.CacheSearchDemo.setup.

   Review the logic. You will notice that it will connect to the CustomerCache (managed by TestCacheManager). It will only store 10 test customers in the cache.
   You should also notice that the Value stored in the cache is based on the Java object called Person.

   Run this service.

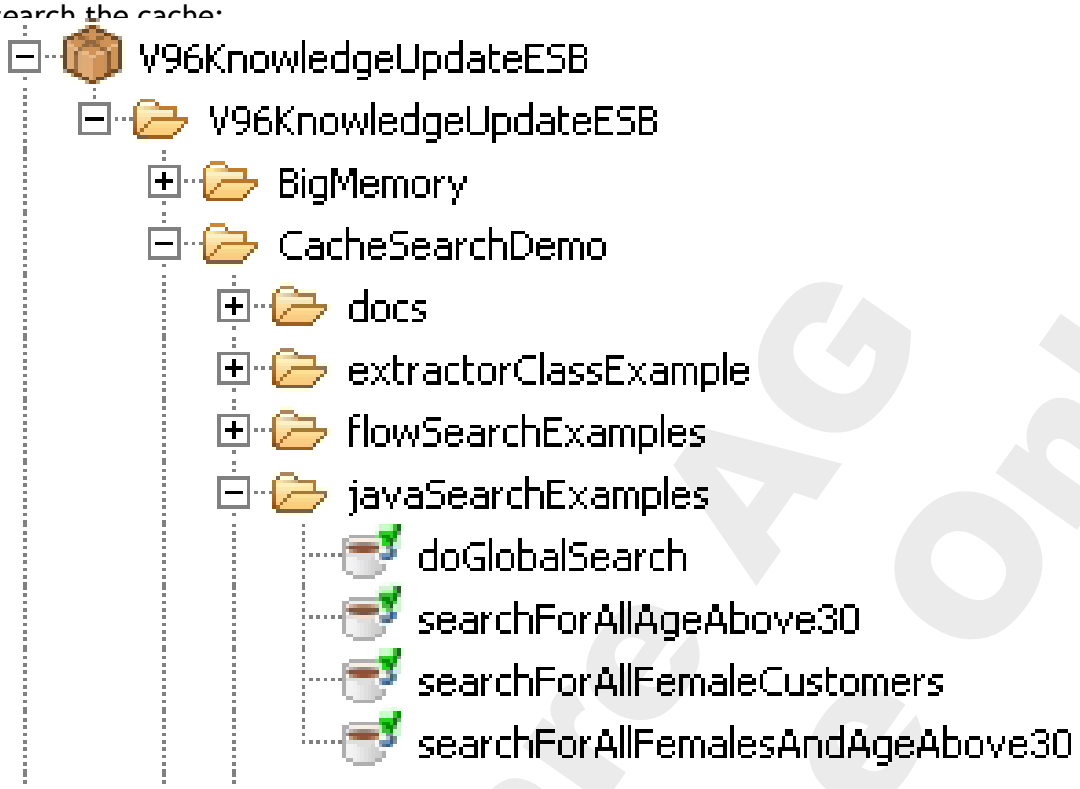
5. Look in the console for “Software AG Integration Server 9.6 (default)”. You should see the test customers that were loaded into the cache:

```
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust0,value=NAME--cust name 0 Age--25<<<
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust1,value=NAME--cust name 1 Age--41<<<
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust2,value=NAME--cust name 2 Age--29<<<
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust3,value=NAME--cust name 3 Age--24<<<
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust4,value=NAME--cust name 4 Age--24<<<
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust5,value=NAME--cust name 5 Age--42<<<
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust6,value=NAME--cust name 6 Age--13<<<
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust7,value=NAME--cust name 7 Age--48<<<
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust8,value=NAME--cust name 8 Age--50<<<
jvm 1    | >>>BasicData:cacheManagerName=TestCacheManager,cacheName=CustomerCache,key=cust9,value=NAME--cust name 9 Age--33<<<
```

Internal Use Only! Software AG

© 2000 – 2014 by Software AG

6. Now that the cache has some test data, you can use the following sample Java services to search the cache:



7. First open the service doGlobalSearch and review the logic – this service will return all entries in the cache. You will also notice that the result object returned from the cache has to be parsed (as mentioned in the topic slides).
   Note: In a real solution your service would likely parse out individual fields and put them into the pipeline for the calling service to use.

   Run this service. Look in the console window for the println output. You should see all entries returned from the cache.

```
| ///*******************RESULT*******************************///
| NAME--cust name 9 Age--33
| NAME--cust name 6 Age--13
| NAME--cust name 1 Age--41
| NAME--cust name 4 Age--24
| NAME--cust name 2 Age--29
| NAME--cust name 7 Age--48
| NAME--cust name 5 Age--42
| NAME--cust name 0 Age--25
| NAME--cust name 8 Age--50
| NAME--cust name 3 Age--24
| ///*******************RESULT*******************************///
```

8. Next open searchForAllAgeAbove30 and review the logic – this service will search the cache for entries where the age attribute is greater than 30.

   Run this service. Look in the console window for the println output. You should see all entries returned from the cache that have an age > 30.

```
| ///*******************RESULT*******************************///
| NAME--cust name 9 Age--33
| NAME--cust name 1 Age--41
| NAME--cust name 7 Age--48
| NAME--cust name 5 Age--42
| NAME--cust name 8 Age--50
| ///*******************RESULT*******************************///
```

Internal Use Only! Software AG

9. Next open searchForAllFemaleCustomers and review the logic – this service will search the cache for entries where the gender attribute is FEMALE.

   Run this service. Look in the console window for the println output. You should see all entries returned from the cache that have a gender of FEMALE.

```
| ///*******************RESULT*******************************///
| Name = cust name 9 Age = 33 Gender = FEMALE
| Name = cust name 1 Age = 41 Gender = FEMALE
| Name = cust name 7 Age = 48 Gender = FEMALE
| Name = cust name 5 Age = 42 Gender = FEMALE
| Name = cust name 3 Age = 24 Gender = FEMALE
| ///*******************RESULT*******************************///
```

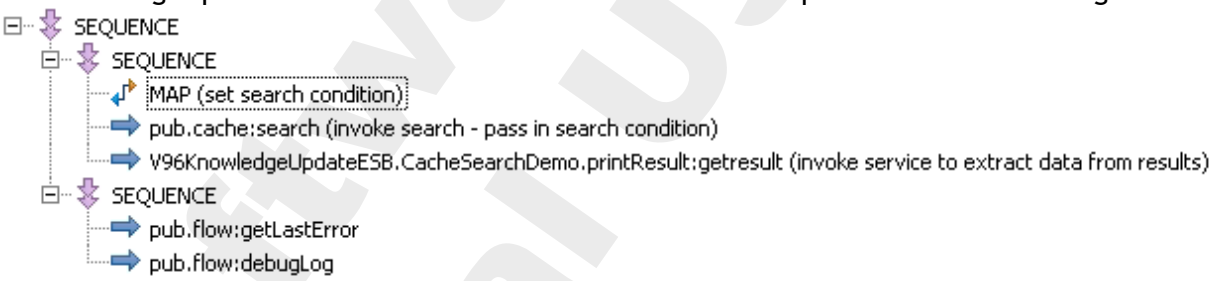
10. Next open searchForAllFemalesAndAgeAbove30 and review the logic – this service will search the cache for entries where the gender attribute is FEMALE and the age is greater than 30.

    Run this service. Look in the console window for the println output. You should see all entries returned from the cache that have a gender of FEMALE and their age is greater than 30.

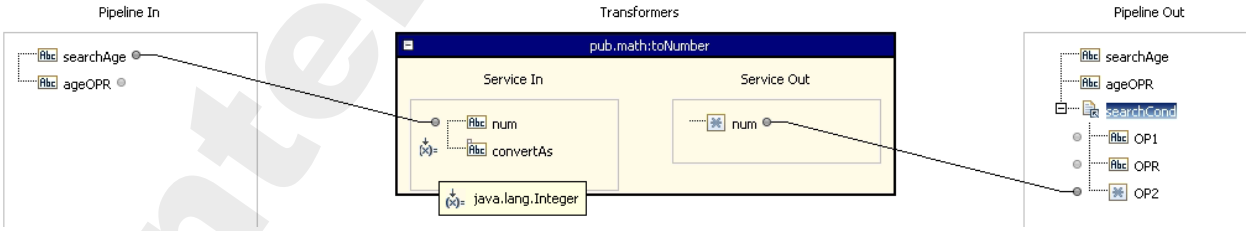
```
| ///*******************RESULT*******************************///
| Name = cust name 9 Age = 33 Gender = FEMALE
| Name = cust name 1 Age = 41 Gender = FEMALE
| Name = cust name 7 Age = 48 Gender = FEMALE
| Name = cust name 5 Age = 42 Gender = FEMALE
| ///*******************RESULT*******************************///
```

11. As mentioned in the topic slides, you can also have a Flow service call pub.cache:search. There is an example of this.

    a. Open the Flow service searchByAge in the namespace V96KnowledgeUpdateESB.CacheSearchDemo.flowSearchExamples and review the logic:



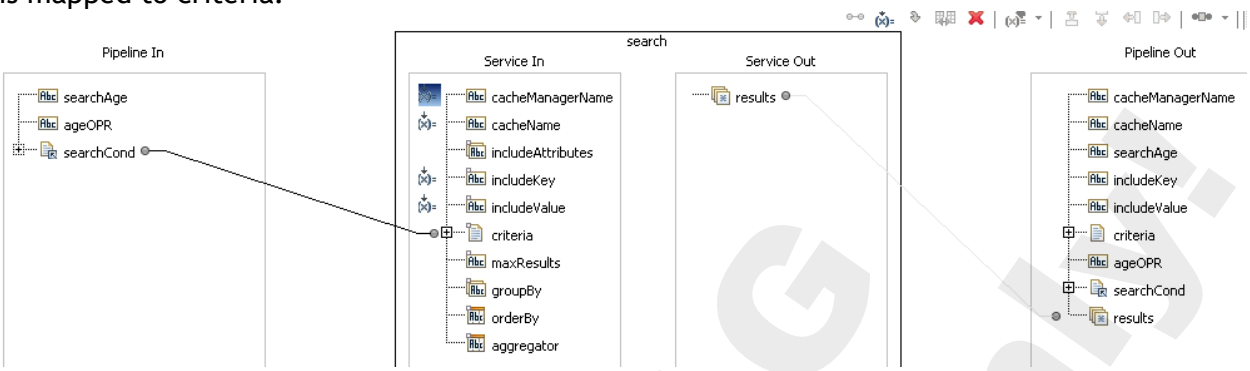
    b. Look at the MAP step. You will see that it is used to set the search condition in a document. Notice that pub.math:toNumber is used to transform the searchAge (String) entered by you as input, into an Integer type. This is very important if your OP2 field has to be an integer. In this example OP1 is set to "age", OPR is mapped with an input value you enter for operator (eg. gt). Basically the search condition is: age <ageOPR> <searchAge>.



Internal Use Only! Software AG

© 2000 – 2014 by Software AG

c. Now look at the pub.cache:search step. It points to the Cache Manager TestCacheManager and the cache CustomerCache. Notice that the document searchCond is mapped to criteria?

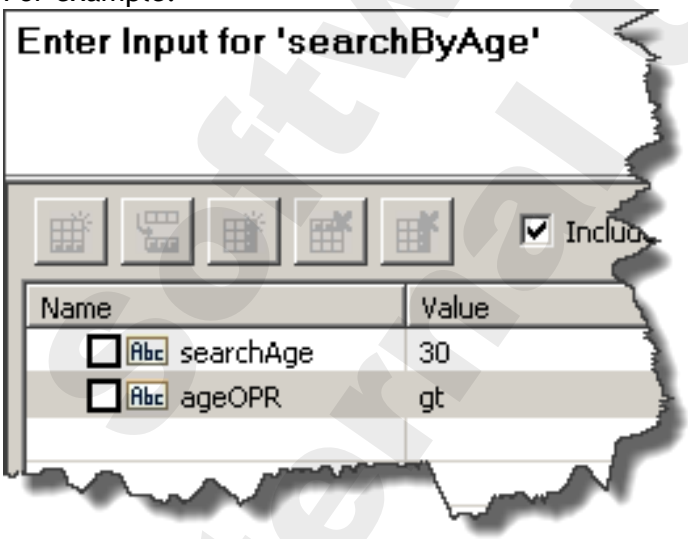


The results are returned in an object list called results. To get the data from the results you need to write a Java service to parse this object. This is done in the next line in the example.

d. Look at the next line: V96KnowledgeUpdateESB.CacheSearchDemo.printResult:getresult

This is a Java service that you have to write to parse the results object. Open getresult and review the logic. It uses the Person class to print out the results. Notice the commented out line "//IData value = …." ? This would be used if you stored an IS document (IData) in your cache value.
This parsing class has to know what type is stored in the cache value and then extract it. In a real solution this service would likely extract individual fields and put them back into the pipeline and returned to the calling service.

e. Run the Flow service searchByAge. Enter a value for searchAge and ageOPR and click OK. For example:



In the Results in Designer you will likely see an error saying "java.lang.OutOfMemoryError: Java heap space" **but the service worked**. Look in the console window for the println output. You should see all entries returned from the cache that have an age greater than 30 (in the above example).

Software AG Internal Use Only

```
///*******************RESULT-START***
NAME--cust name 1 Age--48
NAME--cust name 4 Age--49
NAME--cust name 2 Age--31
NAME--cust name 8 Age--39
///*******************RESULT-END*****
```

Note: The Software AG R&D team is investigating why this error message is appearing.

Feel free to run this service again with different search criteria.

12. When you are finished investigating and running the sample search services, change the Integration Server logging from “console” to the IS log (server.log). To make this happen, do the following:
    a. Shutdown the Integration Server (on port 5555).
    b. Re-register the Integration Server Windows service. To do this, open a command prompt window. Change your working directory to C:\SoftwareAG\IntegrationServer\instances\default\support\win32.
       In the window run the installsvc command twice:

       ```
       installsvc unreg

       installsvc reg
       ```

    c. Start the Integration Server service.

Software AG Internal Use Only!

 © 2000 – 2014 by Software AG

EXERCISE 04:
USING THE ENHANCED XML PARSER WITH BIGMEMORY

Objectives

In this exercise you will configure the Enhanced XML Parsing using the Integration Server Administrator web UI so that:

- partitions of the XML document will be cached in on-heap stored.
- partitions can be cached in the off-heap (BigMemory)
  The enhanced xml parser uses heap allocation limits to determine when to migrate partitions to a local disk store or BigMemory. The heap allocation limits enable the enhanced XML parser to throttle its use of heap.

You will also create and run services to load and query a large XML document.

Steps

1. Setup environment

Copy the folder **BigXML from C:\Training\602D-66E\Terracotta**

to

C:\SoftwareAG\IntegrationServer\instances\default\packages\V96TrainingESBSupport\pub

2. Try to load a small (800KB) XML document from the server by typing the following in your web browser:

   http://localhost:5555/V96TrainingESBSupport/BigXML/rfclistSmall.xml

3. Now create Flow service to load this XML into the pipeline.
   Call it loadXML and create it in V96KnowledgeUpdateESB.EnhancedXMLParsing.

   The service should use the legacy service loadXMLNode in WmPublic (pub.xml).

   In loadXMLNode, set the following values:
   url = http://localhost:5555/V96TrainingESBSupport/BigXML/rfclistSmall.xml
   auth/user = Administrator
   auth/pass = manage

Software AG Internal Use Only!

Run this service. The file rfclistSmall.xml should load successfully. You should see the XML node object in the pipeline

Name	Value
url	http://localhost:5555/V90TrainingESBSupport/BigXML/rfclistSmall.xml
auth	
user	Administrator
pass	manage
node	com.wm.lang.xml.Node:-1:1696933663

4. Now change the url to load a larger (220MB) XML file (**rfclist.xml** in the same folder).

   The load should fail with an OutofMemoryError:

```
Launch started: 2014-06-16 13:00:55.13
Configuration name: loadXML_deletelater
Configuration location: C:/Users/Administrator/workspace96/.metadata/.plugins/org.eclipse.debug.core/.launches/loadXML_deletelater.launch

Could not run 'loadXML_deletelater'
com.wm.app.b2b.server.ServiceException: com.wm.app.b2b.server.ServiceException: java.lang.OutOfMemoryError: Java heap space

com.wm.app.b2b.server.ServiceException: com.wm.app.b2b.server.ServiceException: java.lang.OutOfMemoryError: Java heap space
    at pub.xml.loadXMLNode(xml.java:378)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:606)
    at com.wm.app.b2b.server.JavaSvcHandle.baseInvoke(JavaService[illegible]:443)
```

5. Now you will configure the Enhanced XML Parsing on the Integration Server and then use the new parser service loadEnhancedXMLNode.
   This will enable your Integration Server service to handle very large XML processing by using off heap storage (e.g. Big Memory) AND without writing custom code.
6. First, increase the amount of MaxDirectMemorySize in the setenv.bat file to 1100M.

```
set JAVA_MAX_DIRECT_SIZE=-XX:MaxDirectMemorySize=1100M
```

After changing setenv.bat, reinstall the IS Windows service (see above for instructions).

Start the Integration Server service.

7. Create a Flow service called loadBigXML in V96KnowledgeUpdateESB.EnhancedXMLParsing.

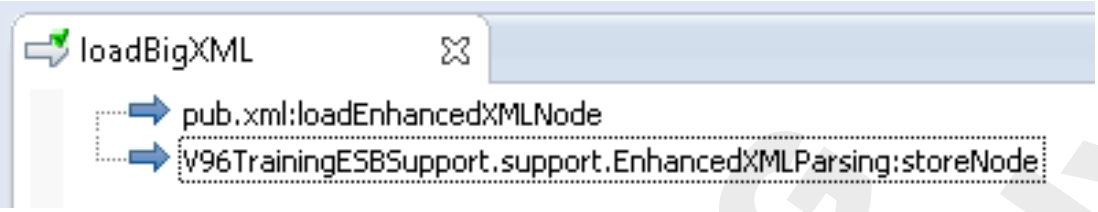
   This service should:

   a. invoke the new XML parser called loadEnhancedXMLNode to load the big XML file (rfclist.xml). In loadEnhancedXMLNode, set the following values:
      url = http://localhost:5555/V96TrainingESBSupport/BigXML/rfclist.xml
      auth/user = Administrator
      auth/pass = manage

Software AG Internal Use Only!

© 2000 – 2014 by Software AG

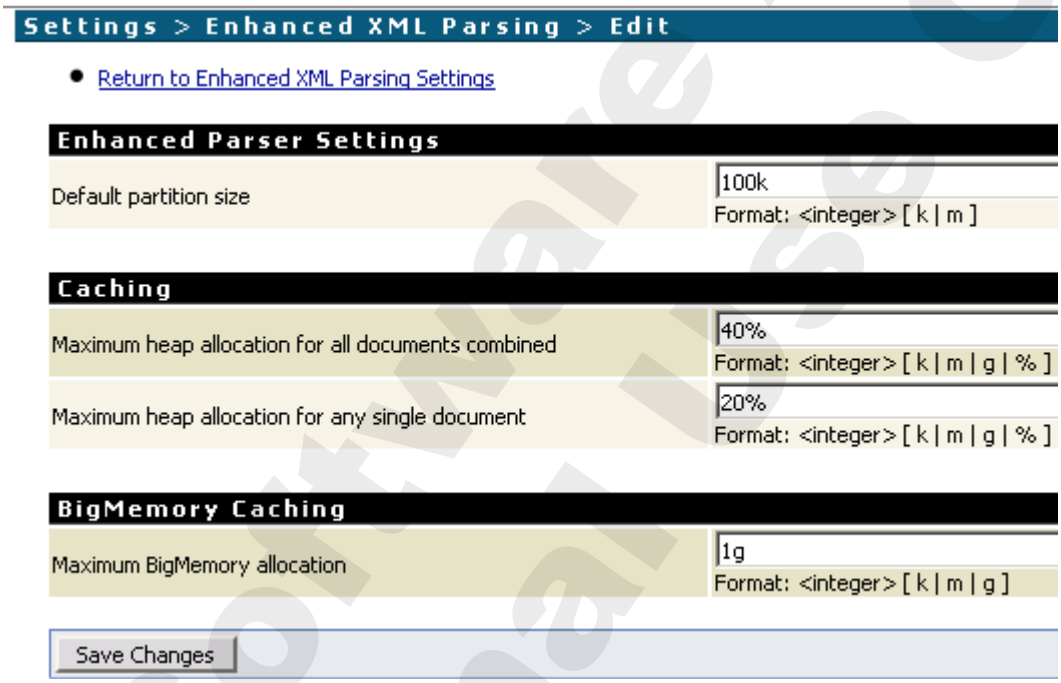
b. invoke the service storeNode in V96TrainingESBSupport.support.EnhancedXMLParsing (node gets auto mapped in this service)
Note: the call to the java service storeNode stores the node that was created by loadEnhancedXMLNode, into a static Java variable. You will re-use later for an XML Query.



8. Next configure the IS to use "Enhanced XML Parsing" so that it uses Caching and BigMemory.

In the IS Administration Web UI go to: Settings > Enhanced XML Parsing

Edit the settings using the screenshot below as guidance.
Save your changes



Enable Caching (click the No link to change to Yes) **-** this will enable the Enhanced XML Parser to use Ehcache. **Note: if you don't enable caching the enhanced XML parser will behave like the legacy parser.**

Enable Big Memory Caching (click the No link to change to Yes) - this will enable the Enhanced XML Parser to off-heap (BigMemory). **Note: if you don't enable Big Memory caching, the local disk will be used for off-heap partitions.**

You should now see:

Software AG Internal Use Only!

Enhanced Parser Settings	
Default partition size	100k

Caching	
Enabled	✔ Yes
Maximum heap allocation for all documents combined	40%
Maximum heap allocation for any single document	20%

BigMemory Caching	
Enabled	✔ Yes
Maximum BigMemory allocation	1g

Peak Usage Statistics	
Peak heap allocation for all documents combined	0m
Peak heap allocation for any single document	0m
Peak BigMemory use	0m

9. Now run the service loadBigXML to load your big document. The load process takes about 40-60 seconds, depending on your machine.

   Look at the Peak Usage Statistics in the IS Administration Web UI. **Note**: you will have to refresh the IS Admin web page to see the updated Peak Usage stats.

Enhanced Parser Settings	
Default partition size	100k

Caching	
Enabled	✔ Yes
Maximum heap allocation for all documents combined	40%
Maximum heap allocation for any single document	20%

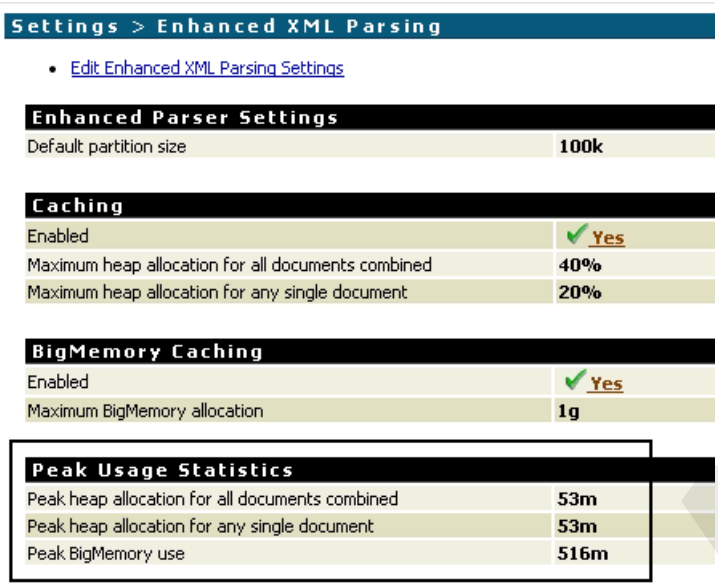
BigMemory Caching	
Enabled	✔ Yes
Maximum BigMemory allocation	1g

Peak Usage Statistics	
Peak heap allocation for all documents combined	84m
Peak heap allocation for any single document	82m
Peak BigMemory use	487m

   The Peak Usage stats will show you “Peak BigMemory use” while the XML document was being parsed by the enhanced XML parser.

Software AG Internal Use Only!

© 2000 – 2014 by Software AG



Notice there was no extra coding to load this large XML file? Before the v9.0 release, developers would have to uses services like pub.xml:getXMLNodeIterator and pub.xml:getNextXMLNode to process large XML files.

10. Once you successfully loaded the document, create a new Flow service called queryXML in V96KnowledgeUpdateESB.EnhancedXMLParsing.
    Since the XML node is enhanced if the entry is stored in BigMemory it will be retrieved.

    This service should:

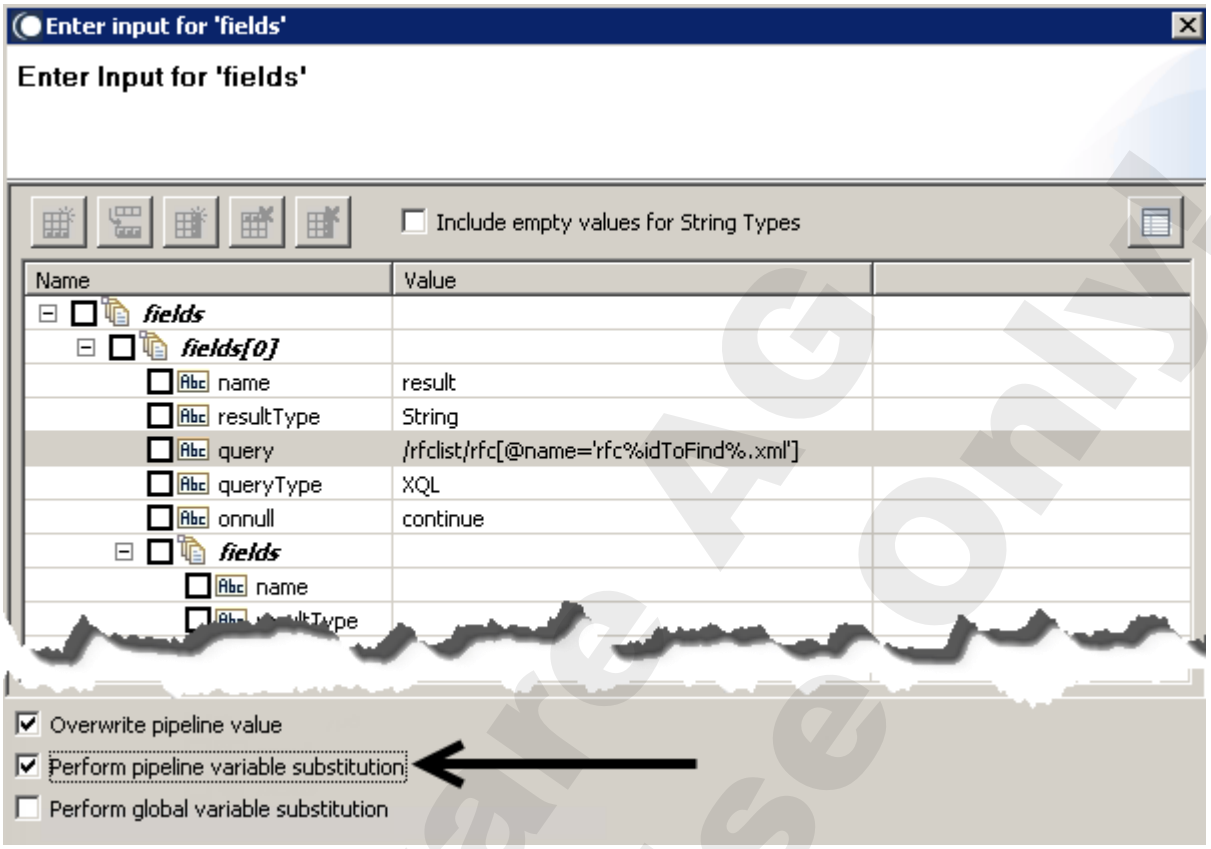
    a. have an String input field called idToFind

    b. invoke the service getLastNode in V96TrainingESBSupport.support.EnhancedXMLParsing
       This service will get the XML node that was stored by the Java service (above).
       node will be auto mapped.

    c. invoke the service queryXMLNode in WmPublic pub.xml
       Set the following values:
       fields[0]/name = **result**
       fields[0]/resultType = **String**
       fields[0]/query = **`/rfclist/rfc[@name='rfc%idToFind%.xml']`**
       fields[0]/queryType = **XQL**
       fields[0]/onnull = **continue**

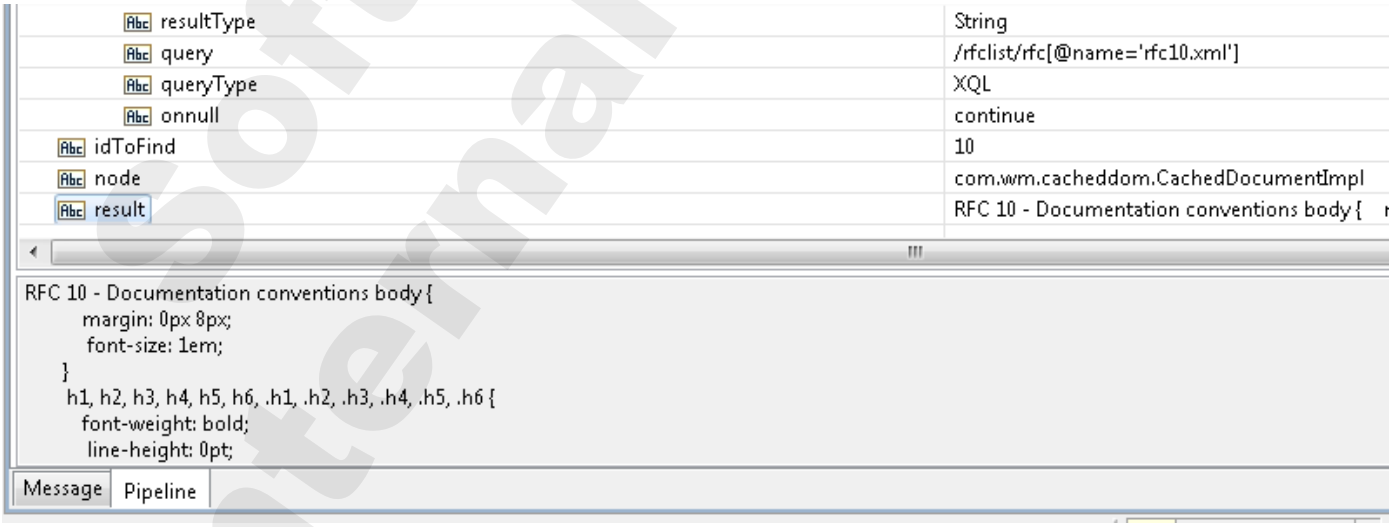
Software AG Internal Use Only!

d. Make sure you also check the box beside **Perform pipeline variable substitution**



11. Run the queryXML service using the input value of 10.

After the service completes, click on the result field. You should see the specific RFC that you searched for (e.g. RFC 10 ---> rfc10.xml) within the stored XML node.



Run the service again using the input value of 100.

Note: As mentioned previously, depending on the memory availability for the Integration Server, the enhanced XML parser will use Ehcache to move already parsed portions of the document to a local disk store or off-heap cache (BigMemory).

Software AG Internal Use Only

© 2000 – 2014 by Software AG

By moving this data, the enhanced XML parser frees up heap space for other documents or other system processes. The ability to move data to **BigMemory or** a local disk store enables the parser to parse multiple XML documents, including multiple large XML documents, without monopolizing the JVM heap.

As mentioned above, if you are curious to know how much BigMemory is being used for enhanced XML parsing on the Integration Server, open the IS Admin web UI, and navigate to Settings > Enhanced XML Parsing. The Peak Usage Statistics are available at the bottom:

Peak Usage Statistics	
Peak heap allocation for all documents combined	96m
Peak heap allocation for any single document	96m
Peak BigMemory use	473m

Software AG Internal Use Only!

This Page intentionally left blank

Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 05:
JSON SUPPORT IN INTEGRATION SERVER

Objectives

You will review and run a service that uses new JSON services in WmPublic.

The service reads a sample file in JSON format and converts it to an IS document (IData).

This service will then convert the document to a String field containing XML and a String field containing JSON.

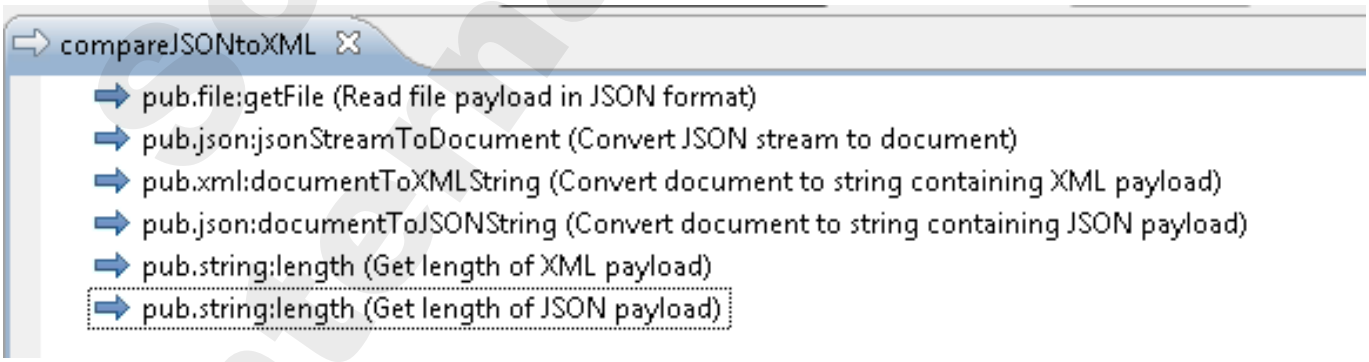
Finally, it will calculate the length of each strings to see how much "lighter" JSON payload is versus XML.

Steps

1. Before starting this exercise free up some resources in the training VM. Do the following:
   a. Turn off Big Memory in setenv.bat – to do this comment out (rem) the line with MaxDirectMemorySize
   b. Reset the Java min, max, and max perm size to the initial values in setenv.bat
   c. Remember to unreg and re-reg the IS service as you did before.
2. Start the Integration Server:

   Open a browser and try to load the following input file and verify that it is contains valid JSON data

   http://localhost:5555/V96TrainingESBSupport/json/SoftwareAGRIPEEntry.json
3. Open and review the service compareJSONtoXML in V96KnowledgeUpdateESB.JSON.



pub.file:getFile - is reading a file containing JSON payload into the pipeline as a stream.

pub.json:jsonStreamToDocument - is a new service in WmPublic. It is converting a stream (containing JSON) to an IS document (IData).

pub.xml:documentToXMLString – converts the document in the pipeline to a string containing XML called xmldata.

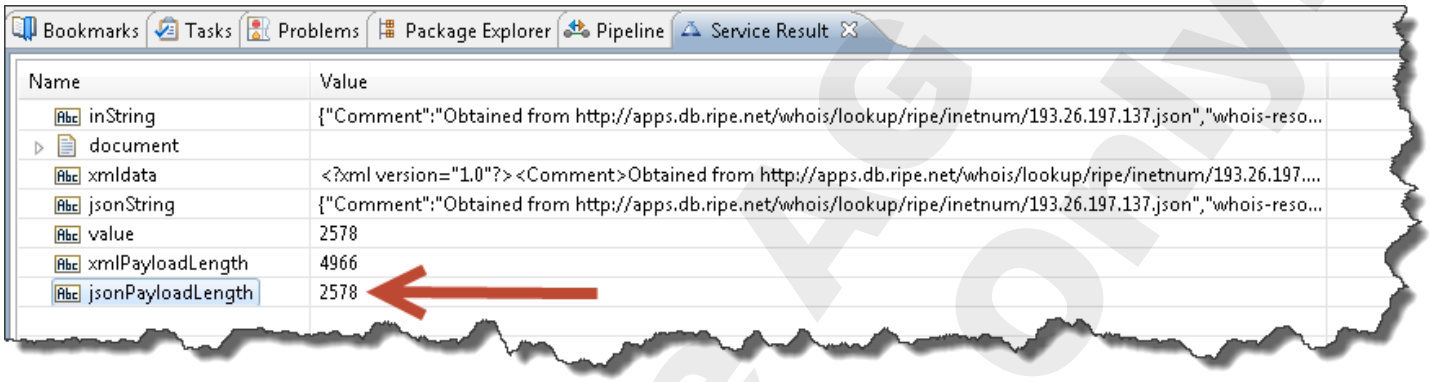
Software AG Internal Use only!

pub.json:documentToJSONString – is a new service in WmPublic. It is converting the document in the pipeline to a string containing JSON called jsonString.

pub.string:length – is calculating the length of both strings so that you can compare.

After reviewing this service, run it.

4. Review the values in the pipeline after running the service.
   Notice that the JSON payload is almost ½ the size of XML ?



Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

Exercise 06: Using Global Variables

Objectives

This is a simple exercise so that you can create and use Global Variables.

You will create global variables using the IS Administration Web UI, and then use them in a Flow service.

Steps

1. Make sure the Integration Server service is started:

Software AG Integration Server	Software A...	Started

2. Open the IS Administrator web UI in a browser (http://localhost:5555).
3. Navigate to Settings > Global Variables.
4. Add a couple variables.
5. Create a simple Flow service using Designer in the GlobalVariables folder within your package namespace.
   The flow service can invoke debugLog.
   In the message field for debugLog reference your Global Variable by wrapping them in percent signs. Make sure you check the box beside “Perform global Variable Substitution”.
6. Run your service and look at the fields in the pipeline. Since you used debugLog you can also see your substituted variables in the IS log.

Software AG Internal Use Only!

This Page intentionally left blank

Software AG
Internal Use Only!

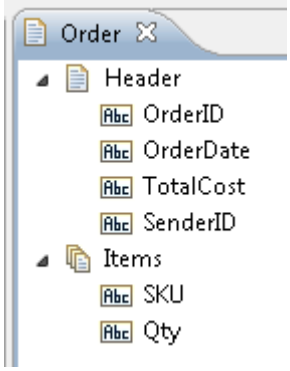
© 2000 – 2014 by Software AG

Exercise 07: Use IDataMap in a Java Service

Objectives

In this exercise you will create a Java Service and add a simple Document Type (reference) as input.

You will then generate the Java code, which will use the old style logic to retrieve the IData variables using the pipeline cursor.
After examining the generated code in an editor, you will add Java code to your Java service and use the new IDataMap class along with the getNested method to read the variables from the pipeline in the Order document. Here is the Order document:



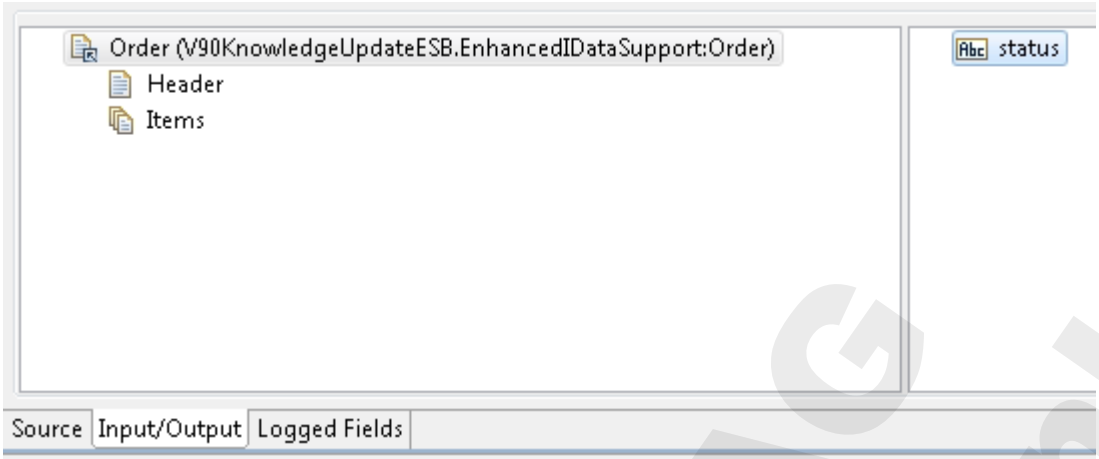
At the end of this exercise you should have an idea of how much simpler, cleaner and easier the Java coding will be for handling Documents (especially documents with a lot of nested variables).

Steps

1. Open Software AG Designer
2. Create a Java service called **useIDataMap** in **V96KnowledgeUpdateESB.EnhancedIDataSupport**.
3. Create the service signature.
   For the Input, add a Document Reference to the Order document

Software AG Internal Use Only!

For the Output, add a String called status.



4. Right click on **useIDataMap** and select **Generate Code…** and then **For implementing this service.**
   As you know the code gets copied into the Windows clipboard.

   Open Notepad or Notepad++ and paste in the code. This should look familiar to you. It uses a lot of getCursor. For a document with a lot of nested data the code can get quite messy.

5. Type the following Java code shown into your Java service:

```
IDataMap pm = new IDataMap(pipeline);

IDataMap order = new IDataMap (pm.getAsIData("Order"));

String orderID = (String) order.getNested("Header","OrderID");
String orderDate = (String) order.getNested("Header","OrderDate");
String totalCost = (String) order.getNested("Header","TotalCost");
String senderID = (String) order.getNested("Header","SenderID");

IData[] items = order.getAsIDataArray("Items");

for (IData item : items) {

   IDataMap im = new IDataMap(item);
   String sku = im.getAsString("SKU");
   String qty = im.getAsString("Qty");
}

pm.put("Status", "complete");
```

   As you can see the new Java class IDataMap and the getNested method are used. The put method is also used to put one variable (status) into the pipeline.

   Notice there are no getCursor and no destroy methods ? There is no error handling in the sample but you can still see how much simpler, cleaner and easier the Java coding will be for handling Documents.
   After you type in the code, save it and run it (or debug it).

   Note: when you save your changes you may see an error because IDataMap cannot be found. If you click on the first red x to the left you should get a pop-up. Select “Import ‘IDataMap’ (com.softwareag.util)”

Software AG
Internal Use Only!

© 2000 – 2014 by Software AG

EXERCISE 08: PUBLISH / SUBSCRIBE WITH INTEGRATION SERVER & UNIVERSAL MESSAGING PROVIDER FOR JMS

IN THIS EXERCISE YOU WILL CREATE THE NECESSARY COMPONENTS ON THE INTEGRATION SERVER TO SEND AND RECEIVE A MESSAGE ASYNCHRONOUSLY VIA A TOPIC ON UNIVERSAL MESSAGING PROVIDER FOR JMS

Objectives

- Confirm necessary JNDI and JMS Connection aliases exist for the Universal Messaging realm.
- View the destinations available in Universal Messaging realm.
- Create a service to send a JMS Message via a topic.
- Create a handling service to receive a JMS Message.
- Create a JMS trigger to receive the JMS Message from the topic and pass it to the handling service.
- Optional: Use Universal Messaging Enterprise Manager to “snoop” the topic.

Steps

1. Make sure the Integration Server and Universal Messaging realm services are started:
   Software AG Integration Server 9.6 (default)
   (do not start the server with “(2)” at the end)

   Software AG Universal Messaging 9.6.0.0.12476 (umserver) 1
2. Review the JNDI and JMS aliases that are configured.
   Use the IS Administration web UI and navigate to Settings > Messaging.

   The following aliases have been pre-created for you:
   The JNDI alias is: UniversalMessagingJNDIalias (click the Test Lookup button)

   The JMS alias is: UniversalMessagingSampleJMSConnAlias (enable this alias)
3. Open Universal Messaging Enterprise Manager from Windows Start menu: Start > All Programs > SoftwareAG > Administration > Universal Messaging Administration 9.6 > umserver

   Click on the umserver realm and then the JNDI tab.
   Change nsp://localhost:9000 to nsp://sagbase:9000 and click the Apply button.
   Most of the Connection Factories, Topics and Queues you see are being used by various webMethods products. We will be using the Connection Factory called SampleConnectionFactory.

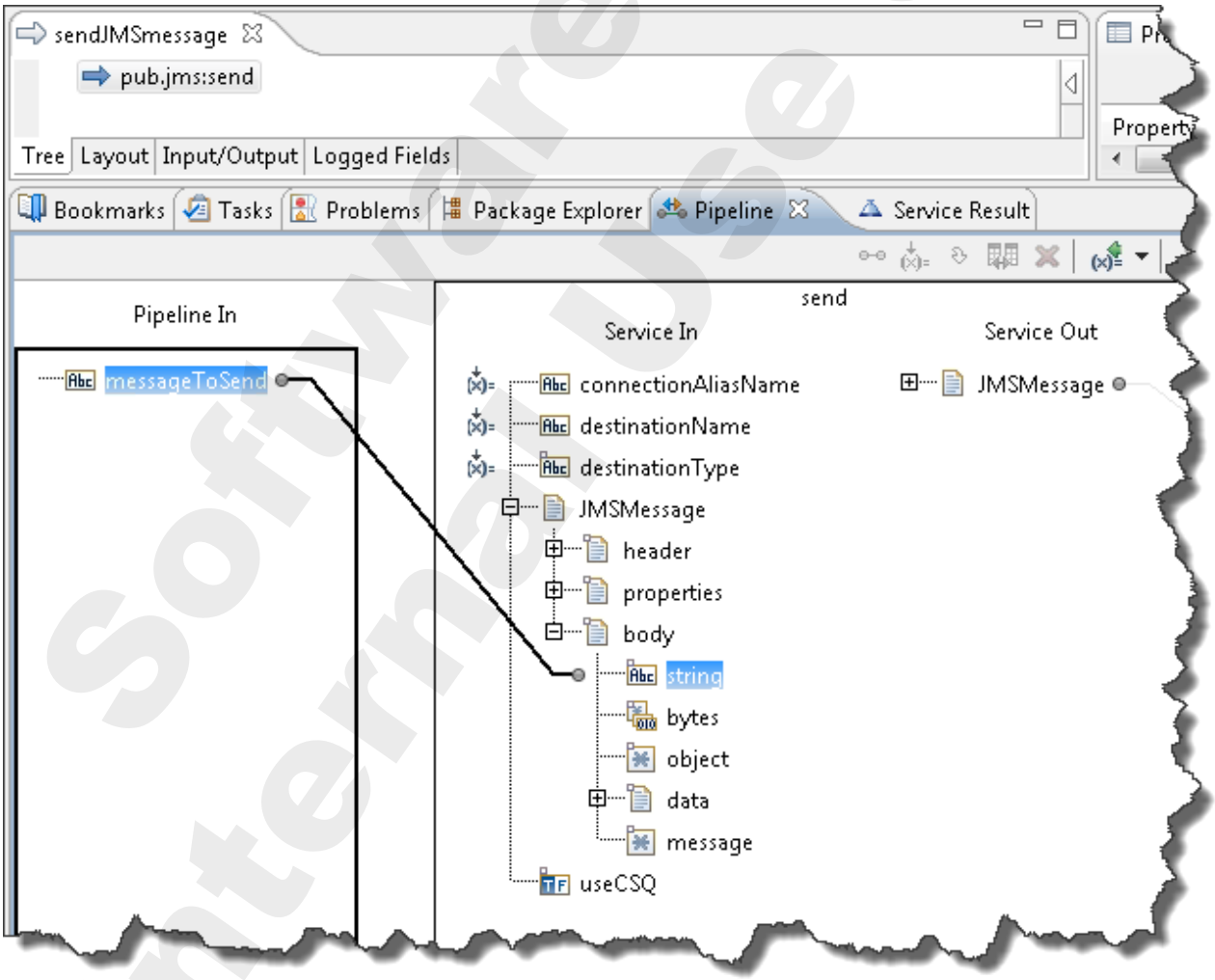
Software AG Internal Use Only!

Make sure topic TestTopic1 exists. If it does not exist create it by right clicking the Topics folder and selecting New Topic.

You will do pub/sub with TestTopic1.

4. Create a Flow service called sendJMSmessage in namespace V96KnowledgeUpdateESB.PubSubWithUM
   This service will be used to send a JMS message to a Universal Messaging Topic called TestTopic1.
   Modify sendJMSmessage as follows:
   a. Add a String input field called messageToSend
   b. Invoke pub.jms:send and set the following values:
      connectionAliasName = UniversalMessagingSampleJMSConnAlias
      destinationName = TestTopic1
      destinationType = TOPIC
   c. Map pipeline variable messageToSend to JMSMessage\body\string

   Save the service.



Software AG Internal Use Only!

© 2000 – 2014 by Software AG

5. Next you will create a Flow service called handleJMSmessage in namespace V96KnowledgeUpdateESB.PubSubWithUM

   This service will handle (receive) the JMS message that is received from the JMS Trigger. Modify handleJMSmessage as follows:

   a. In the input for this service set the specification reference to: pub.jms:triggerSpec

   b. Map the JMSMessage/body/string to the message field in pub.flow:debugLog so that you can see the message received in the Integration Server log.

      Save the service.

6. Now you can create a JMS trigger called receiveJMSmessageFromTestTopic1 in namespace V96KnowledgeUpdateESB.PubSubWithUM

   Configure this trigger as follows:

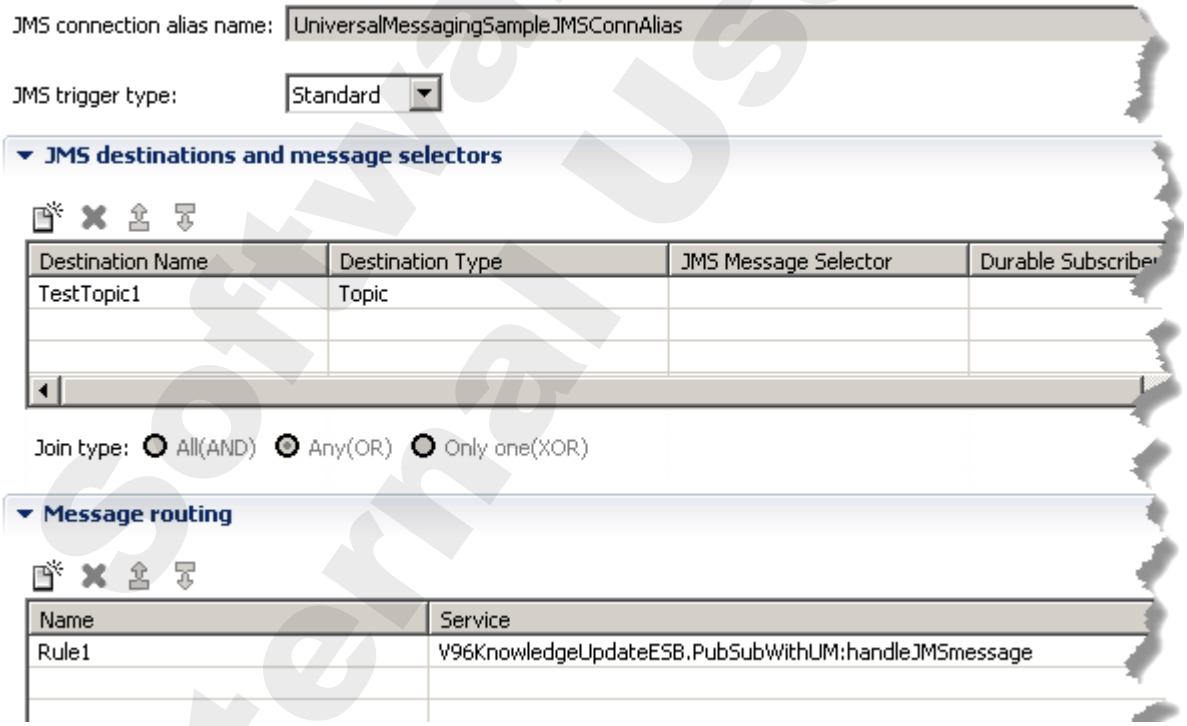
   a. Set the JMS connection alias name to: UniversalMessagingSampleJMSConnAlias

   b. Leave JMS trigger type as: Standard

   c. Select destination TestTopic1

   d. Set the handling service in the Messaging Routing section to: V96KnowledgeUpdateESB.PubSubWithUM:handleJMSmessage

      Save the trigger.



7. You now have all the pieces in place to do publish / subscribe.

Software AG
Internal Use only!

8. Run the service sendJMSMessage and enter something in your input field.
   As soon as the message arrives in TestTopic1, the JMS trigger should receive it, and pass it to the handling service handleJMSmessage.

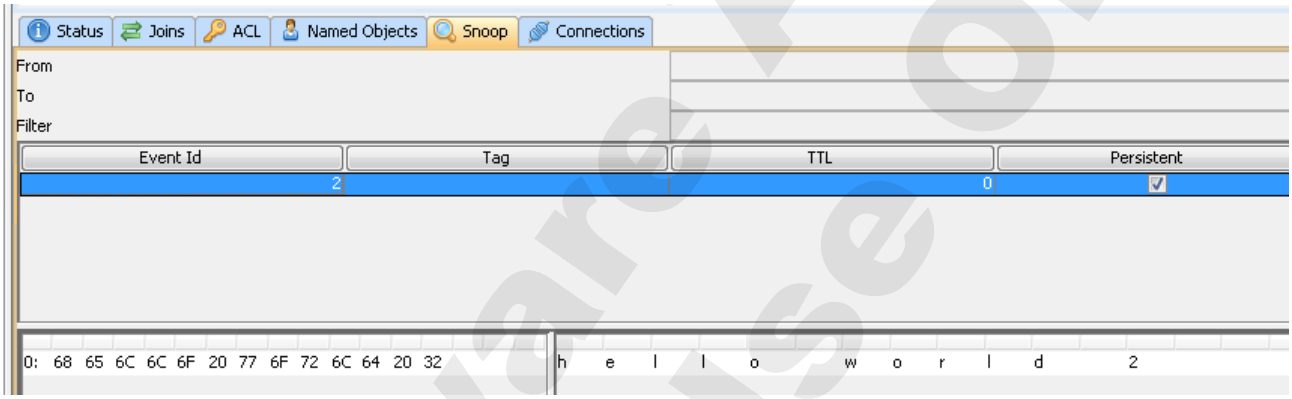
   After running the service look in the IS log and you should see your message.

Optional Bonus steps:

9. Snoop the messages in the destination.

   In Universal Messaging Enterprise Manager select the “snoop” tab on TestTopic1. Click the Start button.

   Run the sendJMSmessage service again. Verify that you see your message from the handling service in the IS log and look at the snoop tab in Enterprise Manager. You can see your message. This is handy for debugging.



Remember to Stop the snoop after you are done.

Software AG
Internal Use Only!

© 2000 – 2014 by Software AG