Using Rhapsody 4.01 with WebLogic JMS for 10.3

Michael.W.Czapski@gmail.com

December 2011

Table of Contents

Introduction

I had an occasion, recently, to work on an integration project which required the Rhapsody 4.01-based integration solution to receive messages from a WebLogic-based JMS Topic. Product documentation and Internet searches did not offer assistance in terms of how the Rhapsody JMS Adapter needs to be configured to support this. While there are a number of articles which discuss the topic of configuring JMS Client to interact with WebLogic JMS Server, none of the solutions described in these articles worked for me. A degree of experimentation and creative adaptation resulted in a working configuration. This article discusses this solution for the benefit of these who will be faced with this problem and for my own benefit if I need to do this again in the future.

On the WebLogic Side

In the olden days one would need to collect a specific set of WebLogic JAR archives, needed to support JMS integration, and make them available to the client application. My article on configuring the QBrowser JMS Browser for interaction with WebLogic JMS, "Using QBrowser v2 with WebLogic JMS for 10.3", at http://blogs.czapski.id.au/2011/05/using-qbrowser-v2-with-weblogic-jms-for-10-3, is an example of how this would have been done in the olden days. The method seems to have changed between then and now. The new method involves explicit creation of a client JAR, **`wlfullclient.jar`**, discussed amongst others in "Using the WebLogic JarBuilder Tool", http://docs.oracle.com/cd/E12840_01/wls/docs103/client/jarbuilder.html.

To summarise:

On WebLogic host, change directory to the WebLogic Server's server/lib directory.

```
cd $WL_HOME/server/lib # or cd %WL_HOME%¥server¥lib or some such
```

Use the following command to create wlfullclient.jar in the server/lib directory:

```
java -jar wljarbuilder.jar
```

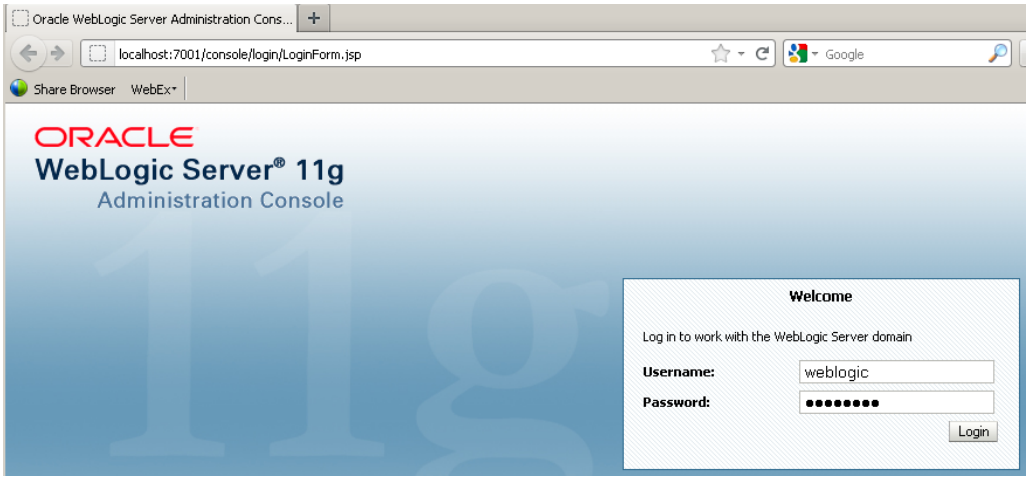
Copy the resulting **`wlfullclient.jar`** to the client application development area and bundle it with the client application, however these need be done for a particular client application. Add the wlfullclient.jar to the client application's classpath, however that needs be done for the particular client application. *(Notes on how to do this for the Rhapsody JMS Adapter are provided later in this document).*

The above gives you the Java Archive containing all Java classes necessary to support JMS interaction between the client application and the WebLogic Server JMS.

To make the discussion more concrete, let's assume that we have a JMS Topic Connection Factory, with a JNDI reference of "jms/MyTopicCF" and a JMS topic with a JNDI reference of "jms/MyTopic".

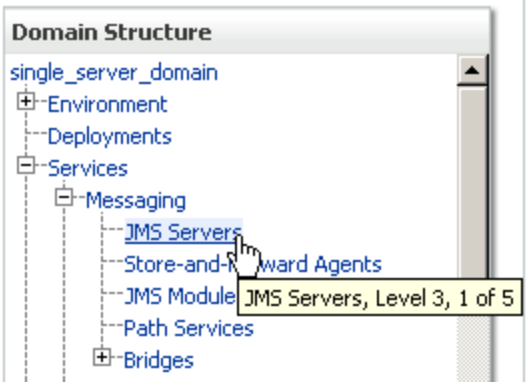
The following steps walk through the process of creating these objects using the WebLogic Admin Console.

☐ Log into the WebLogic Admin Console, http://{WLJMSHost}:{WLJMSPort}

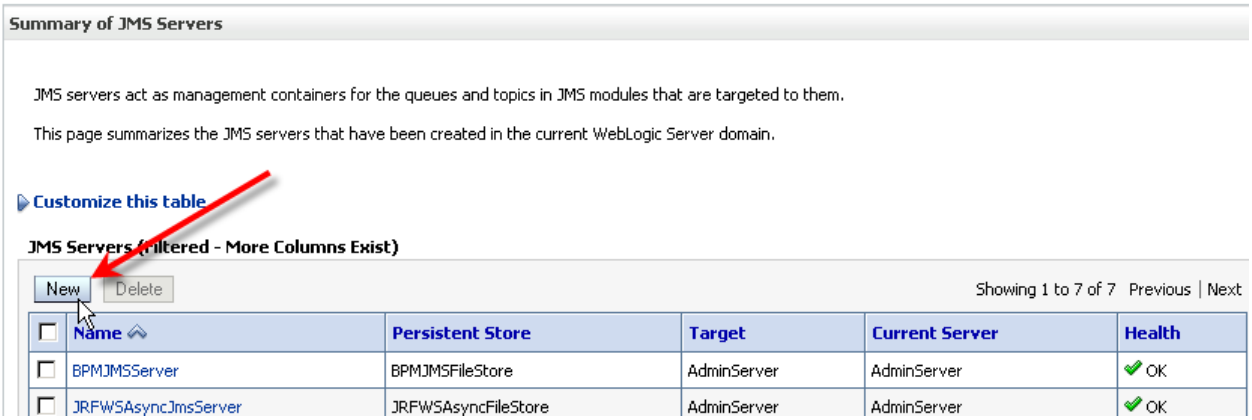


Create JMS Server

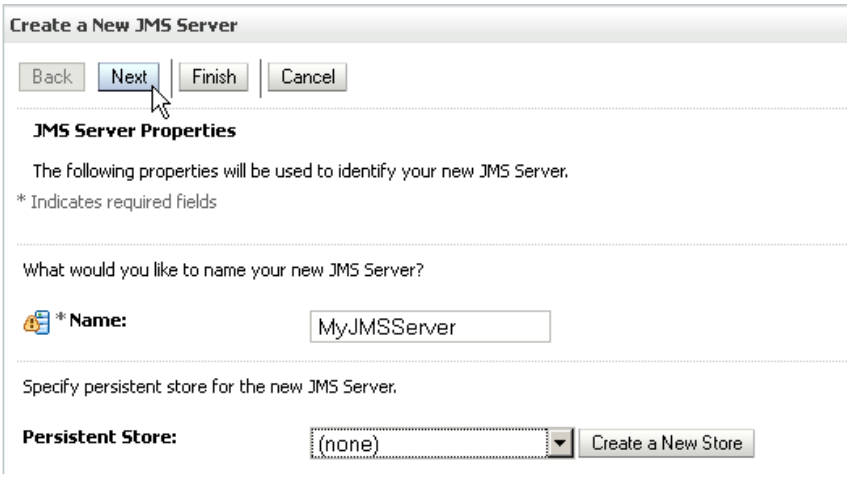
☐ Navigate the hierarchy "Services"→"Messaging" and click the "JMS Server" node



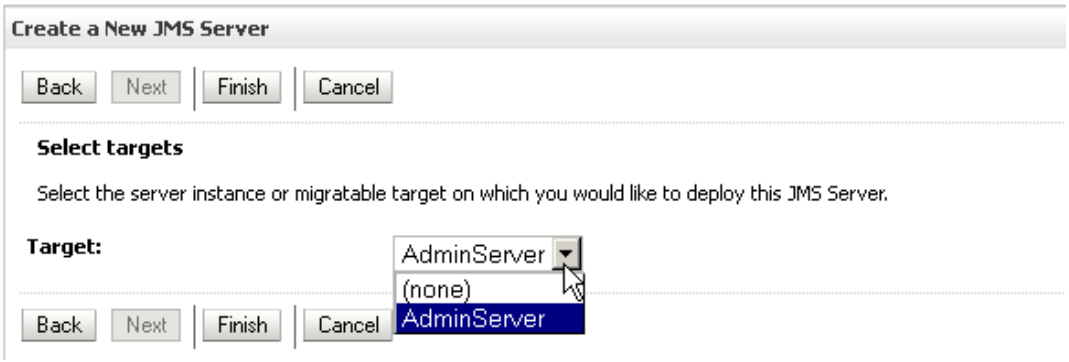
☐ Click "New" (This is not really necessary if there is an existing JMS Server which can be used. I don't know what your environment looks like so I am assuming you don't have a convenient JSM Serve to use)



☐ Enter "MyJMSServer" as new server name and click "Next"



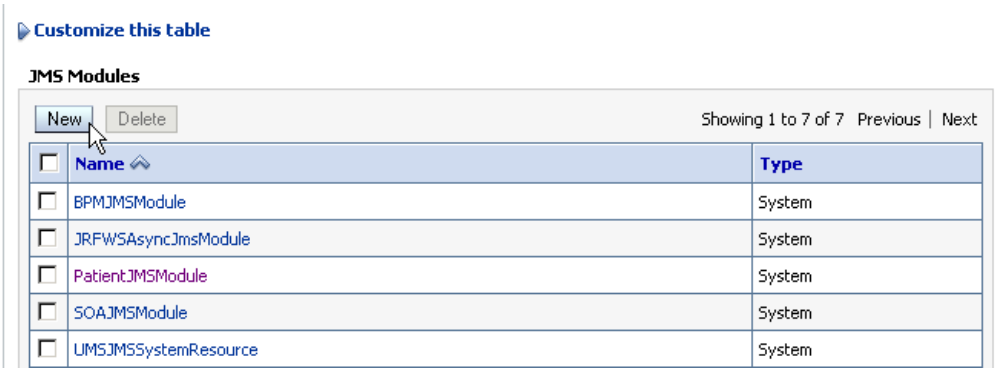
☐ Choose the target from the target list and click "Finish"



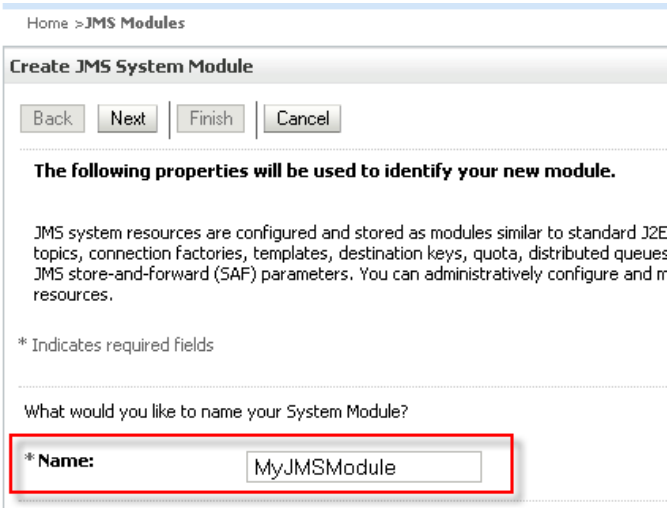
Create JMS Module

☐ Navigate the hierarchy "Services"→"Messaging" and click the "JMS Modules" node

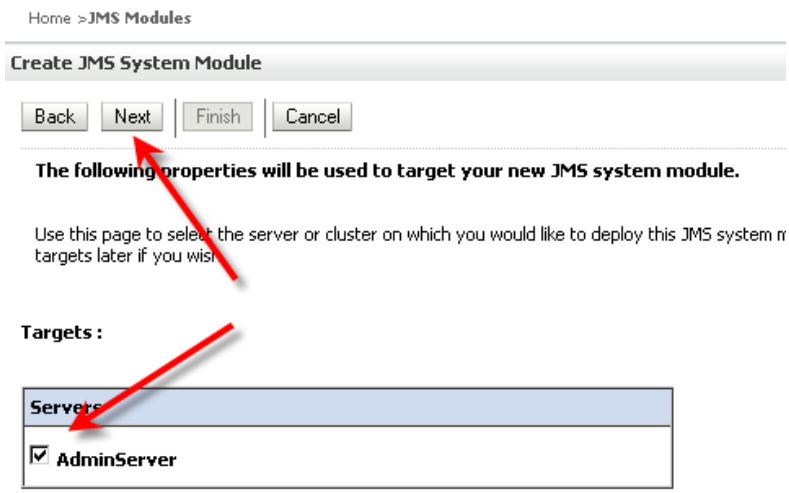
☐ Click "New" (strictly speaking creation of a JMS Module may not be necessary if you have one, for example the "SOAJMSModule" created by the installation of the SOA Suite. I don't know what your environment has so I have to assume that it does not have a JMS Module which can be used)



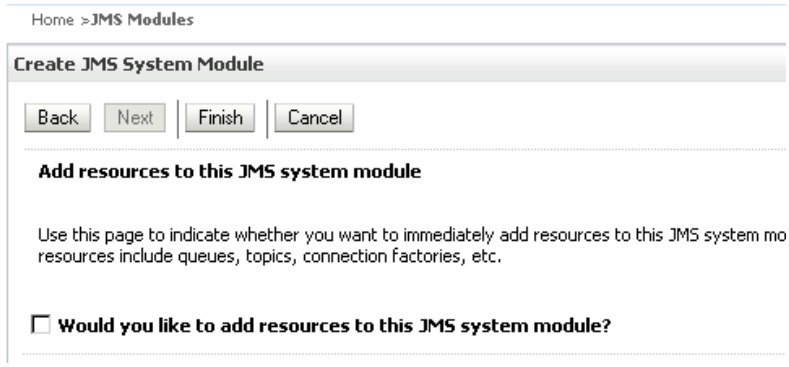
☐ Name the module "MyJMSModule" and click "Next"



☐ Check the "AdminServer" target and click "Next"

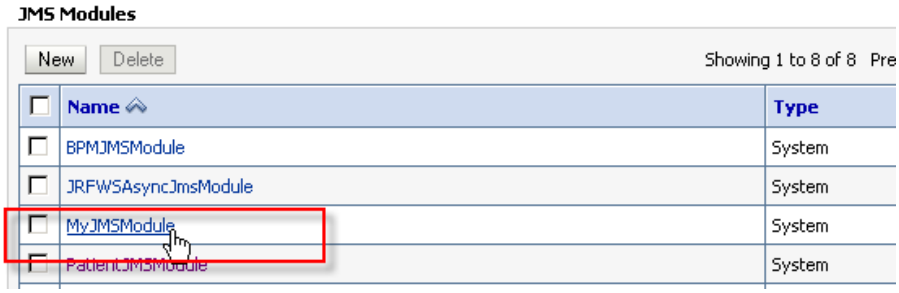


☐ Click "Finish" accepting the default configuration

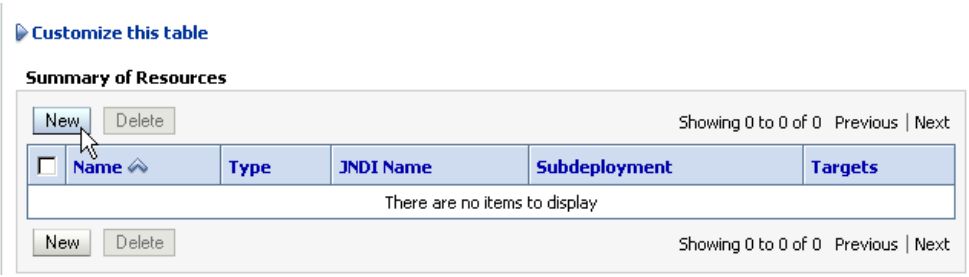


Create JMS Connection Factory

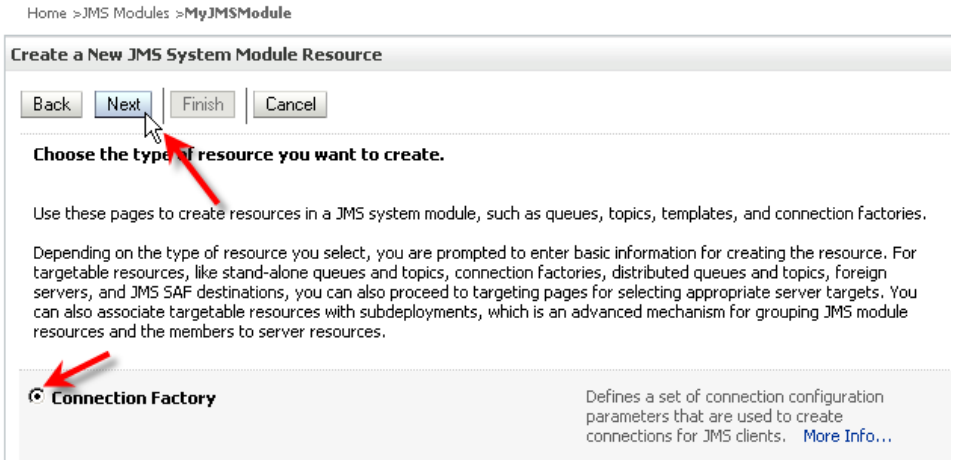
☐ Click the link "MyJMSModule"



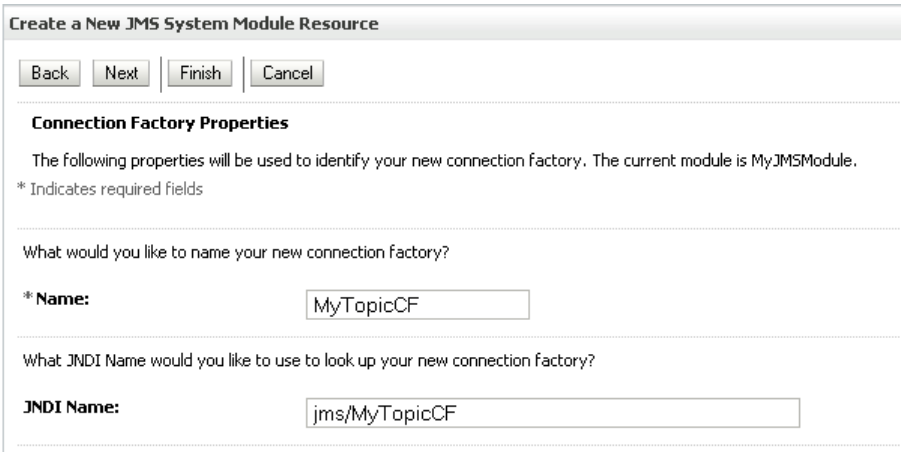
☐ Click "New"



☐ Select the "Connection Factory" radio button and click "Next"



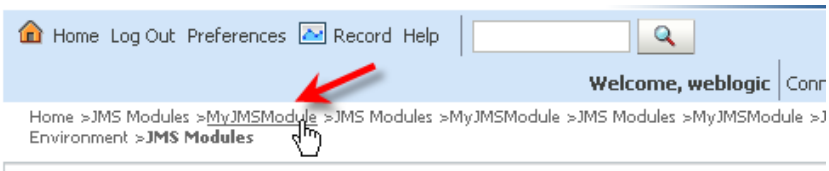
☐ Name the connection factory "MyTopicCF", set the JNDI Name to "jms/MyTopicCF" and click "Next"



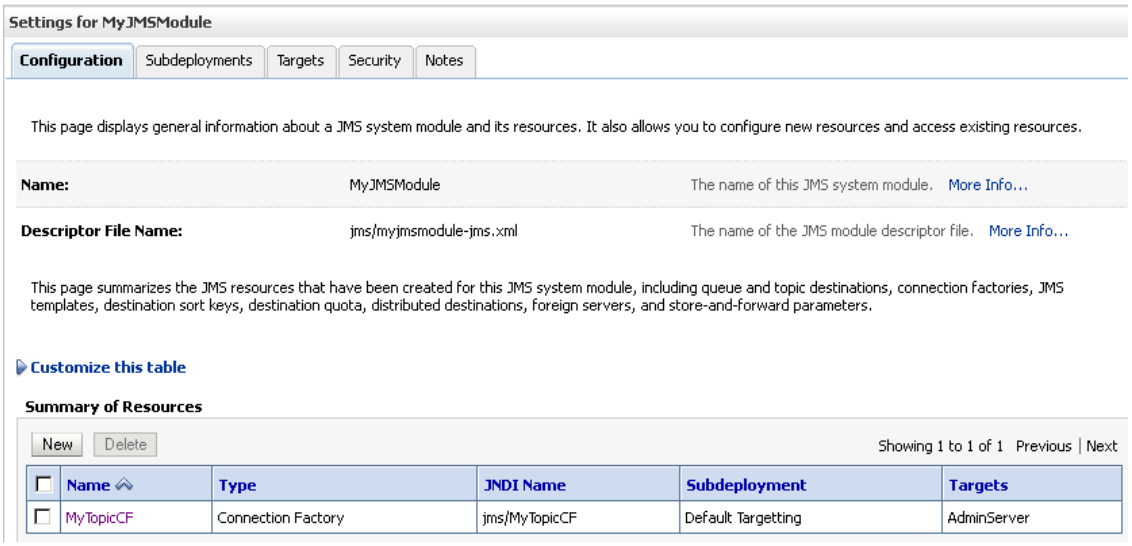
☐ Accept default target by clicking "Finish"

Create JMS Topic

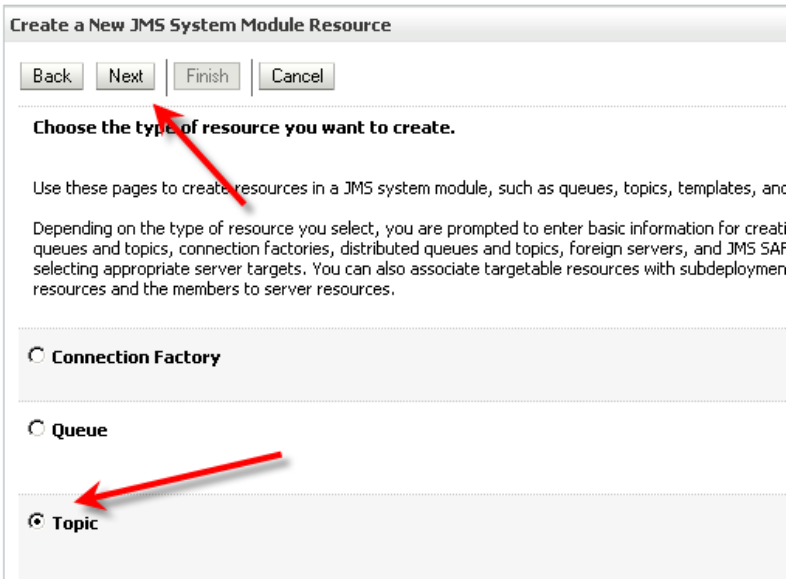
☐ Click the link "MyJMSModule"



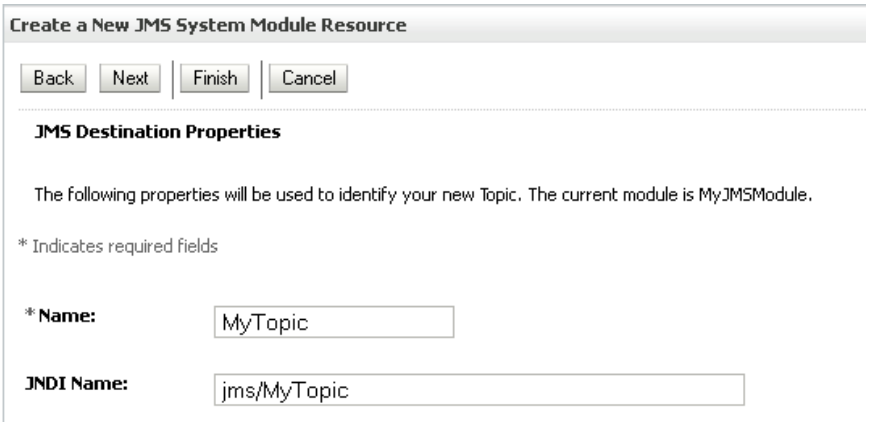
☐ Click "New"



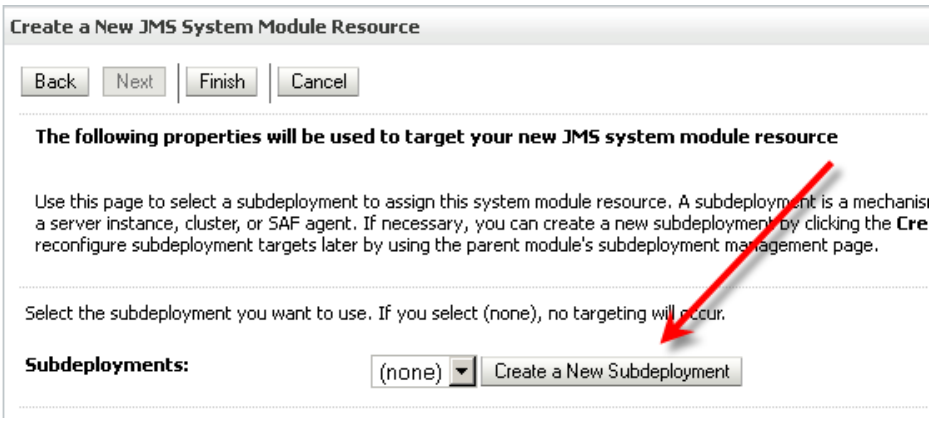
☐ Check the "Topic" radio button and click "Next"



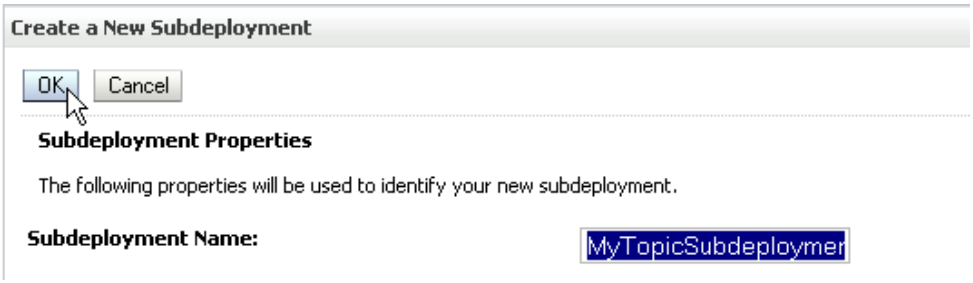
☐ Name the Topic "MyTopic", set the JNDI Name to "jms/MyTopic and click "Next"



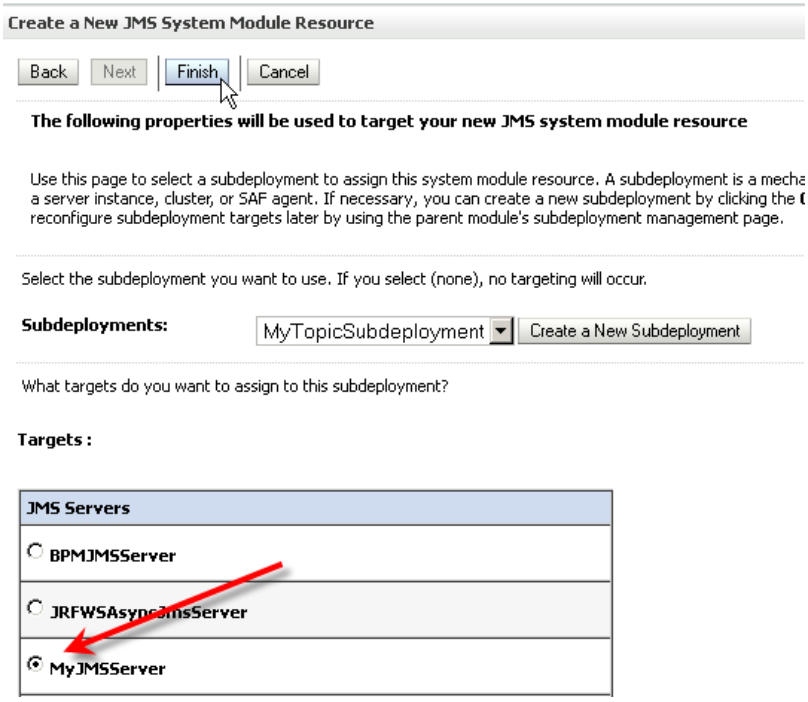
☐ Click the "Create a New Subdeployment" button



☐ Enter "MyTopicSubdeployment" as subdeployment name and click "OK"



☐ Check the radio button next to the "MyJMSServer" JMS Server name and click "Finish"



☐ Inspect the list of objects under our JMS Module then close the web browser – we are done here

On the Rhapsody Side

If the JMS Client Application which needs to communicate with the WebLogic JMS is the Rhapsody 4.01

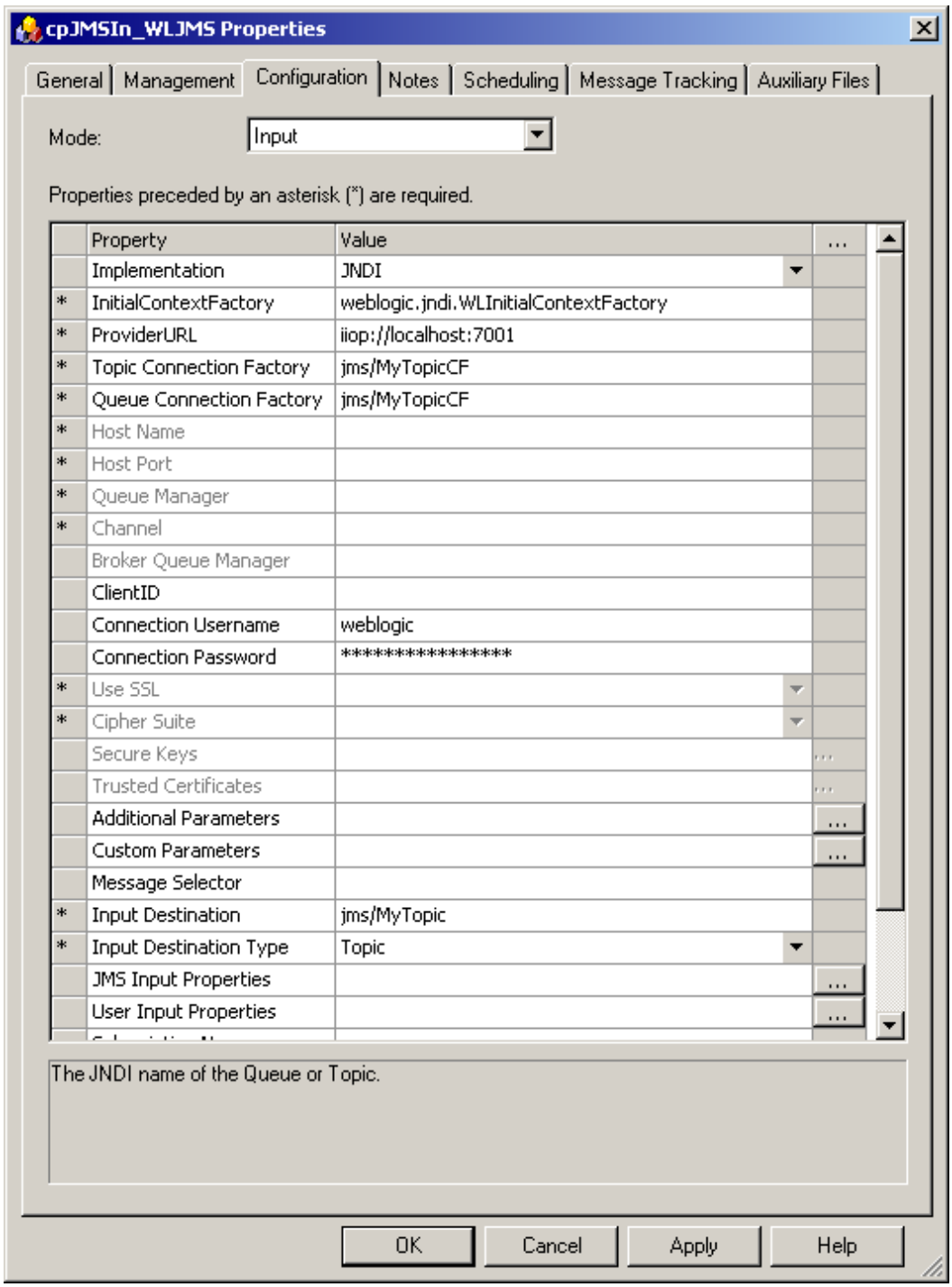
then its JMS Adapter must be correctly configured to support this communication, including making the wlfullclient.jar available to the JMS Adapter. The following discuss the process.

Unlike what is shown in a number of articles which deal with configuring a JMS Client for integration with the WebLogic JMS, the protocol to use is not the “t3” protocol, but the “iiop” protocol. This is what works for Rhapsody. In summary, here are the key configuration options for Rhapsody 4.01 and their settings. Some settings, like the WebLogic JMS Host and JMS Destination name and type, will vary.

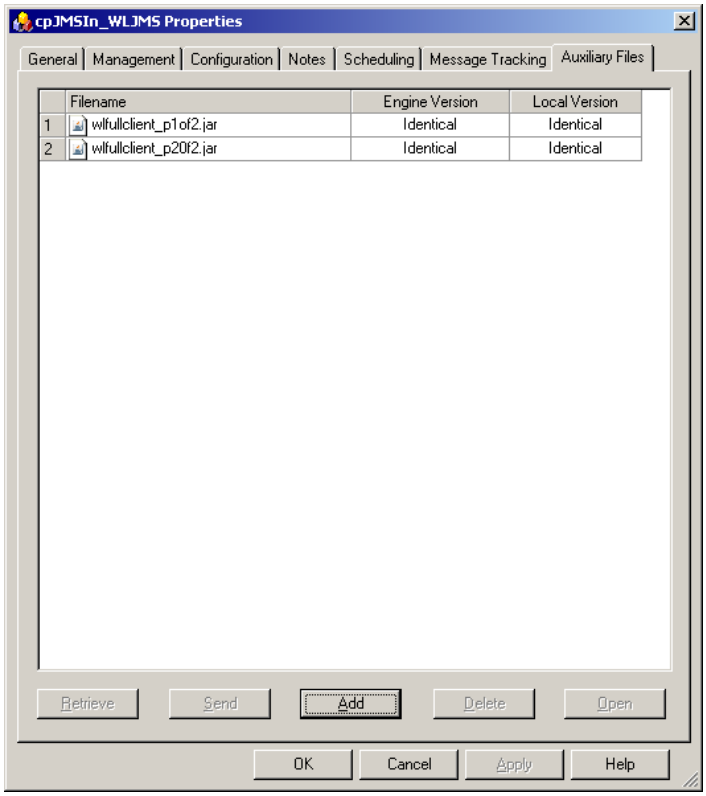
Property Name	Property Value
InitialContextFactory	weblogic.jndi.WLInitialContextFactory
ProviderURL	iiop://{WLJMSHost}:{WLJMSPort}
Topic Connection Factory	jms/{JMSTopicCopnnectionFactory}
Queue Connection Factory	jms/{JMSQueueConnectoinFactory}
Connection Username	{WLUsername}
Connection Password	{WLPassword}
Input Destination	jms/{MyTopicOrQueueName}
Input Destination Type	Topic (or Queue, depending on what you use)
Receiving Mode	Listening

In the table above values in {}, including {} themselves, must be replaced with the actual names from your configuration. For example, as shown in the picture below, my {WLJMSHost} would be replaced with localhost or some more appropriate host name or IP Address. My {WLJMSPort} would be replaced with 7001.

Rhapsody JMS Adapter configuration panel has an annoying idiosyncrasy where one must specify a Queue Connection Factory even if one uses a JMS topic. This does not have to be a Queue Connection Factory, as long as it is a connection factory. Without this the configuration cannot be saved.



The creation of a wlfullclient.jar results in a JAR which if around 58Mb in size. The Communication Point wizard seems unable to deal with a JAR this size, when adding it through the “Auxiliary Files” tab. To get around this issue, copy the wlfullclient.jar renaming the copy and the original so that you have two JARs – wlfullclient_p1of2.jar and wlfullclient_p2of2.jar. From the former delete the "weblogic" hierarchy, using 7-zip or some other archiver which can cope with JAR archives. From the later delete everything _except_ the “weblogic” hierarch. Add both archives to Auxiliary Files for the JMS Communication Point.



Now create a route with whatever additional communication points you might need and start the lot. Expect, if all is configured correctly, to receive messages from the topic hosted by the WebLogic JMS.

For the JMS objects, whose creation on the WebLogic Server side was discussed earlier, the settings would be:

Property Name	Property Value
InitialContextFactory	weblogic.jndi.WLInitialContextFactory
ProviderURL	iiop://localhost:7001
Topic Connection Factory	jms/MyTopicCF
Queue Connection Factory	jms/MyTopicCF
Connection Username	Weblogic
Connection Password	weclome1
Input Destination	jms/MyTopic
Input Destination Type	Topic
Receiving Mode	Listening

Summary

In this article I walked through the process of setting up JMS Topic and its dependencies on WebLogic 10.3 platform and configuring Rhapsody 4.01 JMS Communication Point to receive messages from the JMS Topic hosted by the WebLogic 10.3 Server. Perhaps this will save you the time I spent figuring out how to do this.