Using Directory API to Configure Communication Channels

Applies to:

SAP NetWeaver Process Integration XI 3.0, PI 7.0 and PI 7.1.

Summary

There are situations where the Integration Directory might not be the ideal tool to create configuration objects. When such situation occurs, we have the option to use the Directory API. Using the Directory API, we have the ability to edit configuration objects using a different user-interface. In the article, we will explore a fully functional tool, using Directory API and Excel spreadsheet, to configure communication channels.

Author: William Li

Company: SAPLabs LLC

Created on: 29 April 2010

Author Bio

William Li is a member of the SAP NetWeaver Integration Team in SAP NetWeaver RIG Americas focusing on Process Integration. He has been with SAP since 1998 - originally as a SAP America consultant then as a developer at SAPLabs. He joined SAP Intelligence Platform and NetWeaver RIG Americas in January of 2003.

Table of Contents

Disclaimer

Note: The free software programs provided here may be freely distributed, provided that the disclaimer below is always attached to it.

The programs are provided as is without any guarantees or warranty.

Although the author has attempted to find and correct any bugs in the free software programs, the author is not responsible for any damage or losses of any kind caused by the use or misuse of the programs.

The author is under no obligation to provide support, service, corrections, or upgrades to the free software programs.

There are occasions when the Integration Directory cannot meet our requirements when configuring objects for our integration scenarios. SAP provides the Directory API to meet those special situations. However, due to the nature of the product, there is little documentation available to help the developers. In this article, we will introduce a fully-functional program to create, change and delete communication channels. The user-interface to administer the configurations will be through the Excel spreadsheet.

The source code of the program can also be used as a template to administer other Integration Directory objects.

Scenario

The program can be used to meet the requirements of the following examples, which are frequently requested by PI developers, consultants and administrators.

- The PI developer needs to create 60 different File Communication Channels to integrate with MDM. The developer would like to do this quickly and, due to the repetitive tasks, possibly give the task to a non-PI-trained person.
- After transport of communication channels from DEV to QA, the PI administrator needs to re-enter several pieces of data which were blanked out after transport. These data typically consist of server names, usernames and passwords for various types of communication channels, e.g. JDBC, SOAP, JMS.

Introduction

In order to configure the scenarios, a simple Excel user-interface is used. The information can be entered offline using the spreadsheet. Then, a batch or script can be executed, referencing the spreadsheet file, to configure the communication channels.

The program has the following capabilities and limitations:

1. The PI 7.0 version of the Directory API is used. This API can be used on XI 3.0, PI 7.0 and PI 7.1x systems. However, new features introduced in PI 7.1 are not available.
2. Create, change and delete of communication channels can be done.
3. For change, only the communication channel properties, e.g. file directory, file name, host name/address, username, password, etc., can be changed. Changing other values, such as direction (sender or receiver), cannot be done. However, the program can be modified to do so.
4. Option to activate or not to activate the changelist is part of the program. When the changelist is not activated, the changelist can be viewed in the Integration Directory.
5. The program includes a subset of communication channels, specifically, File, JMS-MQ, JDBC, SOAP and RFC. Other communication channels can be included without programming (details below).
6. Excel 2003 version is used. The Excel API is a very light version. Any special formatting and formulas must not be used. If Excel 2007 is used to create the spreadsheet, then you must save the spreadsheet as a 2003 version.
7. The locations or positions of the data used in the spreadsheet must not be changed. Their column and row positions are hard-coded in the program.
8. Before creating any communication channels, the Service Component and Party (if used) must already exist.

The source code of the program is available for download and can be used as template to configure other Integration Directory objects and to extend the capability of the current program. The download location is:

http://blog.tadi.com/download/DirAPI.zip

How to Use the Program: Step-By-Step Directions

The program is spreadsheet-driven, there are 3 types of worksheets:

1. Runtime information, e.g. which PI server to execute, username/password to use, etc.
2. Adapter information, .e.g. adapter name, transport and message protocols, SWCV id, etc. This worksheet can allow us to specify PI systems and adapters to use during runtime. It will also allow us to introduce new adapters, which has not been included with this program. It is important to note that SWCV ids are different between PI releases.
3. Communication channel configuration, e.g. party, component, channel name, direction, etc. There is one worksheet for each adapter type. The reason for this is that adapter specific properties differ considerably between each other. To include all the properties for all the adapters in one worksheet will make the spreadsheet too large and can render the configuration too confusing to use.

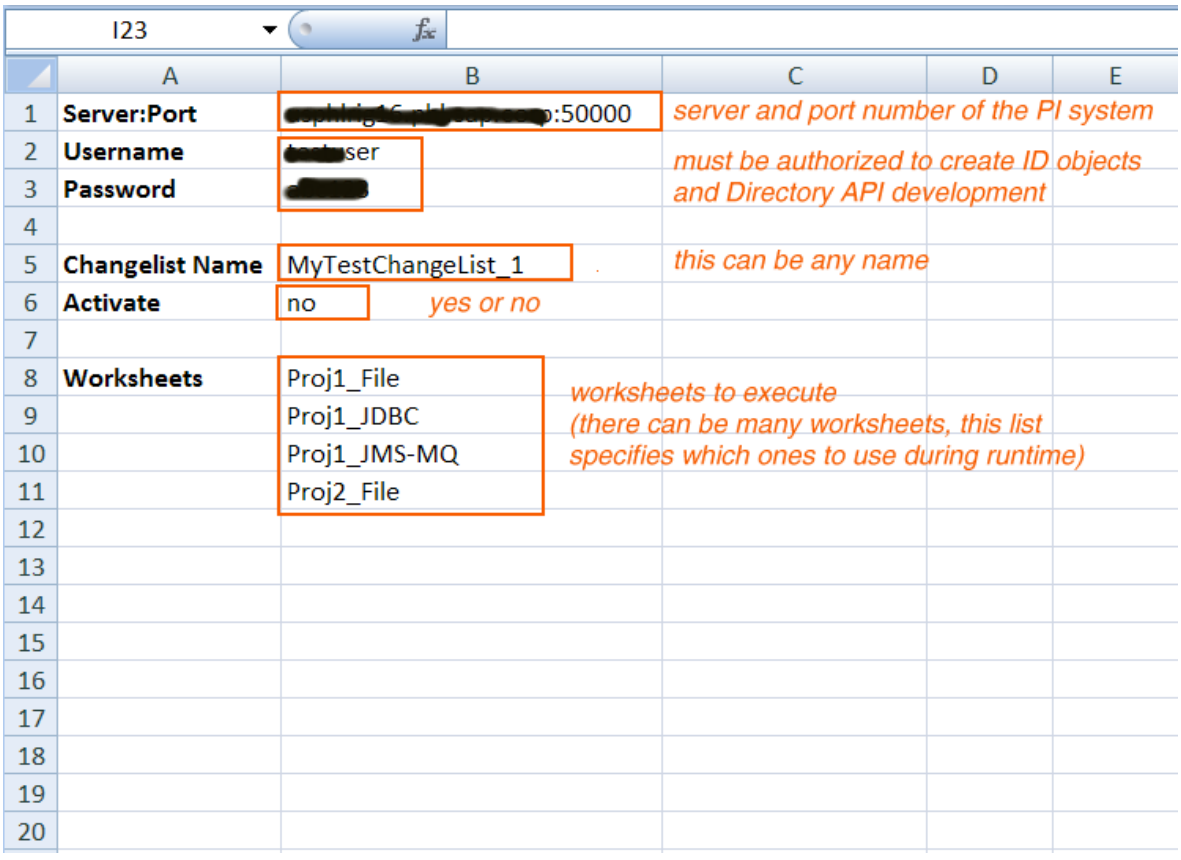
   New adapter types can be introduced using this worksheet, by including a list of properties for that adapter.

Worksheet: Runtime Properties

Following are the information needed during runtime:

1. Server:Port: PI system server name/address and port number
2. Username and Password: authorized user to execute the Directory API
3. For steps on how to configure the user for Directory API, please reference the blog:

   http://www.sdn.sap.com/irj/scn/weblogs?blog=/pub/wlg/11655
4. Changelist Name: this can be any name. It is name that will show up in the “Change List” tab of the Integration Directory.
5. Activate: whether to activate the changes or not. Valid values are “yes” or “no”.
6. Worksheets: a list of worksheet that will be used for configuration during this runtime execution.

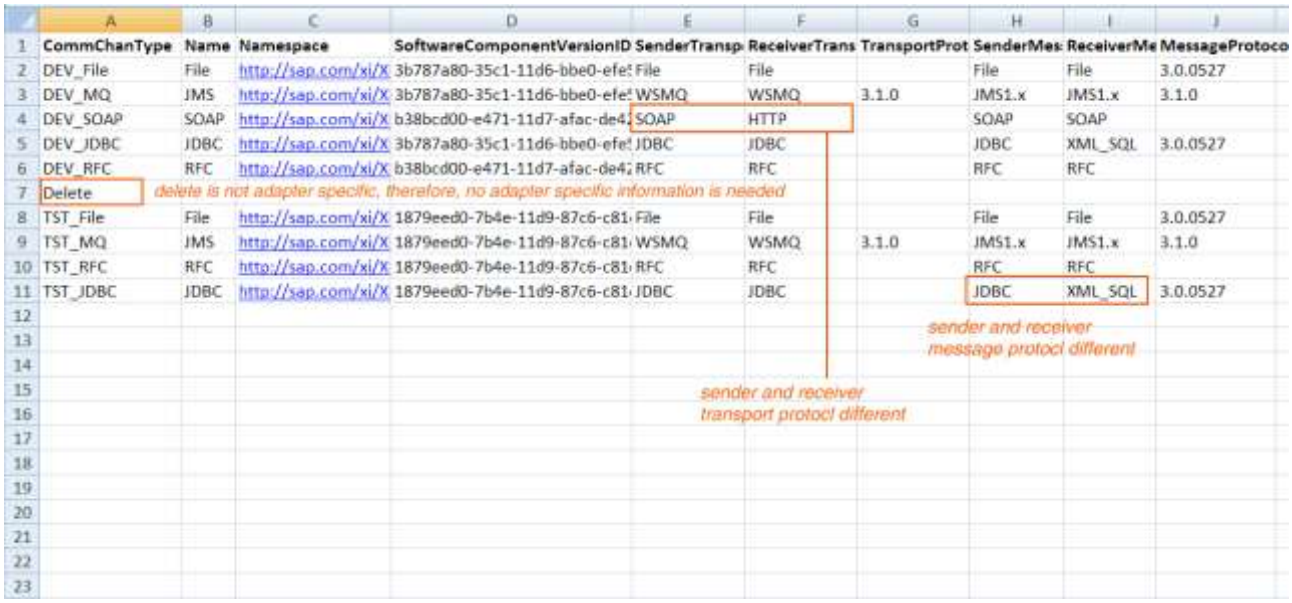


Worksheet: Adapter Properties

Following are the properties for each adapter:

1. CommChanType: this can be any value. Typically, I like to use the system and the adapter type as the value. This allows me to enter the adapter information for multiple systems. This value is used in the communication channel configuration worksheets.
2. Name: name of the adapter
3. Namespace: namespace used by the adapter
4. SoftwareComponentVersionID: this is GUID of the SWCV where the adapter metadata is located. This value is unique based on the release of the PI installation.
5. SenderTransportProtocol: this is the sender transport protocol of the adapter. Normally, the sender and receiver have the same protocol, but not always. For SOAP adapter, they are different.
6. ReceiverTransportProtocol: this is the receiver transport protocol of the adapter. Normally, the sender and receiver have the same protocol, but not always. For SOAP adapter, they are different.
7. TransportProtocolVersion: the version of the transport protocol, if it exists.
8. SenderMessageProtocol: this is the sender message protocol of the adapter. Normally, the sender and receiver have the same protocol, but not always. For JDBC adapter, they are different.
9. ReceiverMessageProtocol: this is the receiver message protocol of the adapter. Normally, the sender and receiver have the same protocol, but not always. For JDBC adapter, they are different.
10. MessageProtocolVersion: the version of the message protocol, if it exists

Below is an example of the worksheet:



	A	B	C	D	E	F	G	H	I	J
1	CommChanType	Name	Namespace	SoftwareComponentVersionID	SenderTransp	ReceiverTrans	TransportProt	SenderMes	ReceiverMe	MessageProtoco
2	DEV_File	File	http://sap.com/xi/X	3b787a80-35c1-11d6-bbe0-efe	File	File		File	File	3.0.0527
3	DEV_MQ	JMS	http://sap.com/xi/X	3b787a80-35c1-11d6-bbe0-efe	WSMQ	WSMQ	3.1.0	JMS1.x	JMS1.x	3.1.0
4	DEV_SOAP	SOAP	http://sap.com/xi/X	b38bcd00-e471-11d7-afac-de4	SOAP	HTTP		SOAP	SOAP	
5	DEV_JDBC	JDBC	http://sap.com/xi/X	3b787a80-35c1-11d6-bbe0-efe	JDBC	JDBC		JDBC	XML_SQL	3.0.0527
6	DEV_RFC	RFC	http://sap.com/xi/X	b38bcd00-e471-11d7-afac-de4	RFC	RFC		RFC	RFC	
7	Delete									
8	TST_File	File	http://sap.com/xi/X	1879eed0-7b4e-11d9-87c6-c81	File	File		File	File	3.0.0527
9	TST_MQ	JMS	http://sap.com/xi/X	1879eed0-7b4e-11d9-87c6-c81	WSMQ	WSMQ	3.1.0	JMS1.x	JMS1.x	3.1.0
10	TST_RFC	RFC	http://sap.com/xi/X	1879eed0-7b4e-11d9-87c6-c81	RFC	RFC		RFC	RFC	
11	TST_JDBC	JDBC	http://sap.com/xi/X	1879eed0-7b4e-11d9-87c6-c81	JDBC	JDBC		JDBC	XML_SQL	3.0.0527

In the latter part of this article, we will discuss how these values can be obtained easily.

Worksheet: Communication Channel Configuration

The worksheet name can be anything; this name is referenced in the **Runtime Properties** worksheet, as to which worksheet will be used during runtime.

Each worksheet references only 1 adapter type. The adapter type must be in the list of **CommChanType** of the **Adapter Properties** worksheet above.

There can be multiple columns in the worksheet. Each column specifies a communication channel. The columns must be contiguous (or adjacent to each other). There must not be any empty or blank columns in between.

The worksheet contains the information below:

1. Adapter Type: this value must be in the list of **CommChanType** of the **Adapter Properties** worksheet above.
2. Action: specifies what we want to do with the communication channel: create, change or delete.
3. PartyID
4. ComponentID
5. ChannelID
6. Direction: whether the communication channel is a sender or receiver
7. Adapter specific properties: this is a list consists only properties for a specific adapter, e.g. File adapter will have a different set of properties from JDBC adapter. (In the latter part of this article, we will discuss how these values can be obtained easily.)

Below is an example of the worksheet:

	A	B	C	D	E	F
1	**Valid Values**	**Adapter Type**	DEV_File	*this value must exist in the Adapter Properties worksheet*		
2	create	**Action**	create	delete	create	change
3						
4		**PartyID**				
5		**ComponentID**	BLService	BLService	BLService	BLService
6		**ChannelID**	File_test_sender1	File_test_sender	File_test_receiver1	File_dirapi_test_receiver
7						
8	Sender/Receiver	**Direction**	Sender		Receiver	
9						
10		**file.sourceDir**	/tmp			
11		**file.sourceFileName**	fileSource2.xml			
12	EO/EOIO	**file.qualityOfService**	EO			
13		**file.queueName**	*one communication channel per column*			
14		**file.pollInterval**	5			
15	readonly	**file.processingMode**	readonly			
16		**file.archiveWithTimestamp**				
17		**file.archiveDir**				
18	byName	**file.processingOrder**	byName			
19	bin	**file.type**	bin		bin	
20		**file.encoding**				
21	*adapter*	**createTargetDirectory**			1	
22	*specific*	**file.targetDir**			/tmp	/data
23	*properties*	**file.targetFileName**			fileTarget1.xml	
24		**file.writeMode**			addCounter	
25		**file.overwrite**			1	
26		**file.writeDirectly**			YES	
27		**file.counterSeparator**			_	
28		**file.counterFormat**			00	0

The sample spreadsheet for this program contains the following adapter properties:

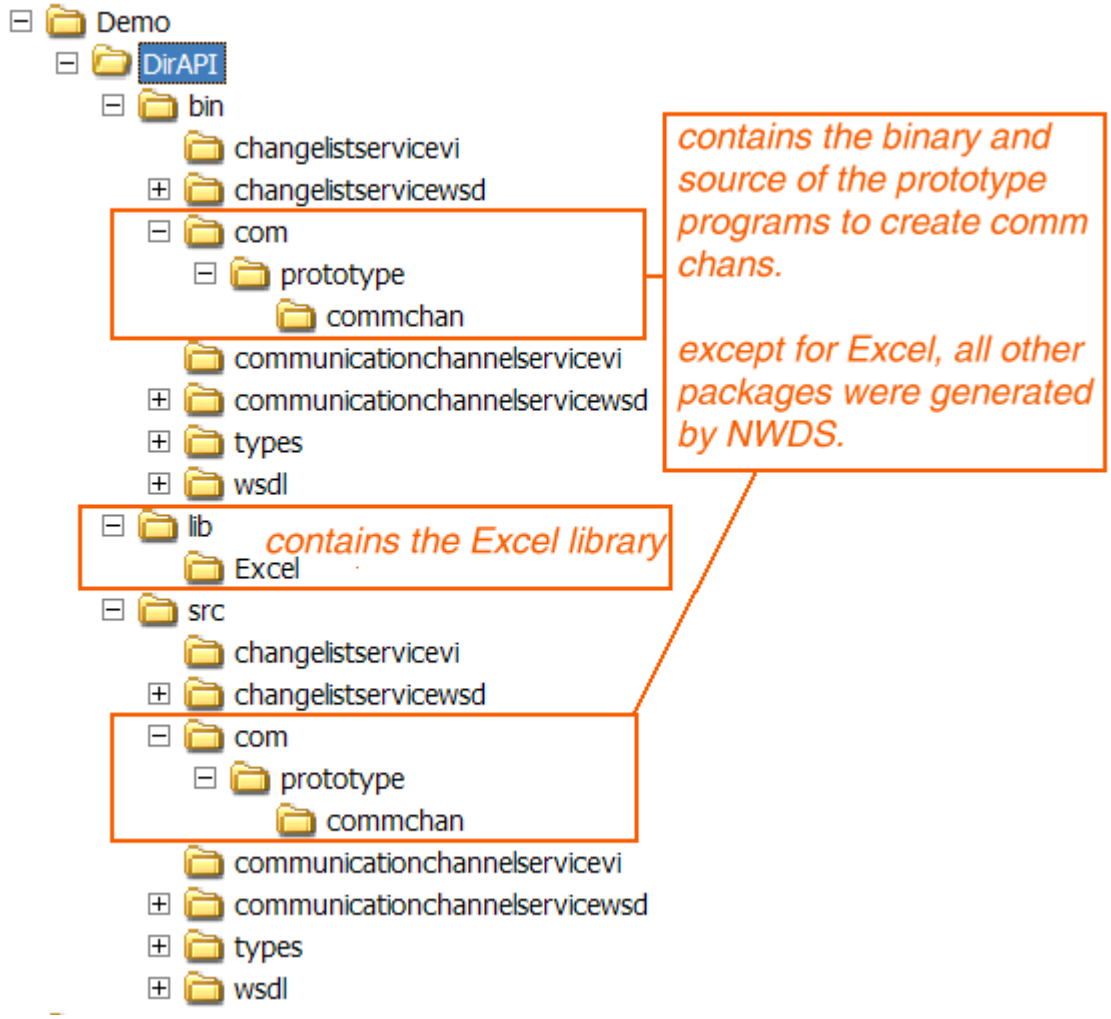
- File
- JDBC
- JMS-MQ
- SOAP
- RFC

I also included a sheet that does only delete, disregard to the adapter type.

Executing the Program

The java version I used for runtime is “1.6.0_18”, “1.5” will also work.

After unzip the downloaded zip file, the following contains the structure:



bin directory content:

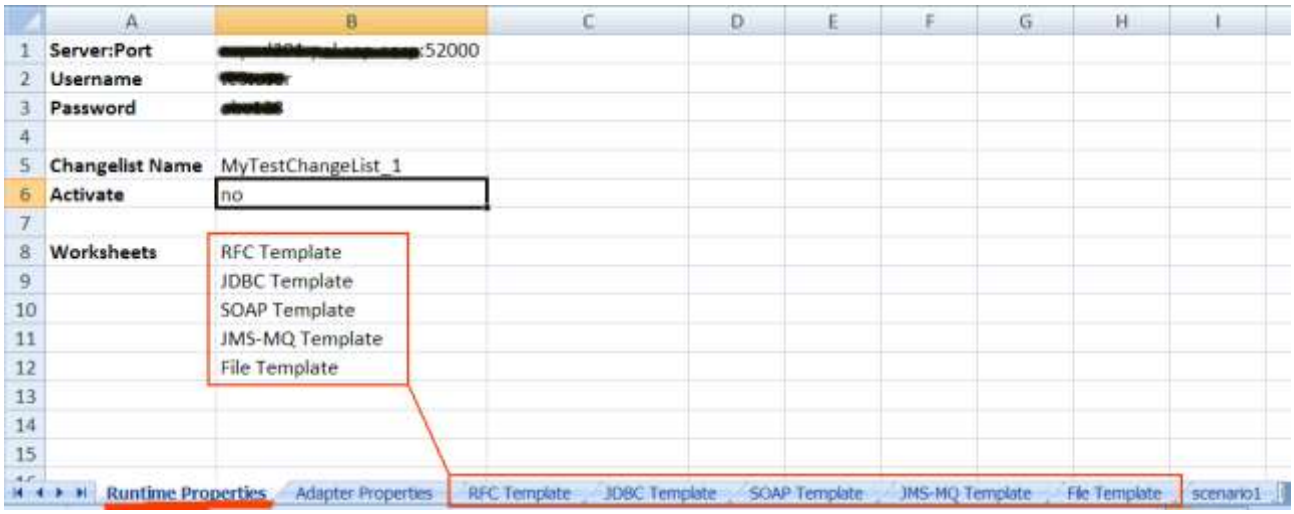
```
 Directory of C:\Demo\DirAPI\bin

04/06/2010  11:45 AM    <DIR>          .
04/06/2010  11:45 AM    <DIR>          ..
04/06/2010  10:11 AM    <DIR>          changelistservicevi
04/06/2010  10:11 AM    <DIR>          changelistservicewsd
04/06/2010  10:11 AM    <DIR>          com
04/06/2010  10:11 AM    <DIR>          communicationchannelservicevi
04/06/2010  10:11 AM    <DIR>          communicationchannelservicewsd
04/06/2010  10:11 AM    <DIR>          types
04/06/2010  10:11 AM    <DIR>          wsdl
04/06/2010  11:45 AM            52,224 CommChans.xls  sample spreadsheet
04/06/2010  11:14 AM               102 run.bat  script file to execute the spreadsheet
               2 File(s)         52,326 bytes
               9 Dir(s)  29,307,770,368 bytes free
```

The sample spreadsheet contains templates for File, JDBC, SOAP, RFC and JMS-MQ. Each of these templates is a worksheet.

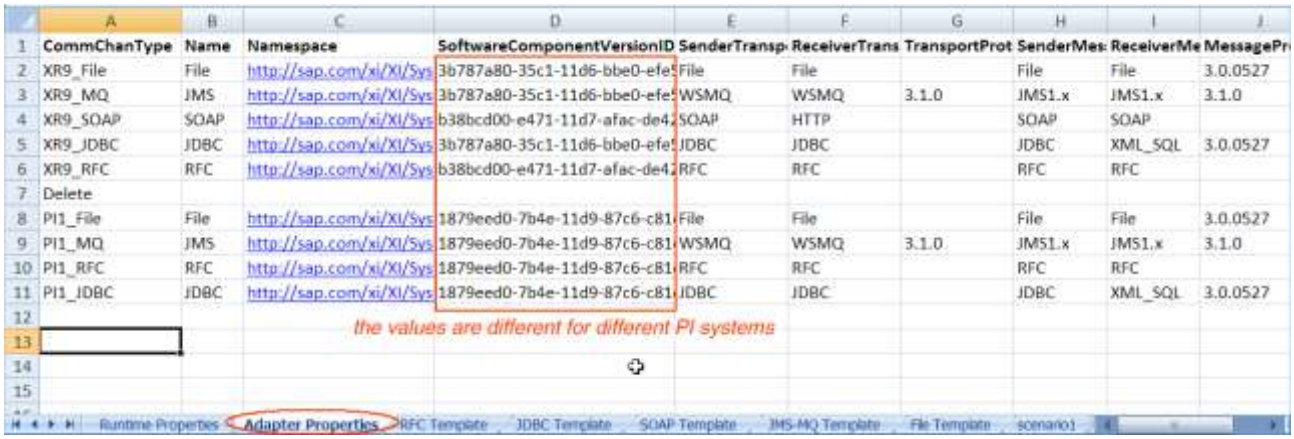
For demo purpose, I will execute all the templates on a PI 7.1 system.

Activation will be set to “no”, so we can look at the changelist to verify the configurations.

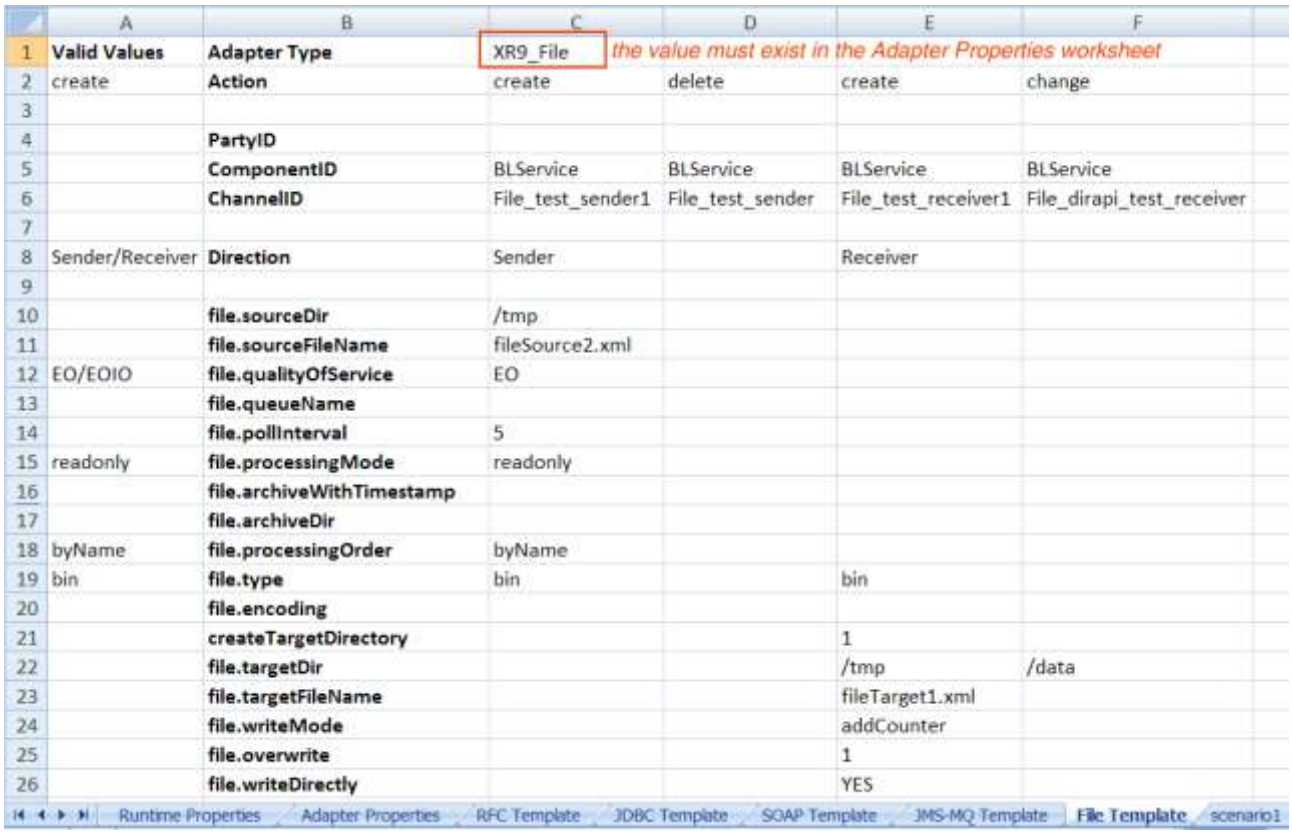
The Runtime Properties worksheet contains the following:



The Adapter Properties worksheet contains the following:



Example of a communication channel configuration worksheet:

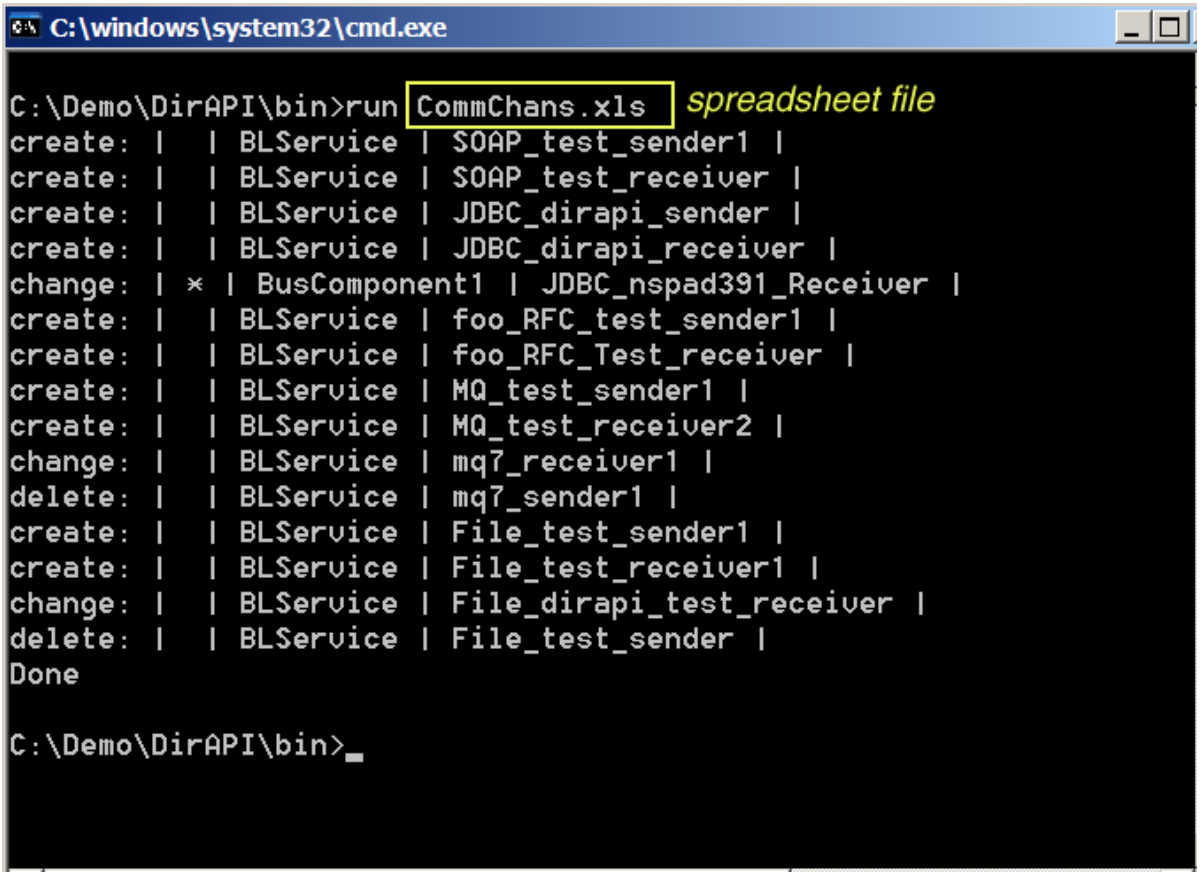


	A	B	C	D	E	F
1	**Valid Values**	**Adapter Type**	XR9_File	*the value must exist in the Adapter Properties worksheet*		
2	create	**Action**	create	delete	create	change
3						
4		**PartyID**				
5		**ComponentID**	BLService	BLService	BLService	BLService
6		**ChannelID**	File_test_sender1	File_test_sender	File_test_receiver1	File_dirapi_test_receiver
7						
8	Sender/Receiver	**Direction**	Sender		Receiver	
9						
10		**file.sourceDir**	/tmp			
11		**file.sourceFileName**	fileSource2.xml			
12	EO/EOIO	**file.qualityOfService**	EO			
13		**file.queueName**				
14		**file.pollInterval**	5			
15	readonly	**file.processingMode**	readonly			
16		**file.archiveWithTimestamp**				
17		**file.archiveDir**				
18	byName	**file.processingOrder**	byName			
19	bin	**file.type**	bin		bin	
20		**file.encoding**				
21		**createTargetDirectory**			1	
22		**file.targetDir**			/tmp	/data
23		**file.targetFileName**			fileTarget1.xml	
24		**file.writeMode**			addCounter	
25		**file.overwrite**			1	
26		**file.writeDirectly**			YES	

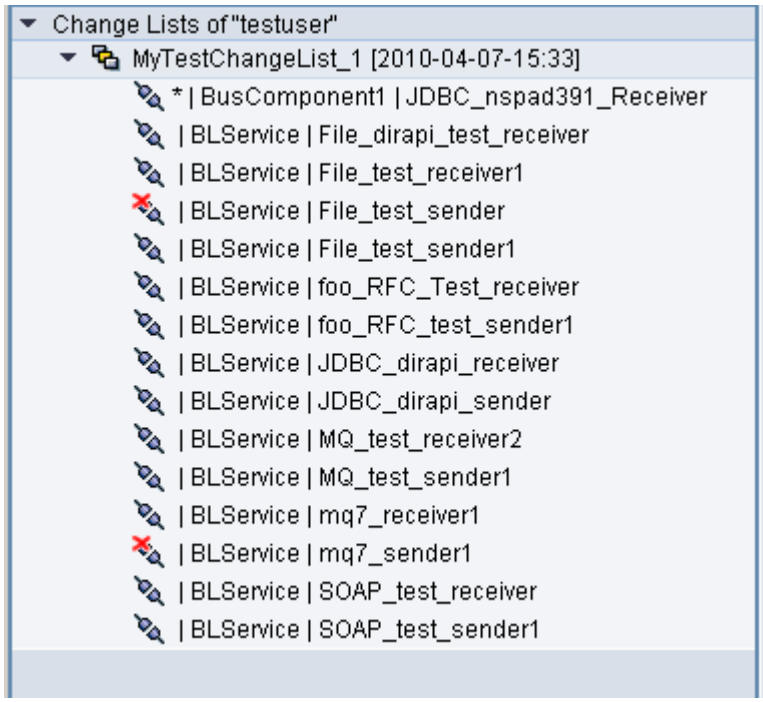
Runtime Properties | Adapter Properties | RFC Template | JDBC Template | SOAP Template | JMS-MQ Template | File Template | scenario1

Make sure the Adapter Type, using the SWCV id, matches the Server:Port of the Runtime Properties.

Execute the program:



In the Integration Directory, we see the following: (after clicking “Refresh” in the “Change Lists” tab)

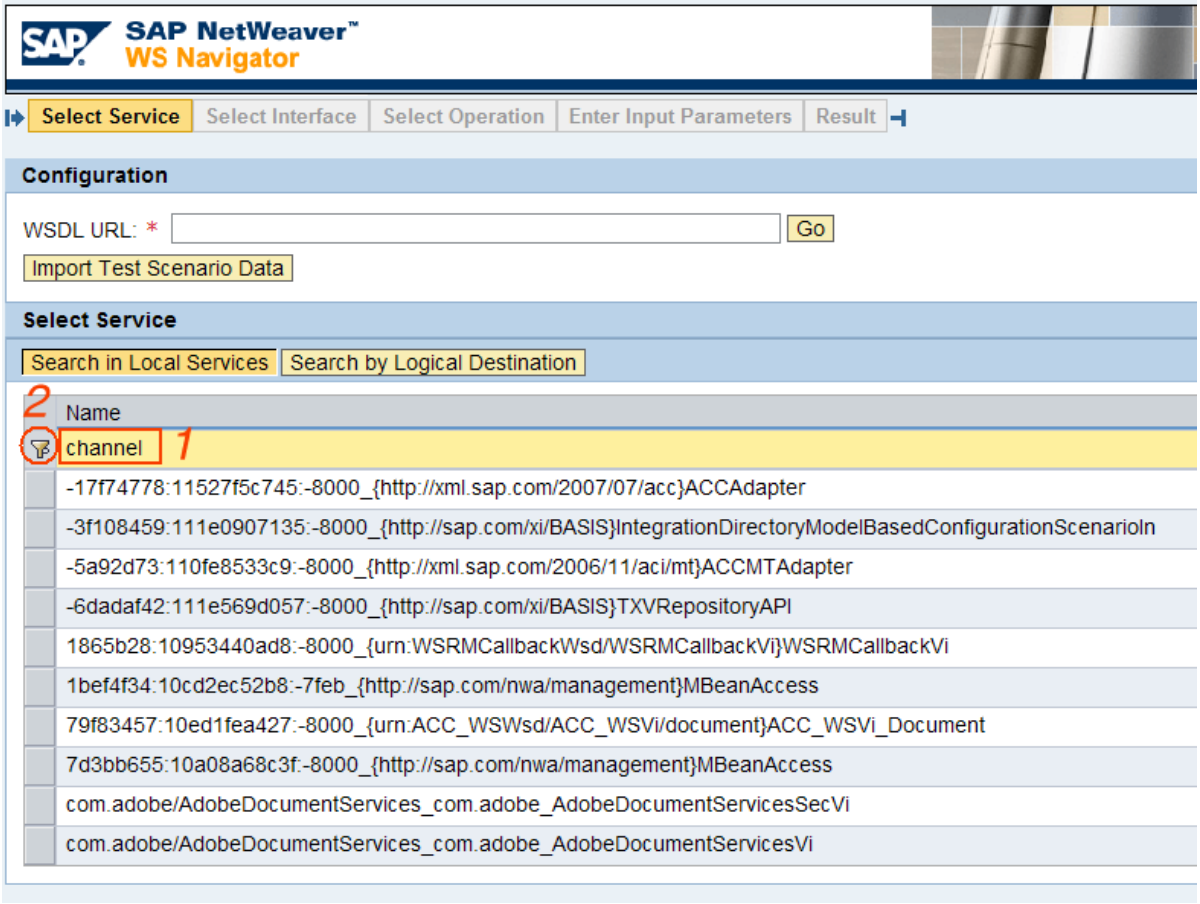


How to Obtain Adapter Information to Populate the Worksheets

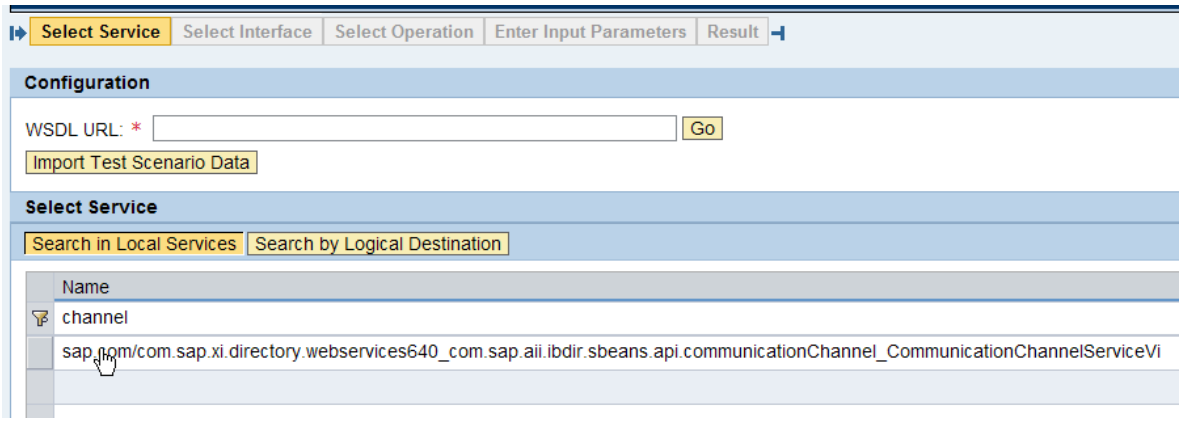
In the worksheets, we need to enter SWCV id, Transport and Message Protocols, and also the adapter specific properties. To obtain this information, we can use the WSNavigator.

Using PI 7.1 system, we can do the following: (PI 7.0 will have a little different UI for WSNavigator)

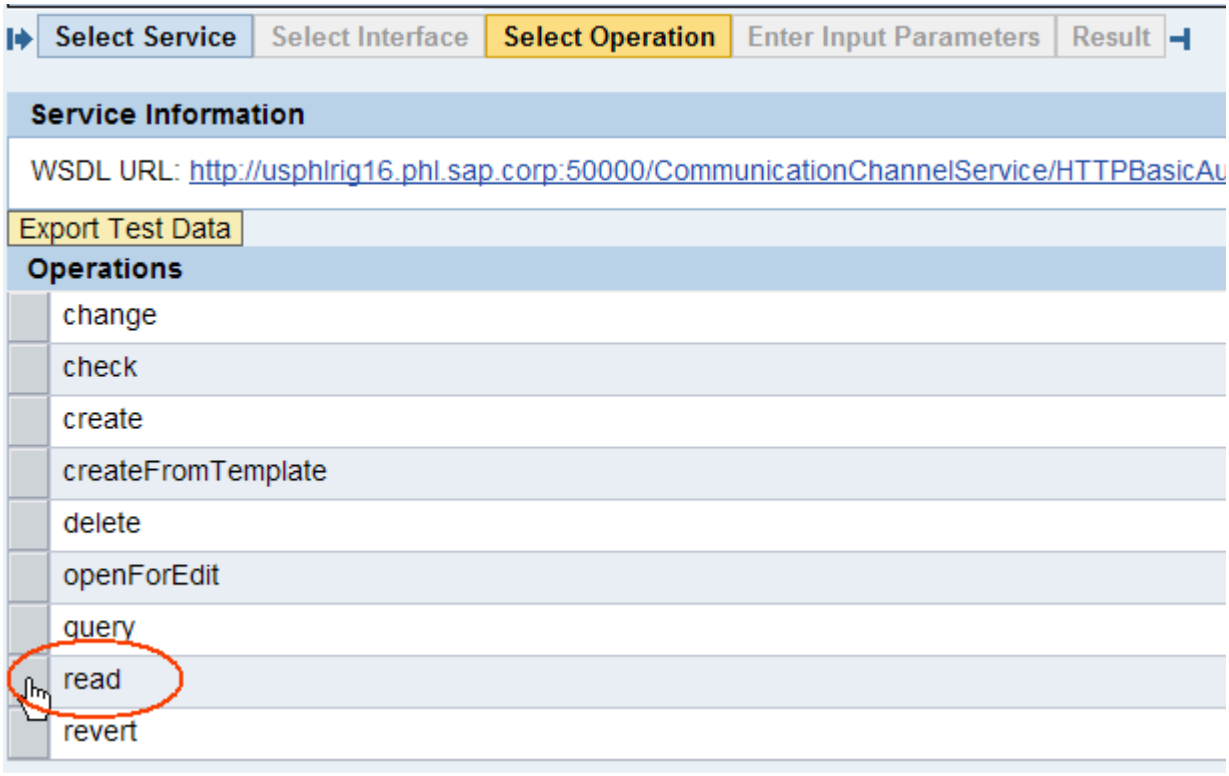
1. Enter URL: http://server:port/WSNavigator
2. After logged in, you should see the following. Enter "channel" to filter the listing:



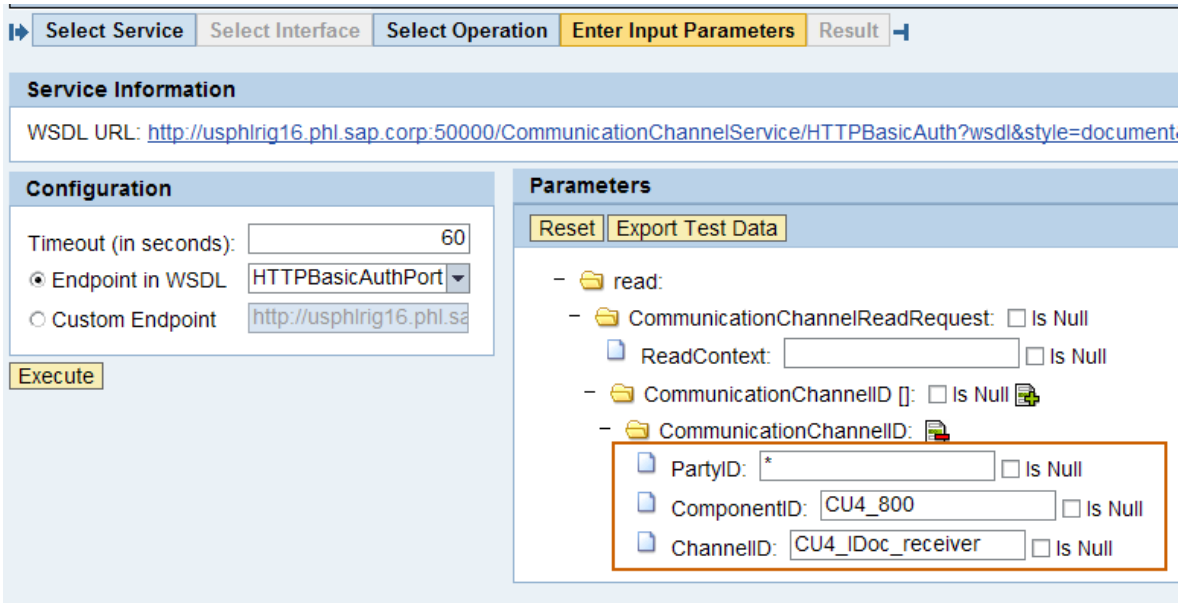
3. Select the "Communication Channel" service:



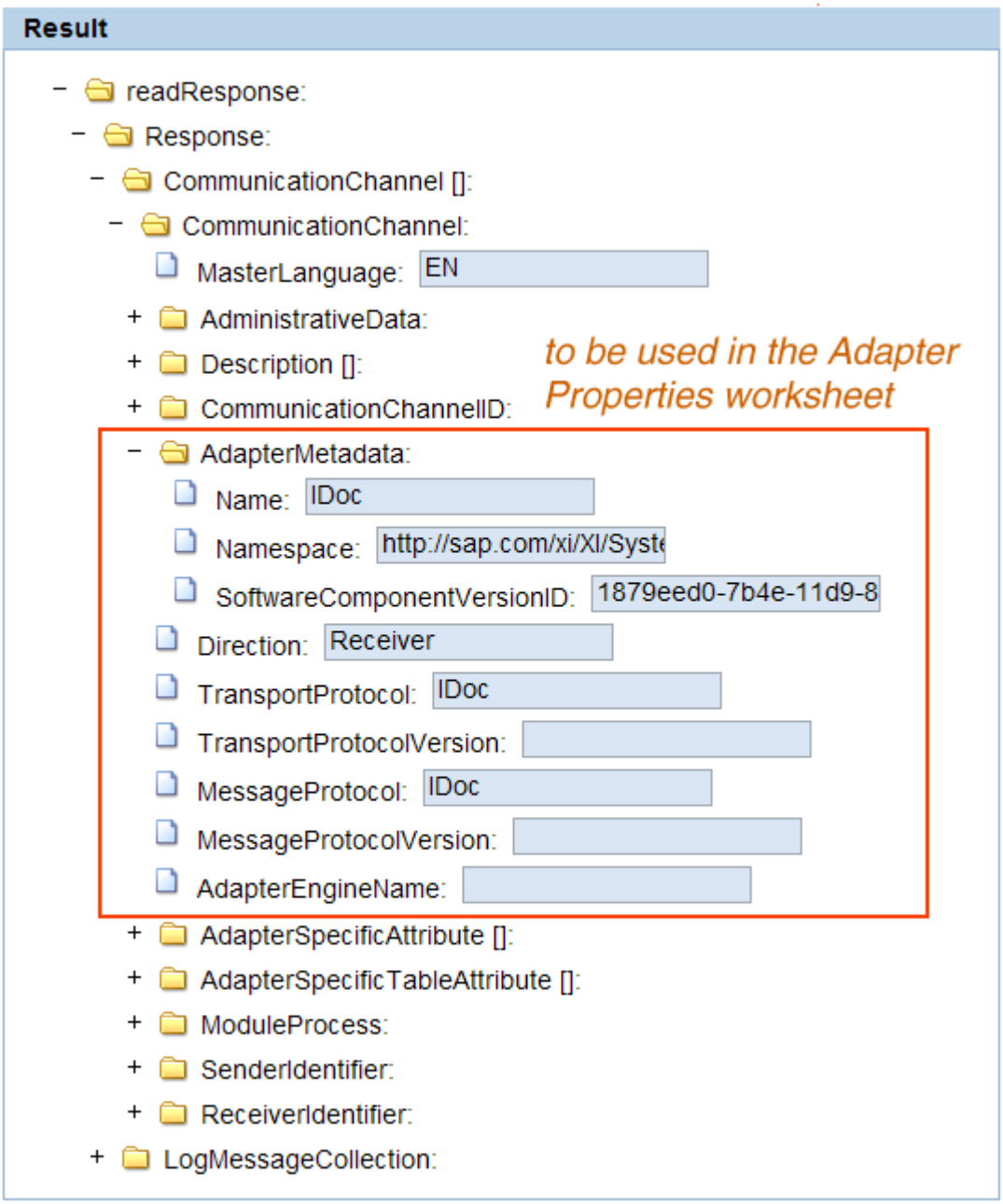
4. Select the “Read” operation:

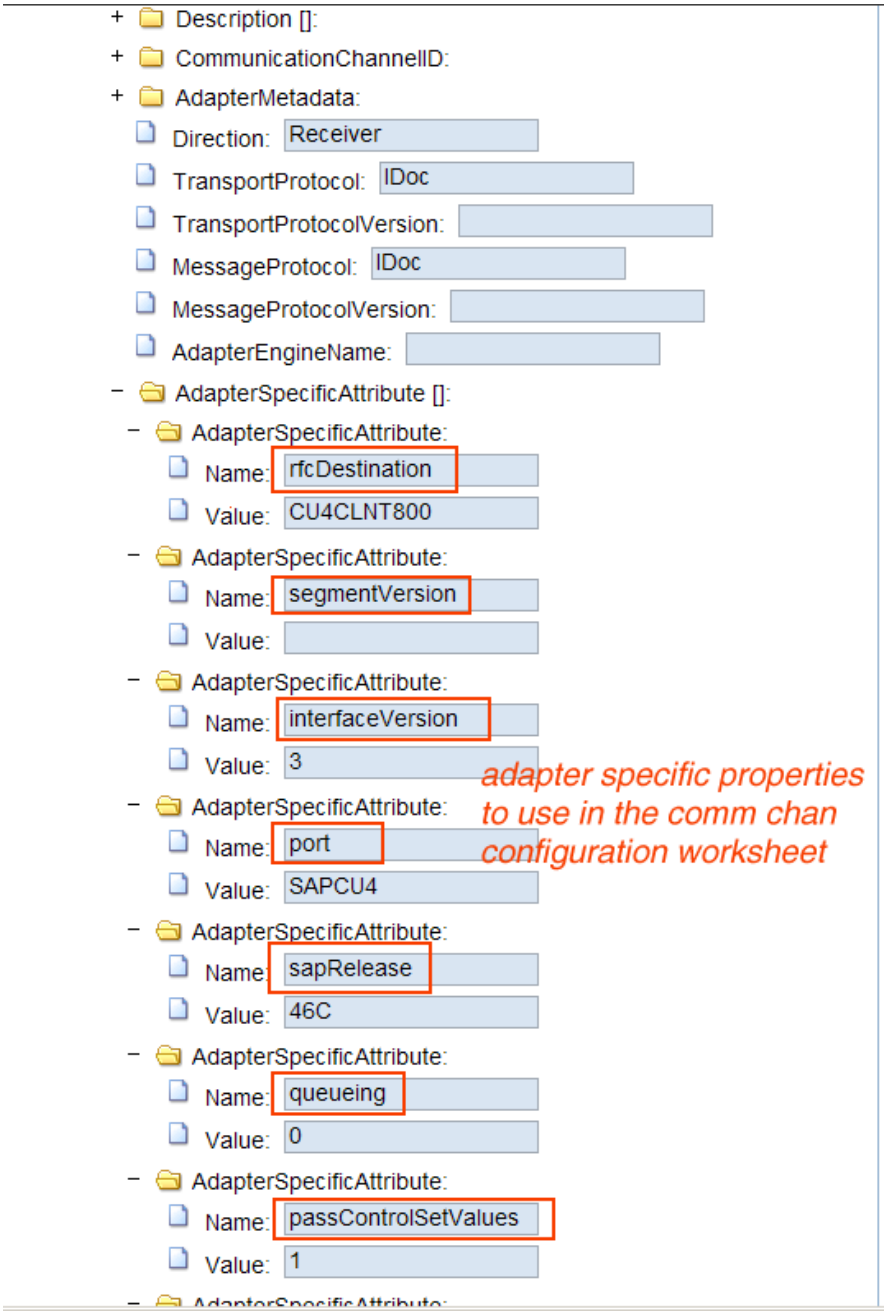


5. Retrieve an existing communication channel type of interest, which we can use to find out all the information. For here, we will find the information for the IDoc adapter:



6. We can find the following information:





As an alternative method, we can copy the XML content and use a text editor to extract the property names.

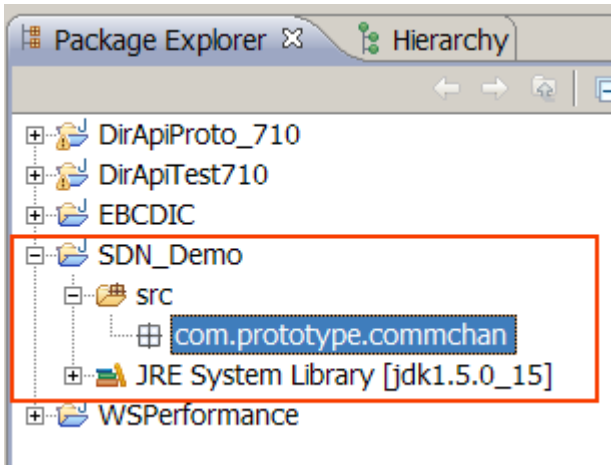
When going through this process, make sure we do this using communication channels for both sender and receiver, in order to get all the property names.

How to Use the Source Code in NWDS 7.1

The NWDS version used for development was NWDS 7.1 SP8.

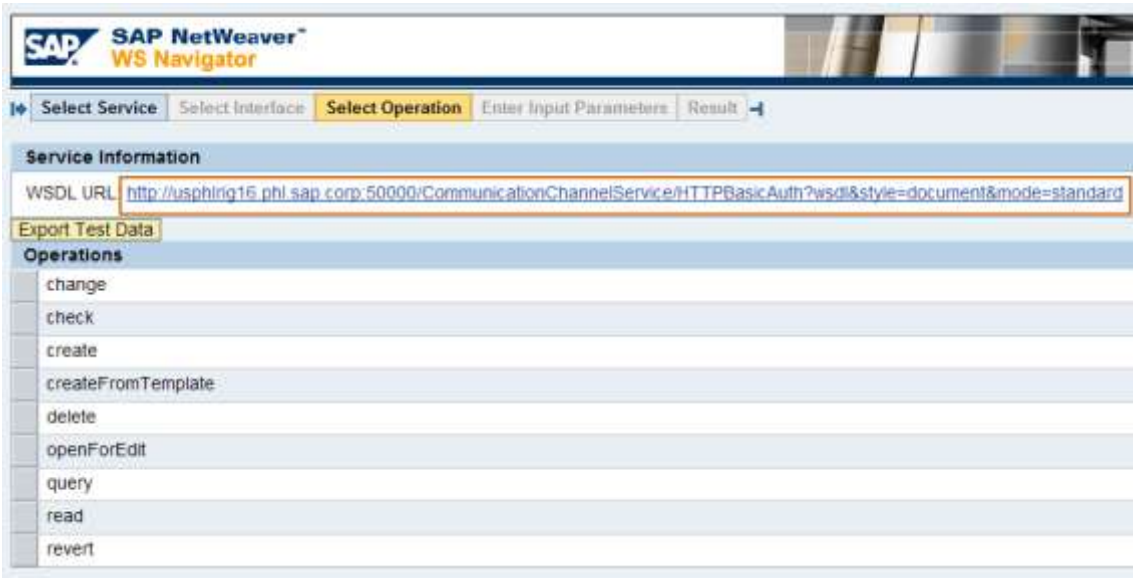
Create a Java Project and Package

Use com.prototype.commchan as the package name. It is the package name used in the program. You can change the package name after the source has been copied into NWDS.

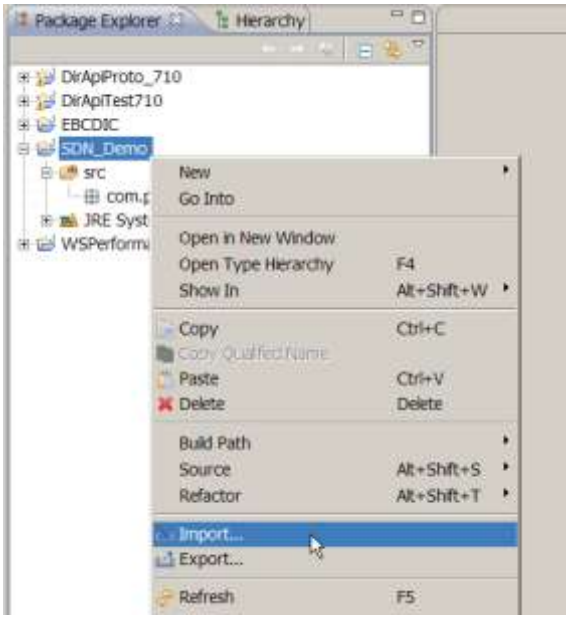


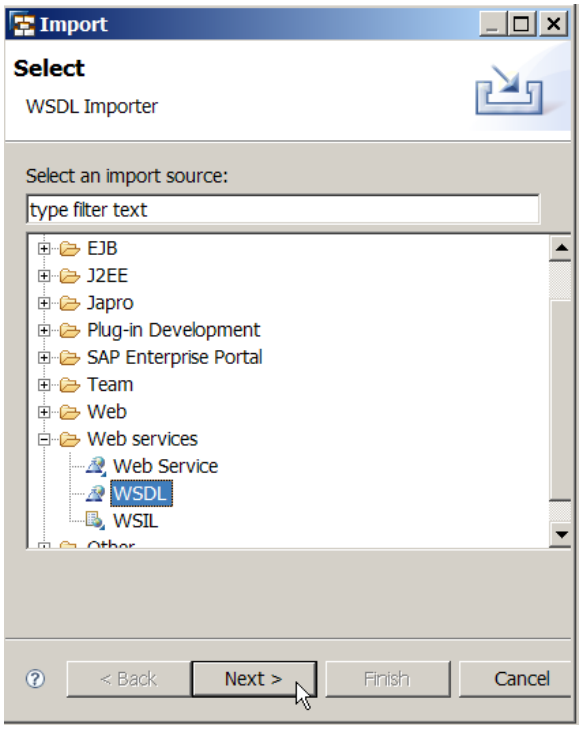
Import the Communication Channel Service and Change List Service

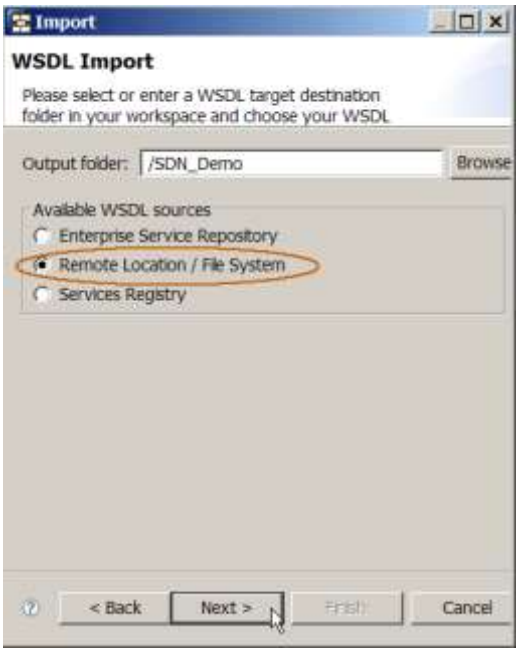
1. Obtain the WSDL URL from the WSNavigator:

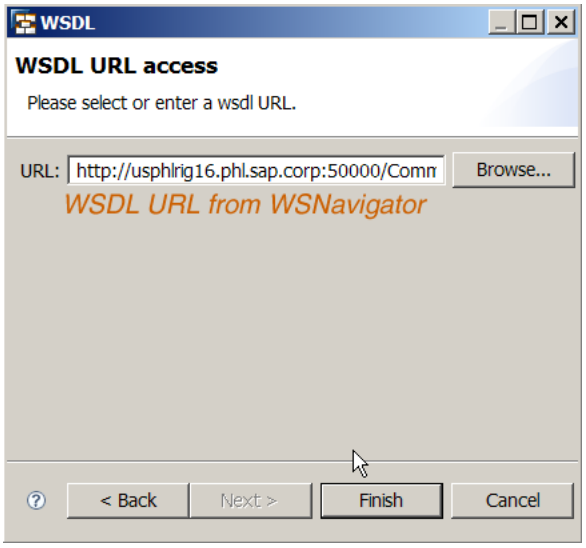


2. In NWDS, import the WSDL URL:

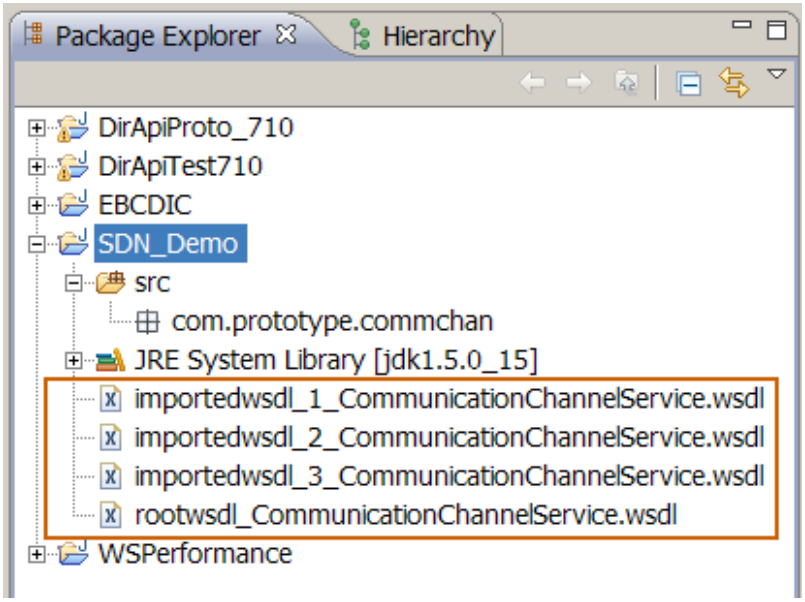






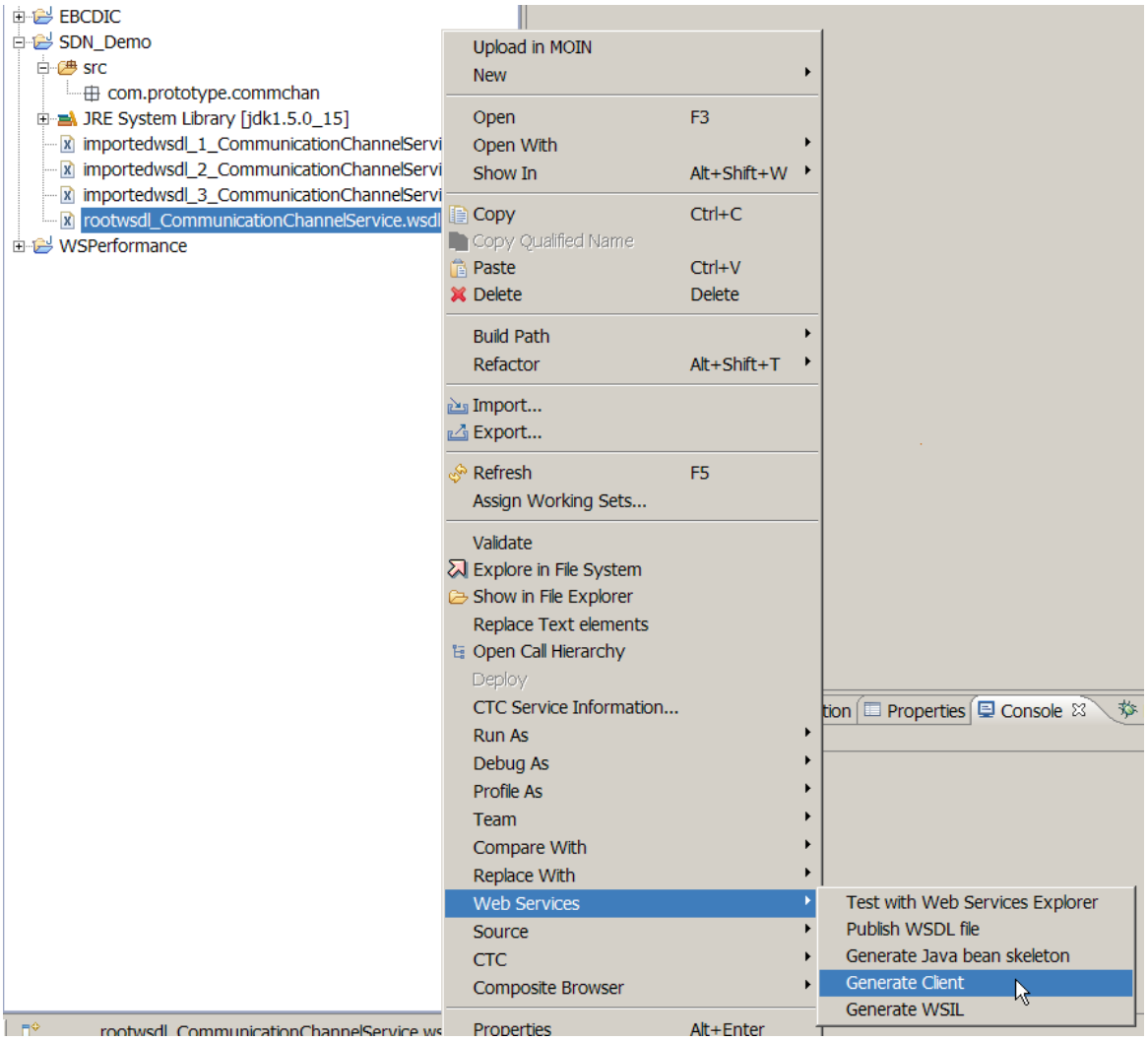


The following will result:

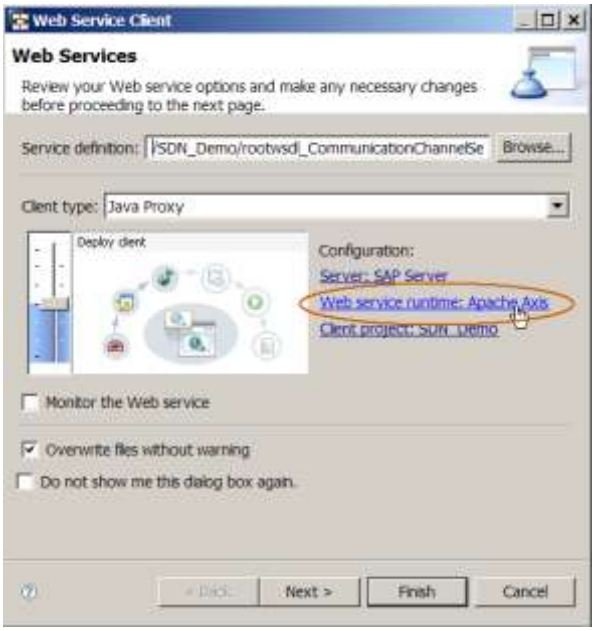


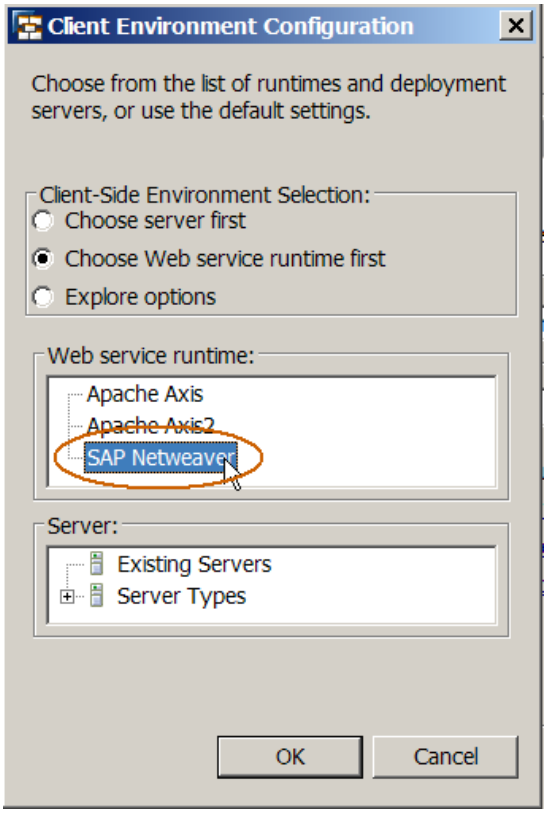
Generate Client Proxy for Each of the WSDL

1. Select “Web Services” → “Generate Client” from the menu:

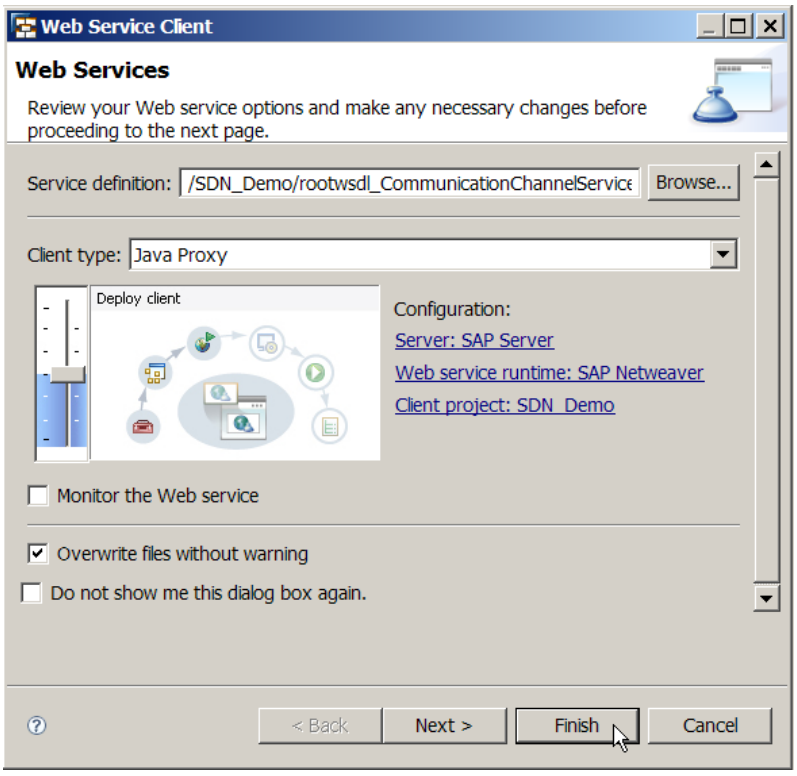


2. Change the Web Service Runtime to NetWeaver by clicking on “Web Service Runtime…”:

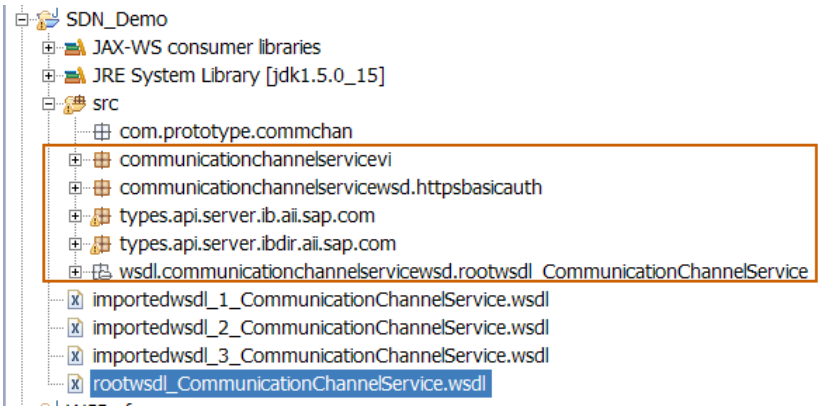




3. Click “Finish”:



Additional packages will be created:



4. Repeat for the other 3 WSDLs. Take default settings when prompted.

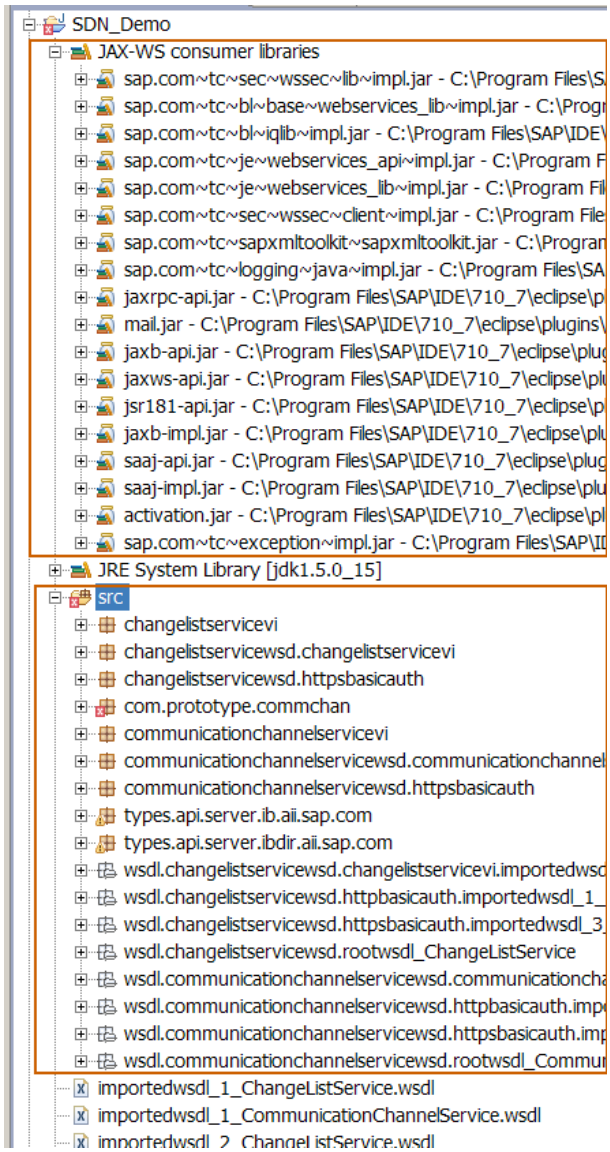
Repeat the Whole Process to Import the ChangeList service and Generate the Proxy

Copy the Source Program from the Un-Zipped Directory

Copy the contents “src \com\prototype\commchan” directory from the un-zipped file to the OS directory of the project’s “src \com\prototype\commchan” in NWDS.

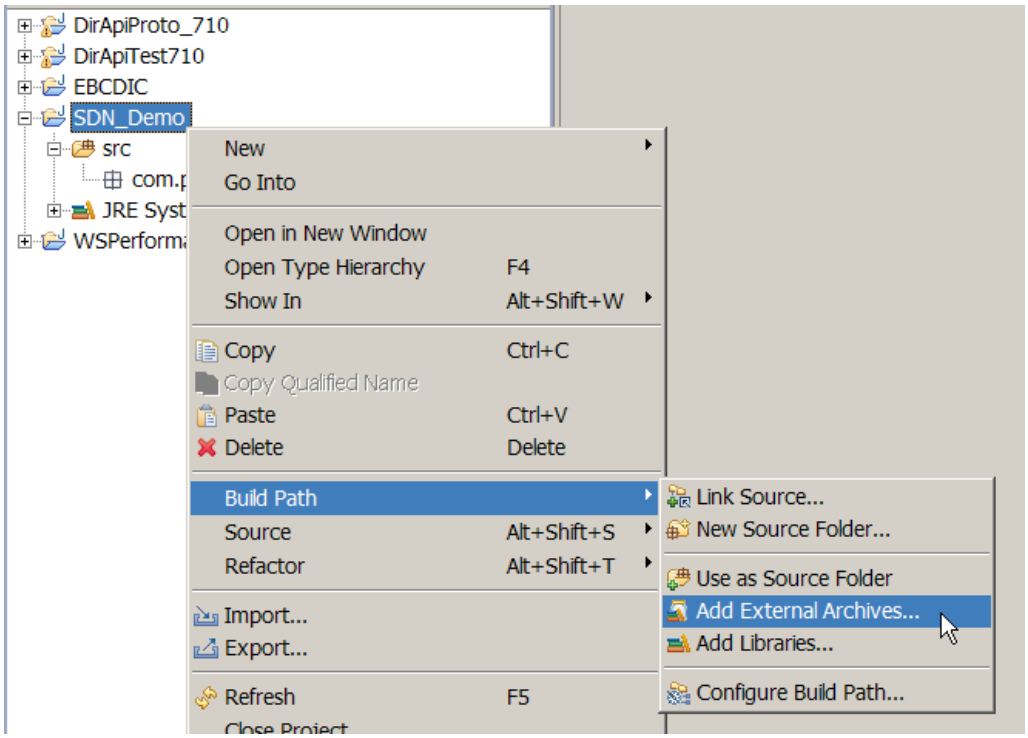
After copying, refresh the project. The following should result.

There are still unresolved references in the project due to references to the Excel library.



Import the Excel Library into the Build Path

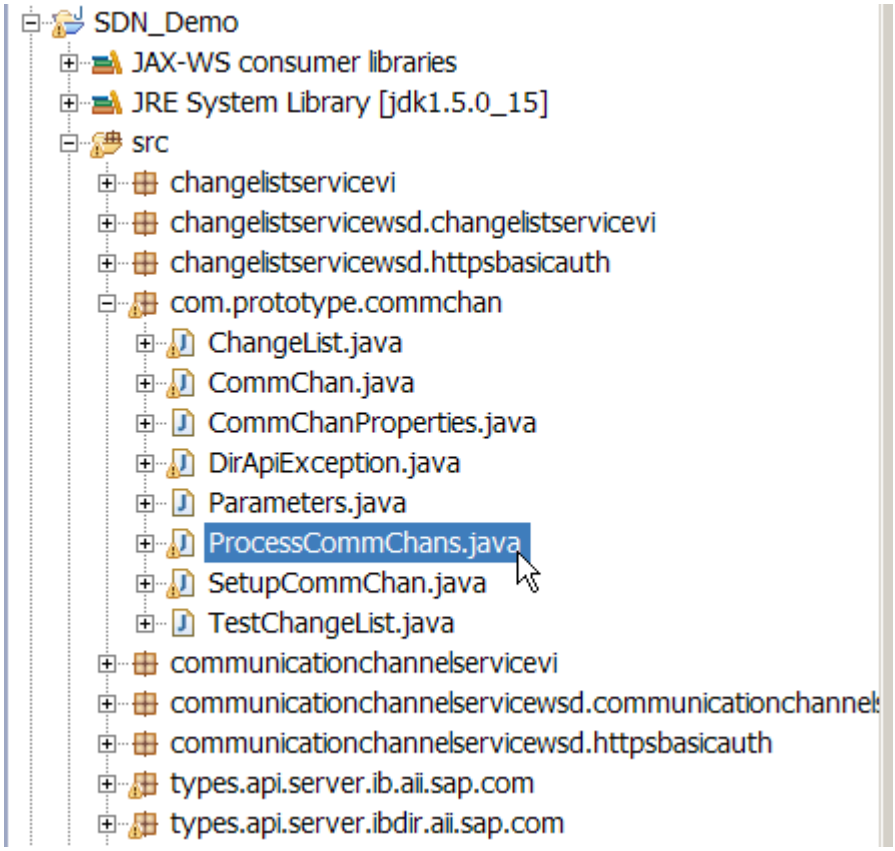
The Excel library or jar file can be found in the un-zipped file.



After importing the Excel jar file, all reference errors should be resolved.

Executing Program from NWDS

The class, ProcessCommChans.java, contains the main method initiating the execution. The runtime argument must be set with the Excel file name.



Related Content

SDN Blog - Directory API Development

SAP Help – Integration Directory (NW 2004) Programming Interface

SAP Help – Integration Directory 7.1 Programming Interface

Copyright