TIBCO iWay® Service Manager

Extensions User's Guide

Version 8.0 and Higher
March 2021
DN3502288.0321

Contents

Chapter 1

Introducing iWay Service Manager Extensions

This section provides an introduction to extensions offered for iWay Service Manager.

In this chapter:

About iWay Service Manager Extensions

Extensions supplement iWay Service Manager by adding new or extended capabilities for servicing messages. These extension services integrate with the native services of Service Manager during execution and for configuration. iWay itself provides several extensions that are fully supported as part of the Service Manager product. These extensions are installed as packages.

Chapter 2

iWay RVI Proxy Extension

The iWay RVI Proxy (also called RVI Gateway) extension links two or more iWay Service Managers in a message receiver and message executor relationship for the purpose of tunneling through secure firewalls.

Configuration of the iWay RVI Proxy extension takes place in the following order:

1. Installing the iWay Gateway extension on the iWay Proxy server and the execution engine.
2. Configuring the RVI Attach listener on the iWay Proxy.
3. Adding the RVI Relay service to the appropriate listener(s) configured on the Proxy server.
4. Configuring the RVI Gateway listener on the execution engine.

In this chapter:

RVI Proxy Extension Overview

Reverse Invocation (RVI) queue (also referred to as gateway) processing links two or more iWay Service Manager (iSM) instances in a message receiver or a message executor relationship to tunnel through secure firewalls.

To configure RVI queue (gateway) processing, you must:

1. Install the iWay Gateway extension on the iWay Proxy server and the execution engine.

   To install the iWay RVI Proxy, you must add the Gateway extension to your iSM instance during the iSM installation. For more information on installing iSM, see the *iWay Installation and Configuration Guide*.

   After the Gateway extension is installed, the RVIAttach listener, RVIGateway listener, and RVI Relay service are added to the design-time registry and run time configurations.
2. Configure the RVIAttach listener on the iWay Proxy server.
3. Add the RVI Relay service to the appropriate listener(s) configured on the iWay Proxy server.
4. Configure the RVIGateway listener on the execution engine.

RVI queue processing is now documented extensively in the *iWay Cross-Channel Services Guide*, which also describes how to:

- Configure the RVIAttach listener.
- Configure the RVI Relay service.
- Configure the RVIGateway listener.
- Configure a service to test the reverse invocation.

Chapter 3

iWay PGP Extension

This section describes how to configure the iWay Service Manager PGP (Pretty Good Privacy) extension using the iWay Service Manager Administration Console.

In this chapter:

iWay PGP Overview

PGP encryption uses public key cryptography and includes a system which binds the public keys to a user name and/or an email address. The first version of this system was generally known as a web of trust to contrast with the X.509 system, which uses a hierarchical approach based on certificate authority and which was added to PGP implementations at a later time. Current versions of PGP encryption include both alternatives through an automated key management server.

iWay Service Manager supports standard OpenPGP, as specified in RFC 2440. The support includes decrypting incoming messages and encrypting outgoing messages using simple PGP and key pair (public key) PGP.

With simple encryption, a message is encrypted with a symmetric key encoded by a pass phrase. A pass phrase is simply a long password. For example:

reality must take precedence over public relations.

Both parties must know the secret pass phrase. The decryption system enables the pass phrase to be configured with the decryptor, or taken from some other source such as a header field on the incoming document itself. Exposing the pass phrase in the document itself eliminates secrecy, but does prevent casual viewers from examining the content of documents.

Key Pair Encryption

Key pair encryption eliminates the need for the shared pass phrase. Also called asymmetric encryption or public key or private key encryption, this eliminates the need for the shared pass phrase. The sender of the message must know in advance the public key of the receiver, which can be obtained from a commercial source or, in standard PGP, generated by a local tool. The public key is exported by the recipient to the sender(s) either by sending a file that the sender can import into his public key ring, or by publishing it to a secure server, such as ldap://keyserver.pgp.com. The sender imports the public key from the server into his public key ring. At the same time that the public key is prepared, the private key is also prepared, and stored in the secret key ring.

The sender, using the public key of the recipient, encodes the session key. The recipient uses the private key to decode the session key. This eliminates the need to share the secret pass phrase, however it is more complicated to configure and use.

Digital Signature

A signature confirms the identity of the sender of an email. It confirms that an email has not been tampered or altered during transmission.

For signing, an algorithm that does work is to use a public key algorithm to encrypt only the signature. In particular, the hash value is encrypted using the private key of the signer, and anybody can check the signature using the public key. The signed document can be sent using any other encryption algorithm including none if it is a public document. If the document is modified, the signature check will fail. However, this is precisely what the signature check is supposed to catch. The Digital Signature Standard (DSA) is a public key signature algorithm that works just as described. DSA is the primary signing algorithm used in GnuPG.

Supported algorithms include:

Algorithm	Description
none	Text is not encrypted.
cast5	128-bit key as per RFC 2144. This is the default.

Algorithm	Description
blowfish	128-bit key, 16 rounds. A symmetric block cipher like DES and IDEA. Generally fast.
safer	SAFER-SK 128-bit, 13 rounds, using a secure key schedule. It does not operate with blocks, unlike IDEA and DES.
triple DES	DES-EDE, 168-bit key derived from 192 bits.
idea	A DES-like block cipher algorithm that uses a 128-bit key length to encrypt successive 64-bit blocks of plain text.

Supported Components

This section lists and describes the iWay components that are supported by the iWay PGP extension.

- **Preparser**

  com.ibi.preparsers.PGPDecrypt (Incoming documents)

  The PGPDecrypt preparser decrypts an incoming message into the original unencrypted format.

- **Preemitter**

  com.ibi.preemit.PGPEncrypt (Outgoing Documents)

  The PGPEncrypt preemitter encrypts an outgoing message into an encrypted XML document. This must be the last preemitter in a chain, since a channel cannot process the encrypted document unless it is decrypted by a preparser first.

- **Service**

  com.ibi.agents.PGPEncrypt (Outgoing Documents)

  The PGPEncrypt service also performs the same functionality as the preemitter. It encrypts outgoing documents. Like the preemitter, it needs to be the last component in the channel before the emitter, since the document would be in encrypted form. It also provides the same parameters as the preemitter.

Encrypting Outgoing Documents

Outgoing documents can be encrypted in PGP using the PGP preemitter. The following table lists and describes the available parameters:

Parameter	Type	Description
Pass Phrase or Alias	Text	For simple encryption, this is the agreed upon pass phrase. For key pair, this is the public key alias of the recipient. Can be an SREG() or XPATH() specification. The form of an alias depends upon the key ring that is used.
Public Key Ring	Path	Full path to the public key ring. Used for key pair encryption.
Secret Key Ring	Path	Full path to the secret key ring. Used for key pair encryption.
Armor	Boolean	If set, an armored message is generated. For most purposes, armoring should be set ON.
Algorithm	Enumeration	The algorithm that is used. Select one of the following: ❏ none ❏ cast5 ❏ blowfish ❏ safer ❏ triple DES ❏ IDEA
FingerPrint	TEXT(Hex Decimal Number)	Fingerprint of the Encryption Hex Decimal Key.

Note: The FingerPrint parameter is helpful when selecting a specific key from a list of available keys. If the FingerPrint parameter is not used, then the first available encryption keys for the encryption of data is used.

```
pub   1024D/584E38E6 2009-02-26
   Key fingerprint = 5E1F 0BEC A314 6379 EBA4  97EA 9925 772A 584E 38E6
uid     elgam (iway) <elgam@ibi.com>
sub  3008g/8D86CFF8 2009-02-26
   Key fingerprint = 2109 1680 A87E DA48 BF84  AA9A 237E D723 8D86 CFF8
sub   1088R/992532D9 2009-03-17
   Key fingerprint = 4389 BD56 9B53 A7BB AD60  AACE 8008 85F2 9925 32D9
```

Listing Multiple Sub Keys

Type the following command at the command prompt:

```
gpg --fingerprint --fingerprint elgam
(iway) <elgam@ib.com>
```

where:

elgam (iway) <elgam@ib.com>

Is the alias being used.

The following is a sample listing of multiple sub keys:

```
pub   1024D/584E38E6 2009-02-26
   Key fingerprint = 5E1F 0BEC A314 6379 EBA4  97EA 9925 772A 584E 38E6
uid          elgam (iway) <elgam@ib.com>
sub   3008g/8D86CFF8 2009-02-26
   Key fingerprint = 2109 1680 A87E DA48 BF84  AA9A 237E D723 8D86 CFF8
sub    1088R/992532D9 2009-03-17
   Key fingerprint = 4389 BD56 9B53 A7BB AD60  AACE 8008 85F2 9925 32D9
```

Decrypting Incoming Documents

Any incoming document can be PGP-encoded. Decoding is performed using the PGPDecode preparser. The decryptor works with either simply encoded or key pair encoded messages. The pass phrase, used for simple decryption, can be specified directly, or as the content of a special register. The following table lists and describes the available parameters:

Parameter	Type	Description
Decrypt method	Pass phrase or key pair	Selects the form of decryption to be used.
Pass Phrase	Text	For simple encryption, this is the agreed upon pass phrase. For key pair decryption, this value is ignored.

Parameter	Type	Description
Key Phrase	Text	Phrase used to unlock the secret key ring. Used for key pair encryption.
Public Key Ring	Path	Full path to the public key ring. Used for key pair encryption.
Secret Key Ring	Path	Full path to the secret key ring. Used for key pair encryption.

Installing and Configuring the iWay PGP Extension

To install the iWay Service Manager PGP (Pretty Good Privacy) extension, you must first add the PGP encryption components to your iWay Service Manager instance during the iWay Service Manager installation. For more information on installing iWay Service Manager, see the *iWay Installation and Configuration Guide*.

Installing and Configuring the iWay PGP Extension

Once you have added the PGP extension to your iWay Service Manager instance, you are ready to install and configure the PGP extension.

You must first obtain the following .JAR files:

- bcpg_jdkxx_<version>.jar
- bcprov_jdkxx_<version>.jar

Make sure to download the latest versions of the .JAR files from the following website:

`http://www.bouncycastle.org`

These files must be copied to the following directory:

iwayhome`\lib`

where:

iwayhome

Is the location where iWay Service Manager is installed.

The iwpgp.jar file is also required and is provided with the iWay PGP extension installation. This file must be copied to the following directory:

iwayhome`\etc\manager\extensions`

where:

iwayhome

Is the location where iWay Service Manager is installed.

Download the GNU Privacy Guard (gnupg-w32cli-1.4.9.exe) from the following website:

`http://www.gnupg.org/download/`

Procedure: How to Install and Configure the iWay PGP Extension

To install and configure the iWay PGP extension:

1. Execute the *gnupg-w32cli-1.4.9.exe* file to install the GNU Privacy Guard.
2. Create a system variable called *GNUPGHOME*, which is the directory where your keys are created. For example:

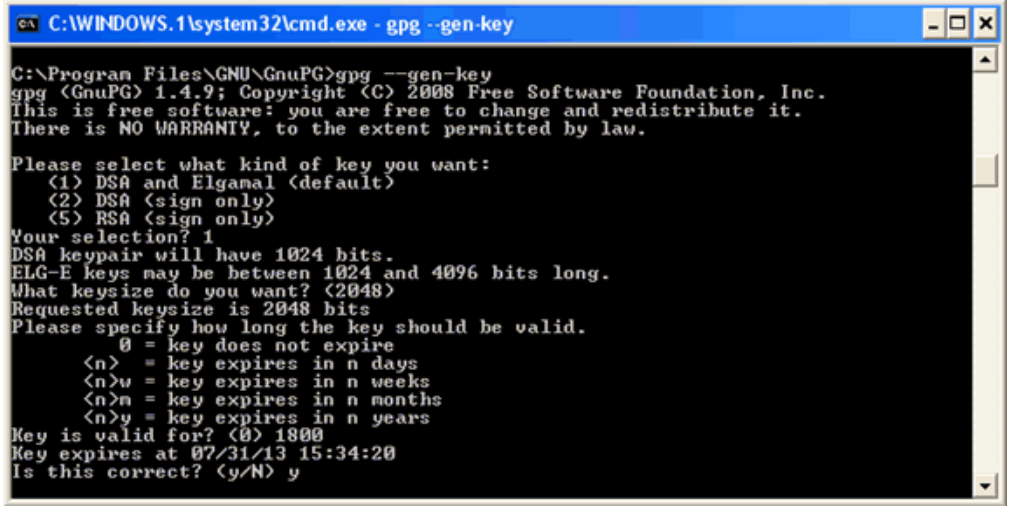
   ```
   C:\Program Files\GNU\GnuPG\keys
   ```

3. Add your GNU Privacy Guard installation directory to the System Path. For example:

   ```
   C:\Program Files\GNU\GnuPG
   ```

4. Open a command prompt and navigate to your GNU Privacy Guard installation directory.
5. Execute the following command:

   ```
   gpg --gen-key
   ```

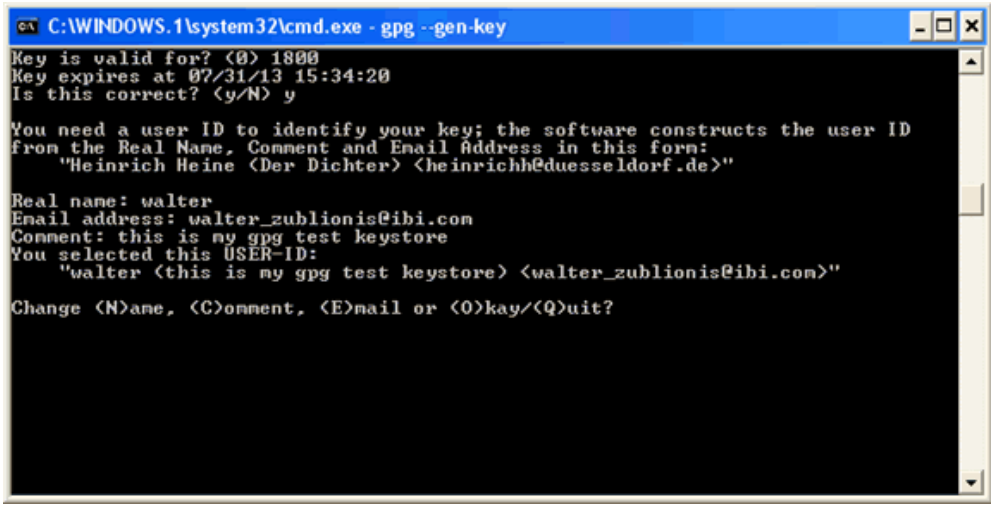


6. Select *5* for RSA key and follow the prompts.

You will be prompted to specify the Real Name. Ensure that the Real Name is not too large, since you will have to type it frequently.

7. As an example, enter *walter* for the Real Name.
8. Enter *walter_zublionis@ibi.com* for the Email Address.
9. Enter *this is my gpg test keystore* for the Comment.

   Thus, the alias or user ID is *walter (this is my gpg test keystore) <walter_zublionis@ibi.com>*. This user ID or the alias is used to identify or lookup the key for editing. This should be noted for future reference.



10. Enter *hello* when prompted for a pass phrase.

    Once confirmed, you will have a secret and public key generated in the GNU Privacy Guard home directory. At this point, your key is not enabled for encryption. You must create a subkey.

11. Execute the following command:

    ```
    gpg --edit-key "walter (this is my gpg test keystore)
    <walter_zublionis@ibi.com>”
    ```

    You are now in the GNU Privacy Guard Edit mode.

12. Enter *addkey* at the command prompt.

    You are prompted for a pass phrase.

13. Enter *hello* at the command prompt.

    The following menu is displayed:

```
Please select what kind of key you want:
(2) DSA (sign only)
(4) Elgamal (encrypt only)
(5) RSA (sign only)
(6) RSA (encrypt only)
```

14. Select *6* to add an RSA encryption key.
15. Enter *list* at the command prompt.

    The pub(public) and sub(private) keys are now available, as shown in the following example:

```
pub  1024R/FA9F8DE1  created: 2008-09-10  expires: never  usage: SC
trust: ultimate      validity: ultimate
sub  1024R/A3DACF28  created: 2008-09-10  expires: never  usage: E
[ultimate] (1). walter (hello) <walter_zublionis@ibi.com>
```

    Your key is now enabled for encryption.

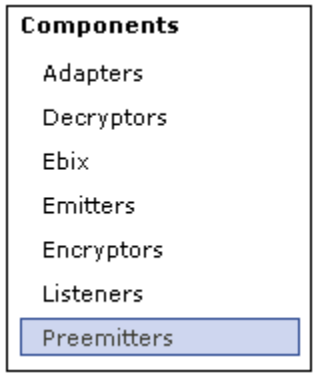
Configuring a PGP Preemitter for Encryption

The following section describes how to configure a PGP preemitter for encryption.

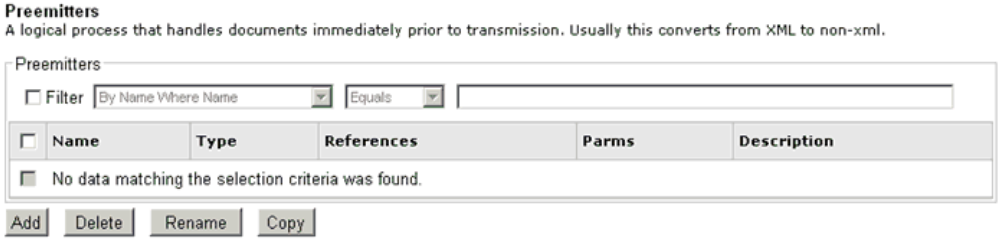
Procedure: How to Configure a PGP Preemitter

To configure a PGP preemitter:

1. In the left console pane of the registry menu, select *Preemitters*.

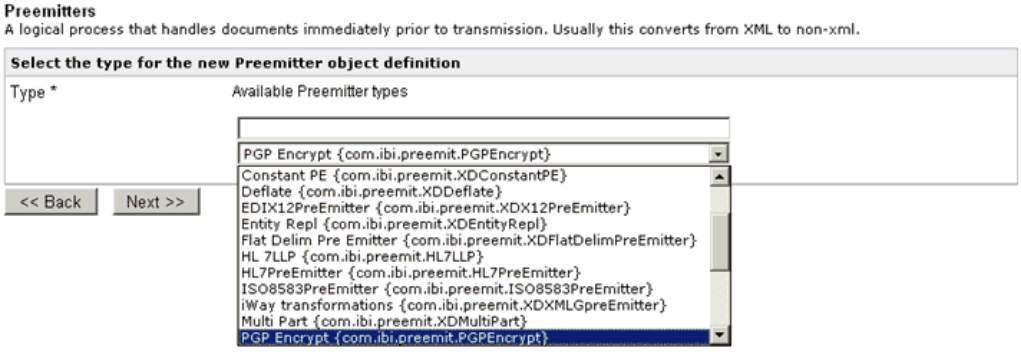


The Preemitters pane opens as shown in the following image.

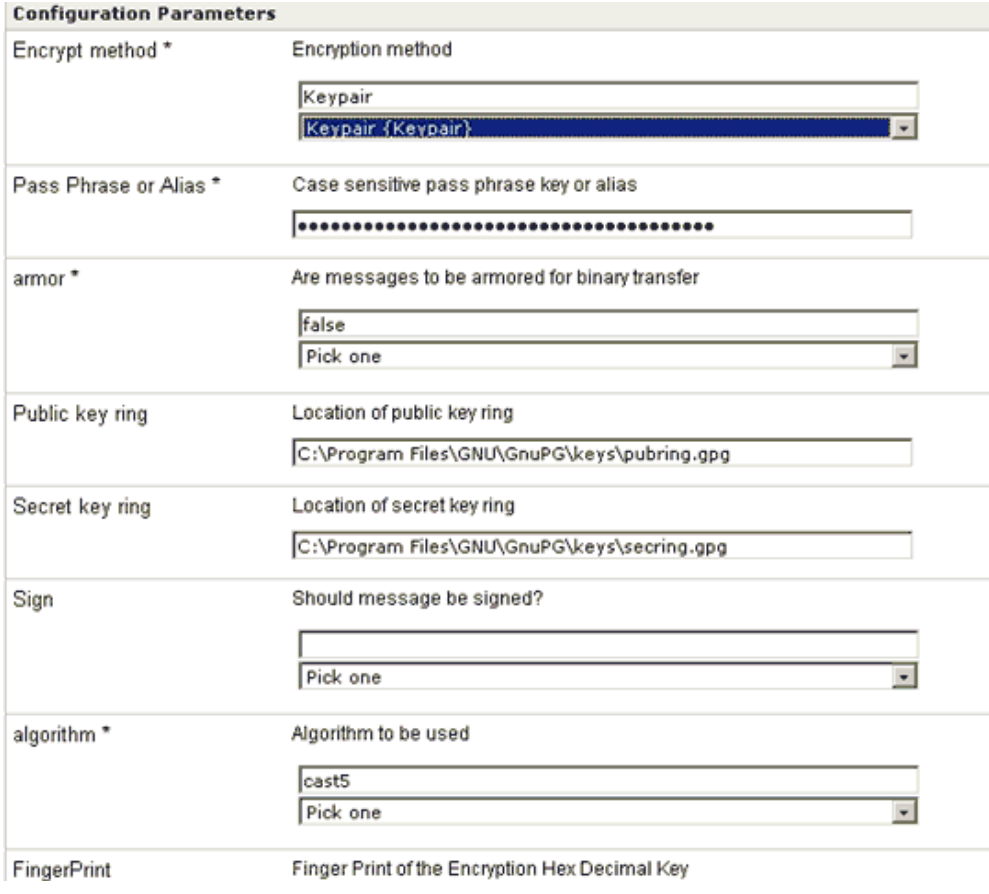


The table provided lists any existing preemitters and short descriptions of each.

2. Click *Add*.
3. Select *PGP Encrypt {com.ibi.preemit.PGPEncrypt}* from the list, and then click *Next*.



4. Provide the required configuration parameters for the new preemitter, as described in *PGP Preemitter Configuration Parameters* on page 22.



If the Encrypt method is set to Keypair, then the pass phrase (alias) should be entered for encryption. In the example that is used, the alias is walter (this is my gpg test keystore) <walter_zublionis@ibi.com>. If the Encrypt method is set to Passphrase, then the pass phrase should be entered. In the example that is used, the pass phrase is hello.

If more than one key is present in the same pub key ring file, then the fingerprint needs to be entered while encrypting the message to indicate the specific key to use for encryption. The fingerprint can be obtained as explained in *Listing Multiple Sub Keys* on page 15.

5. Click *Next*.

   The Name and Description pane opens.

6. Enter a name and an optional description for the preemitter and click *Finish*.

The preemitter is added to the list in the Preemitters pane.

After a preemitter is added to iWay Service Manger, you can assign a preemitter to an outlet that is used to construct a channel.

Reference: **PGP Preemitter Configuration Parameters**

Parameter	Description
Encrypt method	Selects the form of encryption to be used. In this example, select Keypair from the drop-down list.
Pass Phrase or Alias	The configured pass phrase or alias. In this example, the following value is used: `walter (hello) <walter_zublonis@ibi.com>` **Note:** To avoid typos, it is a good idea to cut and paste this value from a text file.
armor	Determines whether an armored message should be generated. In this example, select false from the drop-down list.
Public key ring	Full path to the public key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\pubring.gpg`
Secret key ring	Full path to the secret key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\secring.gpg`
Sign	Determines whether messages should be signed. In this example, select false from the drop-down list.
algorithm	The algorithm that is used. In this example, select cast5 from the drop-down list.
FingerPrint	Fingerprint of the Encryption Hex Decimal Key. In this example, do not provide a value for this parameter. This value can be specified if you are using a key ring, which contains multiple keys.

Configuring a PGP Preparser for Decryption

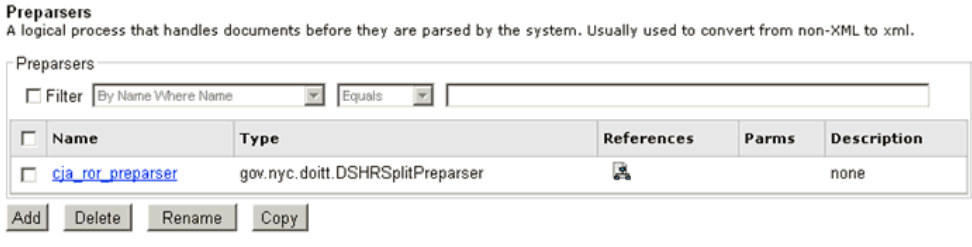
The following section describes how to configure a PGP preparser for decryption.

Procedure: How to Configure a PGP Preparser

To configure a PGP preparser:

1. In the left console pane of the registry menu, select *Preparsers*.

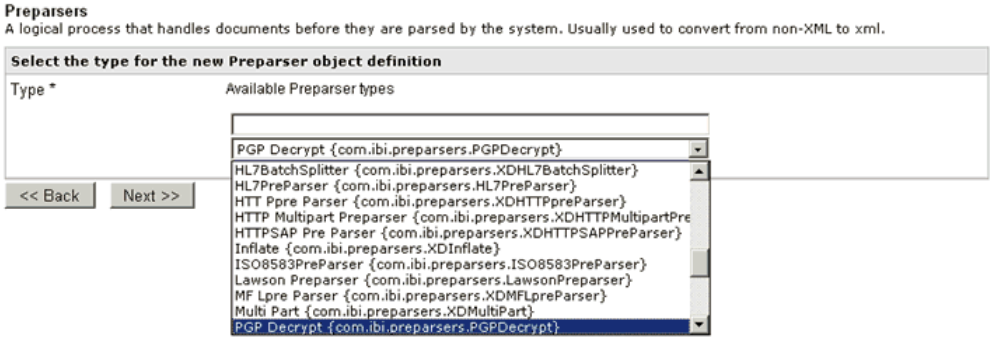
The Preparsers pane opens as shown in the following image.



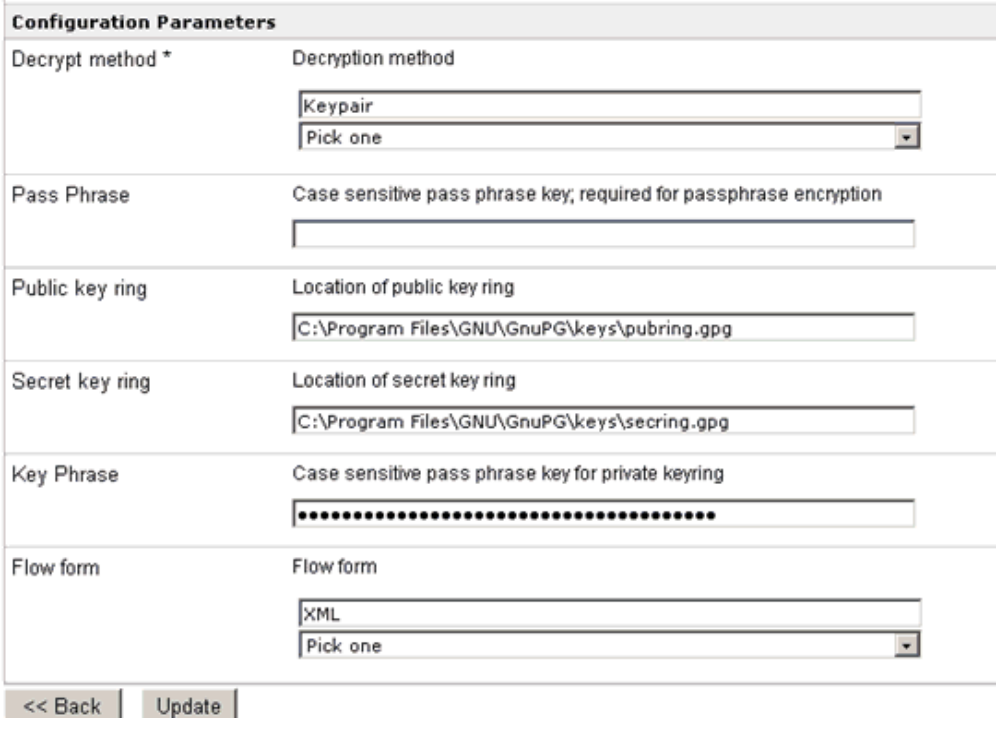
The table provided lists any existing preparsers and short descriptions of each.

2. Click *Add*.

3. Select *PGP Decrypt {com.ibi.preparsers.PGPDecrypt}* from the list, and then click *Next*.



4. Provide the required configuration parameters for the new preparser, as described in *PGP Preparser Configuration Parameters* on page 25.



If the Decrypt method is set to Keypair, then the pass phrase (private key) should be entered in the Key Phrase field. In the example that is used, the pass phrase is hello. If the Decrypt method is set to Passphrase, then the pass phrase should be entered in the Pass Phrase field. In the example that is used, the pass phrase is hello.

5. Click *Next*.

   The Name and Description pane opens.

6. Enter a name and an optional description for the preparser and click *Finish*.

   The preparser is added to the list in the Preparsers pane.

   After a preparser is added to iWay Service Manger, you can assign a preparser to an inlet that is used to construct a channel.

Reference: **PGP Preparser Configuration Parameters**

Parameter	Description
Decrypt method	Selects the form of decryption to be used. In this example, select Keypair from the drop-down list.
Pass Phrase	For simple encryption, this is the agreed upon pass phrase. For key pair decryption, this value is ignored.
Public key ring	Full path to the public key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\pubring.gpg`
Secret key ring	Full path to the secret key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\secring.gpg`
Key Phrase	Phrase used to unlock the secret key ring. Used for key pair encryption.
Flow form	Determines the flow form to be used. In this example, select *XML* from the drop-down list.

Testing Document Encryption and Decryption

You can test document encryption and decryption by constructing a channel that uses the preemitter (com.ibi.preemit.PGPEncrypt) and preparser (com.ibi.preparsers.PGPDecrypt) that was configured.

Procedure: How to Test Document Encryption and Decryption

To test document encryption and decryption:

1. Add the preemitter (com.ibi.preemit.PGPEncrypt) that you configured to an outlet, for example, *pgpoutlet*.
2. Construct a channel, for example, *PGPSender*, with a file listener, a move route, and the outlet (pgpoutlet).
3. Create a new inlet, for example, *PGPInlet*, and add a file listener to this inlet and the preparser (com.ibi.preparsers.PGPDecrypt) that you configured.
4. Construct a second channel, for example, *PGPReceiver*, with the *PGPInlet*, a move route, and a default outlet.
5. Build, deploy, and, start both the channels.
6. Place a file to be encrypted, for example, *hello.xml*, in the *PGPSender* channel.
7. Pick up the file at the outlet of the *PGPSender* channel and place it in the *PGPReceiver* channel.

   The file obtained at the default outlet of the *PGPReceiver* channel should be the same as the original hello.xml file before encryption.

Using the PGPEncrypt Service

The PGPEncrypt service emits byte messages which are input to a following service for transmission to a recipient. In a process flow, for example, this would place the PGPEncrypt service just before an Emit service on an edge. This edge usually has an End object that follows the emit operation, since the message is encrypted and cannot be used further in the process flow.

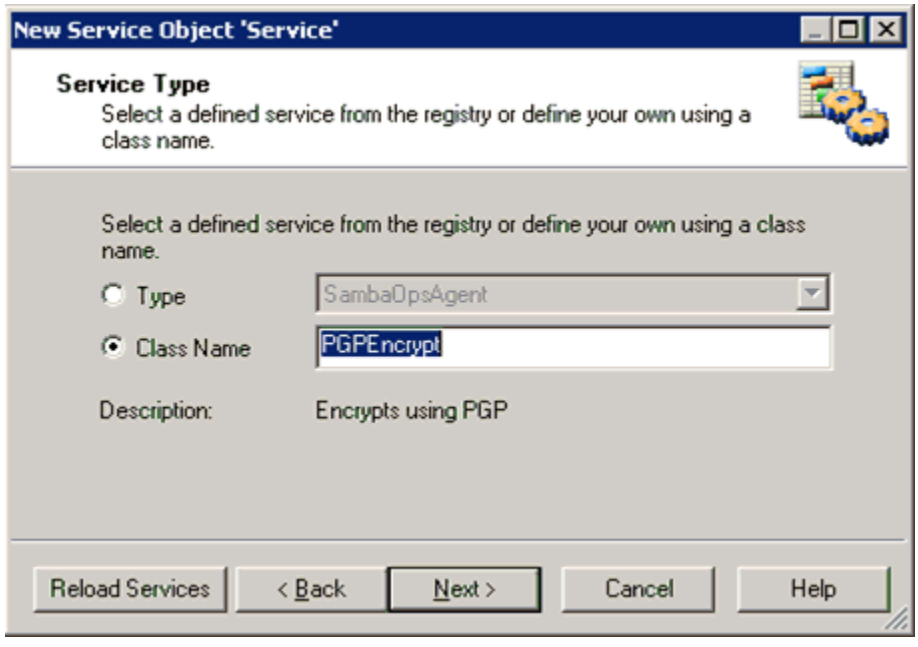
Procedure: How to Configure the PGPEncrypt Service

To configure the PGPEncrypt service:

1. Using iWay Designer, drag and drop the Service object icon from the toolbar to the workspace.

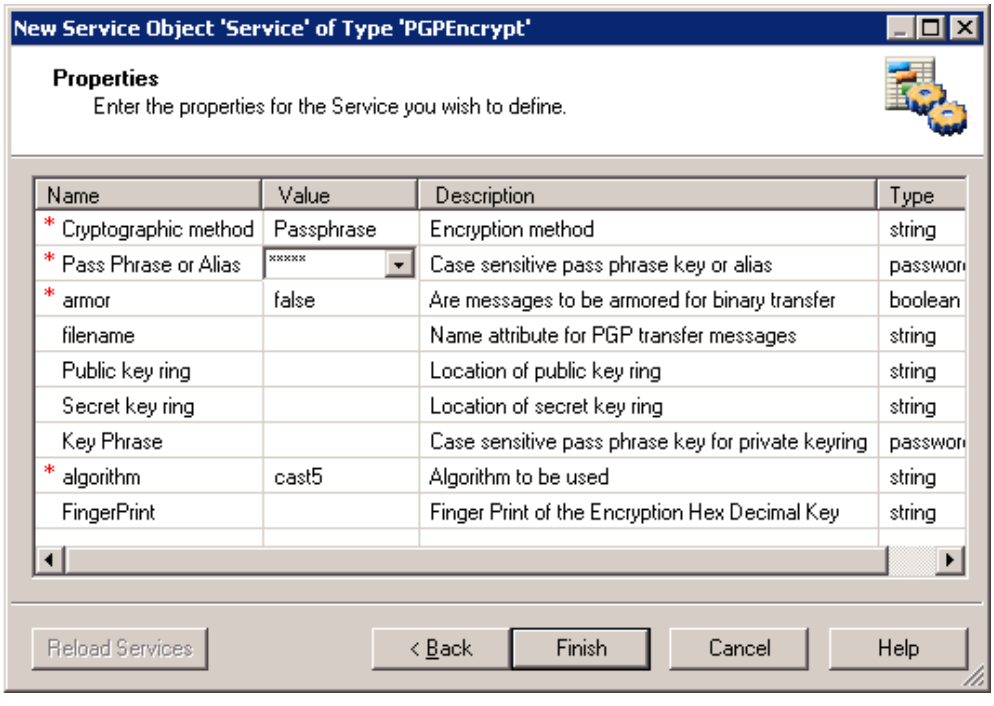
   The Service Name and Description dialog box opens.
2. In the Name field, type a new name for this Service object, and leave the default value (Service object) in the Description field.
3. Click *Next*.

The Service Type dialog box opens.



4. Select *Class Name* and enter *PGPEncrypt*.
5. Click *Next*.

The Properties dialog box opens, as shown in the following image.



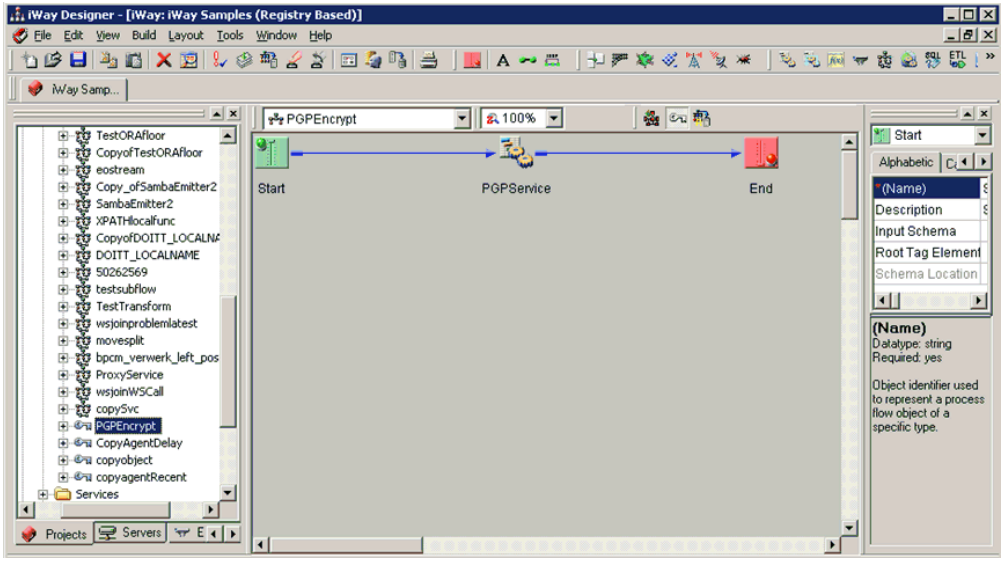
6. Provide the appropriate values for the properties as shown in this example.

   If *Passphrase* is specified for the Cryptographic method property, enter the pass phrase, otherwise enter the alias for the key pair method.

7. Click *Finish*.

   The new Service object appears in the workspace.

You can now construct a process flow for the PGPEncrypt Service object with a Start and an End edge.



Since the file being used is encrypted before any action is performed, the file must be decrypted. In this example, test the process flow with a channel and write the encrypted output to a directory.

8. Validate the process flow and publish it to the Registry for use in channel configuration.
9. Using the iWay Service Manager Administration Console, add the process flow to a new route (for example, pgpencryptRt).
10. Construct a new channel (for example, PGPEncryptSvc).
11. Define an inlet for the channel, which consists of a File listener to pick up the encrypted file.
12. Add the defined route (for example, pgpencryptRt) to the channel.
13. Define a default outlet for the channel.
14. Build, deploy, and start the channel.
15. Input a file (for example, hello.xml) to be encrypted by the PGPEncryptSvc channel.
16. Pick up the encrypted file at the default outlet of the PGPEncryptSvc channel.
17. Input the encrypted file to be picked up by the PGPReceiver channel, which was configured earlier in *Testing Document Encryption and Decryption* on page 25.

The file obtained at the default outlet of the PGPReceiver channel should be the same as the original hello.xml file before encryption.

Signing and Encrypting a Message Using Key Pair Encryption

The current encryption model supports the signing of messages along with encryption. A digital signature is added to each encrypted message. The PGP preemitter and PGPEncrypt service do not support the signing of a message without encryption.

Consider a use case where an outgoing message must be signed and encrypted. In addition, assume that there are multiple keys listed within the same key ring file. For example:

```
H:\>gpg --list-key C:/Program Files/GNU/GnuPG/keyRing\pubring.gpg
------------------------------------------------------------------
pub   1024D/3A4A61BD 2009-05-13
uid     Key for Test server<Soumya_raghavan@ibi.com>
sub   2048g/92DFC2B0 2009-05-13
pub   1024D/DB9570DD 2000-02-23
uid     Production <production@ibi.com>
sub   2048g/8DC224F9 2000-02-23
pub   1024D/FFBDBE5C 2009-04-30
uid     ATSHelp <ATS 24X7 support @ibi.com>
sub   2048g/B0BB9ED1 2009-04-30
```

To successfully encrypt a message using key pair encryption:

1. Follow the steps in *How to Configure a PGP Preemitter* on page 19.

   The values for the PGP preemitter configuration parameters are listed and described in the following table:

Parameter	Description
Encrypt method	Selects the form of encryption to be used. In this example, select Keypair from the drop-down list.
Pass Phrase or Alias	The configured pass phrase or alias. In this example, the following value is used: `Key for Test server<Soumya_raghavan@ibi.com>` **Note:** To avoid typos, it is a good idea to cut and paste this value from a text file.

Parameter	Description
armor	Determines whether an armored message should be generated. In this example, select false from the drop-down list.
Public key ring	Full path to the public key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\pubring.gpg`
Secret key ring	Full path to the secret key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\secring.gpg`
Key Phrase	Enter the key phrase that is required for signing. This parameter is required only when signing.
Sign	Determines whether messages should be signed. In this example, select true from the drop-down list.
algorithm	The algorithm that is used. In this example, select cast5 from the drop-down list.
FingerPrint	Enter the fingerprint of the sub key ring (the encryption key), which can be obtained by using the following command: `gpg --fingerprint --fingerprint "alias"` where: *alias* Is the alias being used.

2. Follow the steps in *How to Configure a PGP Preparser* on page 23.

   The values for the PGP preparser configuration parameters are listed and described in the following table:

Parameter	Description
Decrypt method	Selects the form of decryption to be used. In this example, select Keypair from the drop-down list.

Parameter	Description
Pass Phrase or Alias	The real name that is configured. In this example, the following value is used: `Key for Test server<Soumya_raghavan@ibi.com>` **Note:** To avoid typos, it is a good idea to cut and paste this value from a text file.
Public key ring	Full path to the public key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\pubring.gpg`
Secret key ring	Full path to the secret key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\secring.gpg`
Key Phrase	Enter the key phrase, which is required to verify the signature.
Flow form	Determines the flow form to be used. In this example, select XML from the drop-down list.

3. Construct and deploy two channels (for example, *EncryptChannel* and *DecryptChannel*).
4. Input an XML file (for example, hello.xml) to be encrypted by the encrypt channel (EncryptChannel).
5. Pick up the encrypted file from the default output of *EncryptChannel* and use this file as input for the decrypt channel (DecryptChannel).

   The original file is obtained.

Configuring ASCII Message Armoring

If you need an encrypted file to be in ASCII format, ASCII message armoring must be set to true. This is useful if you need to email an encrypted file in ASCII format.

GPG Command Line

From the gpg command reference, the -a option must be set in the command.

ASCII Armour: Code all PGP output files in printable ASCII characters using Radix 64. a can be used on its own to convert any file to ASCII-armoured.

The following example shows how to create an ASCII armoured file that provides an encrypted file in ASCII format:

```
H:\>gpg --passphrase-file c:\passphrase.txt --sign --encrypt
a -r "soumya" c:\M
on.txt
Reading passphrase from file descriptor 3
You need a passphrase to unlock the secret key for
user: "soumya (sou's key) <soumya_raghavan@ibi.com>"
2048-bit RSA key, ID BC58F8F3, created 2009-03-19
```

Configuring a Preemitter and an Agent for ASCII Message Armoring

The current encryption supports ASCII message armoring. However, ASCII armoring cannot be performed if the message needs to be signed. If the message is signed and encrypted, the output file would be generated only in binary format.

Consider a scenario where an outgoing message must be encrypted in ASCII format. To successfully encrypt a message in this case:

1. Follow the steps in *How to Configure a PGP Preemitter* on page 19.

   The values for the PGP preemitter configuration parameters are listed and described in the following table:

Parameter	Description
Encrypt method	Selects the form of encryption to be used. In this example, select Keypair from the drop-down list.
Pass Phrase or Alias	The configured pass phrase or alias. In this example, the following value is used: `Key for Test server<Soumya_raghavan@ibi.com>` **Note:** To avoid typos, it is a good idea to cut and paste this value from a text file.
armor	Determines whether an armored message should be generated. In this example, select true from the drop-down list.
Public key ring	Full path to the public key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\pubring.gpg`

Parameter	Description
Secret key ring	Full path to the secret key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\secring.gpg`
Key Phrase	Enter the key phrase that is required for signing. This parameter is required only when signing.
Sign	Determines whether messages should be signed. In this example, select false from the drop-down list.
algorithm	The algorithm that is used. In this example, select cast5 from the drop-down list.
FingerPrint	Enter the fingerprint of the sub key ring (the encryption key), which can be obtained by using the following command: `gpg --fingerprint --fingerprint "alias"` where: *alias* Is the alias being used.

2. Follow the steps in *How to Configure a PGP Preparser* on page 23.

   The values for the PGP preparser configuration parameters are listed and described in the following table:

Parameter	Description
Decrypt method	Selects the form of decryption to be used. In this example, select Keypair from the drop-down list.
Pass Phrase or Alias	The real name that is configured. In this example, the following value is used: `Key for Test server<Soumya_raghavan@ibi.com>` **Note:** To avoid typos, it is a good idea to cut and paste this value from a text file.

Parameter	Description
Public key ring	Full path to the public key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\pubring.gpg`
Secret key ring	Full path to the secret key ring. Used for key pair encryption. In this example, the following path is used: `C:\Program Files\GNU\GnuPG\keys\secring.gpg`
Key Phrase	Enter the key phrase, which is required to verify the signature.
Flow form	Determines the flow form to be used. In this example, select XML from the drop-down list.

3. Construct and deploy two channels (for example, *EncryptChannel* and *DecryptChannel*).
4. Input an XML file to be encrypted by the encrypt channel (EncryptChannel).
5. Pick up the encrypted file in ASCII format from the default output of *EncryptChannel* and use this file as input for the decrypt channel (DecryptChannel).

   The original file is obtained.

JCE Reference

The JCE Cryptography polices by default are limited, which does not allow certain encryption algorithms to function properly.

By default, the following permissions are installed:

```
// File: default_local.policy
// Some countries have import limits on crypto strength.
// This policy file is worldwide importable.
grant { permission javax.crypto.CryptoPermission "DES", 64;
   permission javax.crypto.CryptoPermission "DESede", *;
   permission javax.crypto.CryptoPermission "RC2",
   128, "javax.crypto.spec.RC2ParameterSpec", 128;
   permission javax.crypto.CryptoPermission "RC4", 128;
   permission javax.crypto.CryptoPermission "RC5", 128,
   "javax.crypto.spec.RC5ParameterSpec", *, 12, *;
   permission javax.crypto.CryptoPermission "RSA", 2048;
   permission javax.crypto.CryptoPermission *, 128; };
```

With Unlimited Jurisdiction, all the algorithms are supported:

```
// File: default_local.policy
// Country-specific policy file for countries with no limits on
// crypto strength.
grant { // There is no restriction to any algorithms. permission
javax.crypto.CryptoAllPermission; };
Unlimited Jurisdiction can be applied on installing the JCE policy files
from Sun - Java site - with respect to a JRE version.
```

Chapter 4

iWay Telnet Extension

This section describes how to configure the iWay Service Manager Telnet extension using the iWay Service Manager Administration Console.

In this chapter:

iWay Telnet Extension Overview

The iWay Telnet extension is used to remotely access the iWay Service Manager (iSM) command line console through a Telnet session. A Telnet client session can connect to any iSM instance running either in the foreground or the background.

Note: Access to the command line console cannot be performed locally when the iSM instance is running in the background, for example, a Windows service.

Installing and Configuring the iWay Telnet Extension

To install Telnet, you must add the Telnet extension to your iWay Service Manager instance during the iWay Service Manager installation. For more information on installing iWay Service Manager, see the *iWay Installation and Configuration Guide*.

Configuring the Telnet Listener

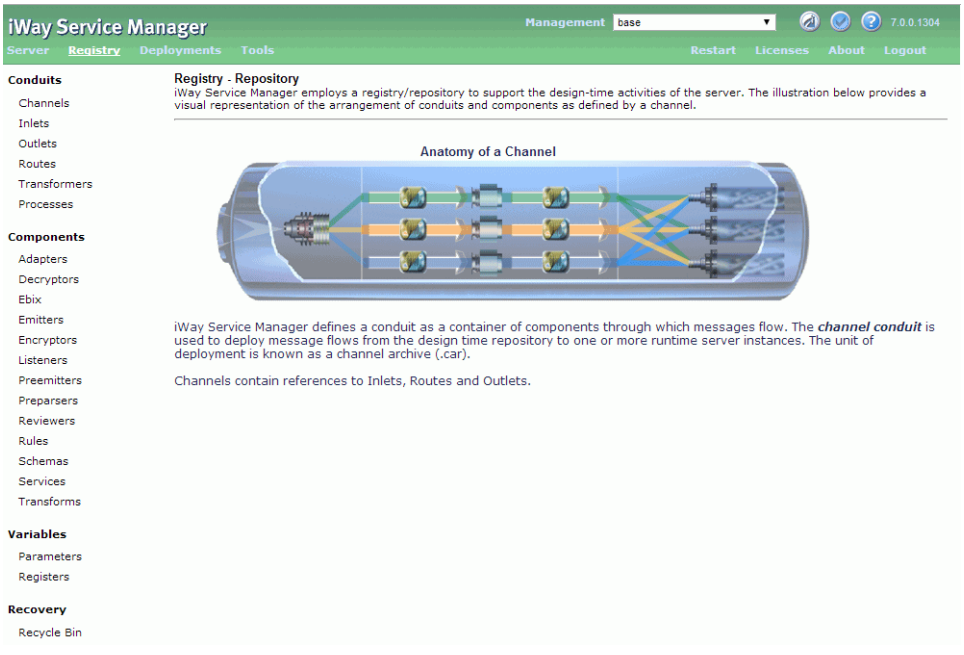
The following section describes how to configure the Telnet Listener.

Procedure: How to Configure the Telnet Listener

To configure the Telnet Listener:

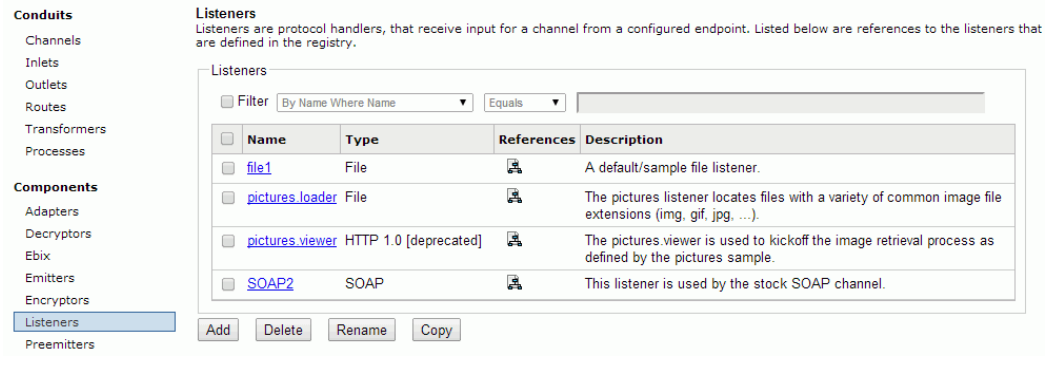
1. Click *Registry*.

   The Registry pane opens, as shown in the following image.



2. Click *Listeners* in the left panel.

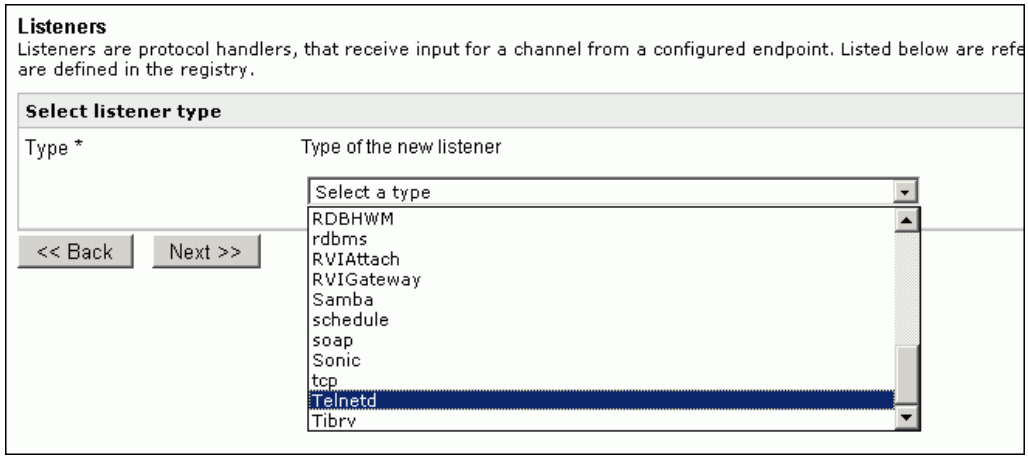
   The Listeners list pane opens, as shown in the following image.



3. Click the *Add* button at the bottom of the list.

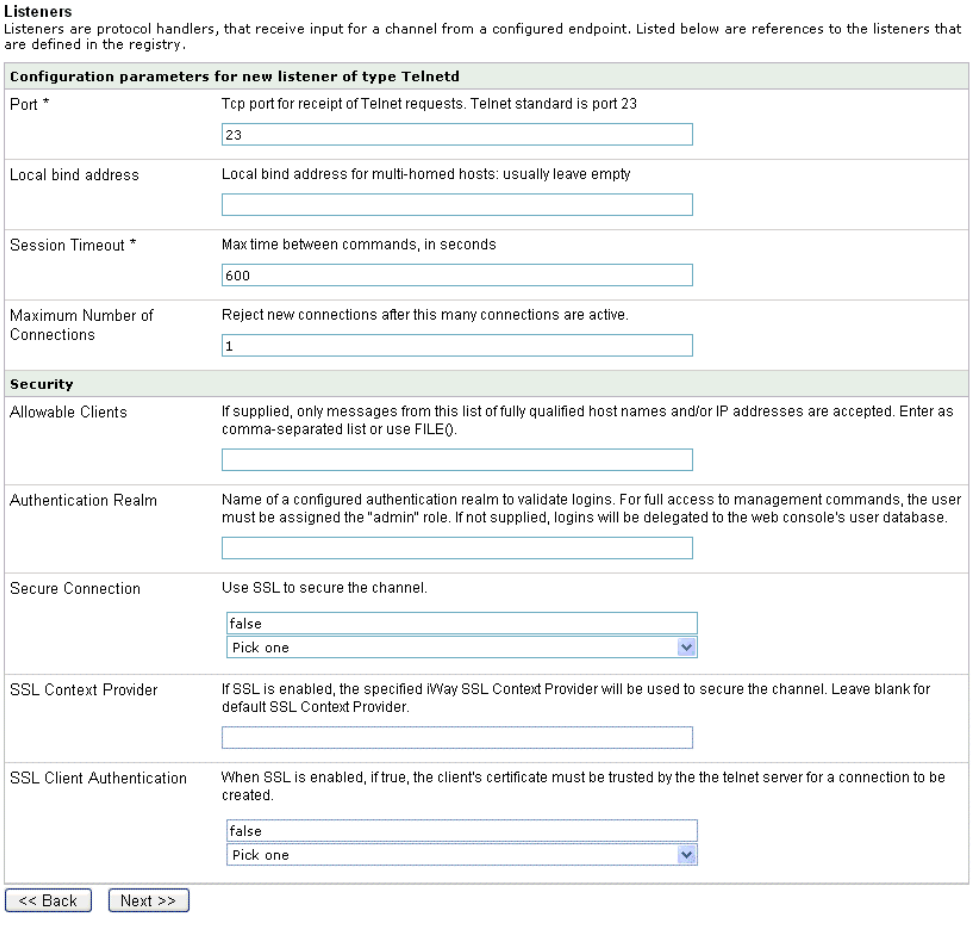
The Select listener type pane opens.

4. From the drop down list, select the *Telnetd* listener type, as shown in the following image.



5. Click *Next*.

The Configuration parameters for new listener of type Telnetd pane opens, as shown in the following image.



6. Leave the defaults for the listener properties, then select *Next*.

   A pane opens with a name and description field for the listener.

7. Give the listener a name and description and click *Finish*.

   This listener will then be added to an Inlet and then a Channel.

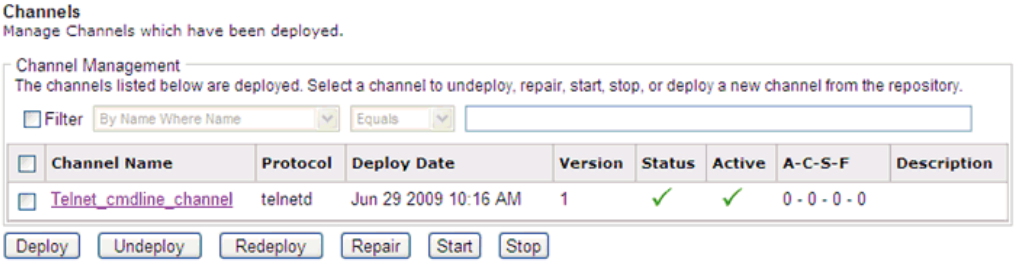
Reference: **Telnetd Listener Configuration Parameters**

Parameter	Description
Port *	The TCP port to receive Telnet requests. The default Telnet port is 23.
Local bind address	Local bind address for multi-homed hosts. This value is usually left blank.
Session Timeout*	Period in seconds of inactivity after which the connection will be timed out. The default value is 600.
Maximum Number of Connections	Rejects any new connections after the specified number of connections are active. The default value is 1.
Security	
Allowable Clients	If supplied, only messages from this list of fully qualified host names and/or IP addresses are accepted. Enter as comma-separated list or use FILE().
Authentication Realm	Name of a configured authentication realm to validate logins. For full access to management commands, the user must be assigned the admin role. If not supplied, logins will be delegated to the user database of the web console.
Secure Connection	Determines whether to use SSL to secure the channel.
SSL Context Provider	If SSL is enabled, the specified iWay SSL Context Provider will be used to secure the channel. Leave blank for default SSL Context Provider.
SSL Client Authentication	When SSL is enabled, if true, the client's certificate must be trusted by the Telnet server for a connection to be created. The permitted values are true and false.

Testing a Telnet Listener Channel

The following is a sample channel configuration where the Telnet listener is configured as an inlet.

For more information on configuring channels, see the *iWay Service Manager User's Guide*.



Once the channel is deployed, you can access the iSM command line console.

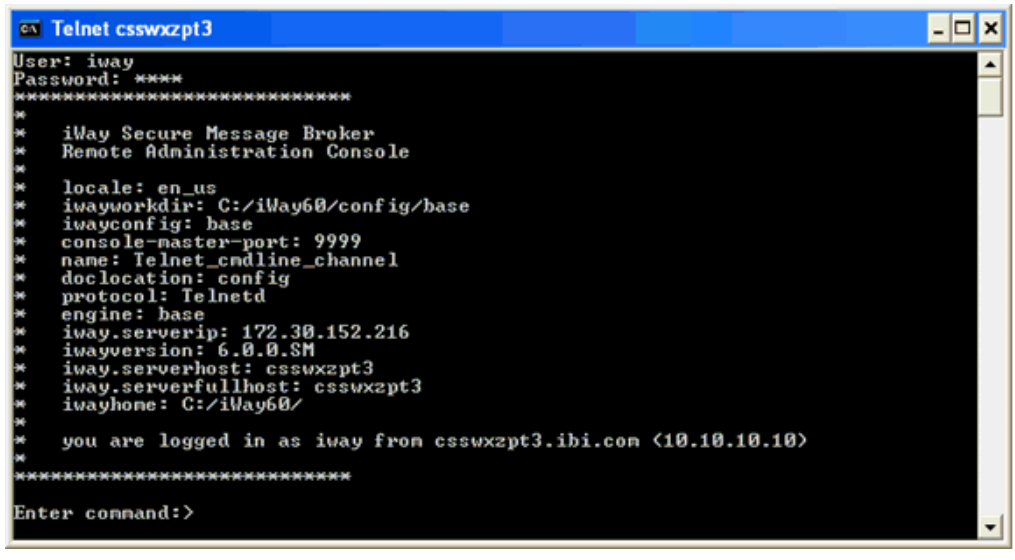
Connecting to iWay Service Manager Using a Telnet Client

In a use case scenario, if you need to test IFL functions or lookup help on iWay Service Manager remotely, the Telnet listener can be used.

1. Connect to iWay Service Manager using the command line. Enter the following command:

   ```
   telnet csswxzpt3
   ```

2. Enter a user name (for example, iway) and a password (for example, iway).



3. Once you are connected and logged in, you can now issue any command to monitor or control your iWay Service Manager instance.

Supported Telnet Commands

This section lists and describes the commands that can be issued from the command line console.

- **diagzip.** Creates a diagnostic information file for use by iWay Support. For example, you can enter the following command:

  ```
  diagzip c:\temp\Diag_from_base_7_01_2009
  ```

- **exits.** Displays loaded exits such as activity log and correlation manager.
- **func.** Displays the list of IFL functions, or the parameters of that function.
- **gc.** Runs the Java garbage collector.
- **help.** Displays help for commands. Use "help <name>..." for help on a specific command.
- **Info.** Displays the status of all channels deployed within the iSM config.
- **license.** Displays currently available iWay license codes.
- **line.** Draws a line on the console (helps mark the start of your trace).

- **memory.** Lists used and free memory [detail := analysis].
- **pools.** Lists resource pools.
- **providers.** Displays providers currently in use.
- **pull.** Loads information from another configuration or installation.
- **quit.** Exits the server.
- **refresh.** Reinitializes a channel.
- **run.** Runs a command file.
- **set.** Sets a parameter [help := list parms that can be set].
- **shell.** Attempts to run an operating system command.
- **sregs.** Displays special registers.
- **start.** Starts one or more channels.
- **stats.** Runs statistics on the current instance or listener.
- **stop.** Stops one or more channels.
- **threads.** Lists outstanding threads.
- **time.** Prints the GMT time on the console.
- **tool.** Runs a named tool such as 'testfuncs'.
- **version.** Display product version and all later versions of jars.

Note: Access to the iSM command line can also be utilized using portable handheld devices that support a Telnet client, for example, BlackBerry.

Telnet Scripting Example

The following is an example of automation or lights out operations that you can achieve after configuring a Telnet listener. A shell script is created containing the following command:

```
#!/bin/sh
host=localhost
port=9023
cmd="info"
( echo open ${host} ${port}
sleep 1
echo "iway"
sleep 1
echo "iway"
sleep 1
echo ${cmd}
sleep 1
echo quit ) | telnet > /home/jay/out.txt
echo " "
echo "* * * command output start * * *"
cat /home/jay/out.txt
echo "* * * command output end * * * *"
echo " "
```

There are other complicated ways of running Telnet on Linux than I/O redirection. For example, the command `expect` is designed to work with interactive commands.

The following example shows more of the script that can be parameterized as an information-only command which does not affect the behavior or configuration of the server.

```
* * * command output start * * *
telnet> Trying ::1...
Connected to localhost.
Escape character is '^]'.

User: iway
Password: ****
***************************
*
*   iWay Secure Message Broker
*   Remote Administration Console
*
*   protocol: Telnetd
*   engine: base
*   iway.serverip: 127.0.1.1
*   locale: en_us
*   iwayversion: 7.0
*   iway.serverhost: UbuntuVM
*   iwayworkdir: /iway/prog/7.0.36971/config/base
*   iwayconfig: base
*   console-master-port: 9999
*   iway.pid: 3392
*   iway.serverfullhost: UbuntuVM
*   iwayhome: /iway/prog/7.0.36971/
*   name: Telnet1
*   doclocation: config
*
*   you are logged in as iway from localhost (0:0:0:0:0:0:0:1)
*
***************************
Enter command:>info
                              completed   failed   active   workers   free
SOAP1
   http          -- active --       0        0        0        3        3
   file          -- active --       0        0        0        3        3
Telnet1          -- active --       0        0        1        1        0
Enter command:>quit
goodbye!
* * * command output end * * *
*
```

Chapter 5

iWay Security Extension

This section describes how to configure the iWay Security extension (Security Developers Tools) using iWay Service Manager.

In this chapter:

iWay Security Extension Overview

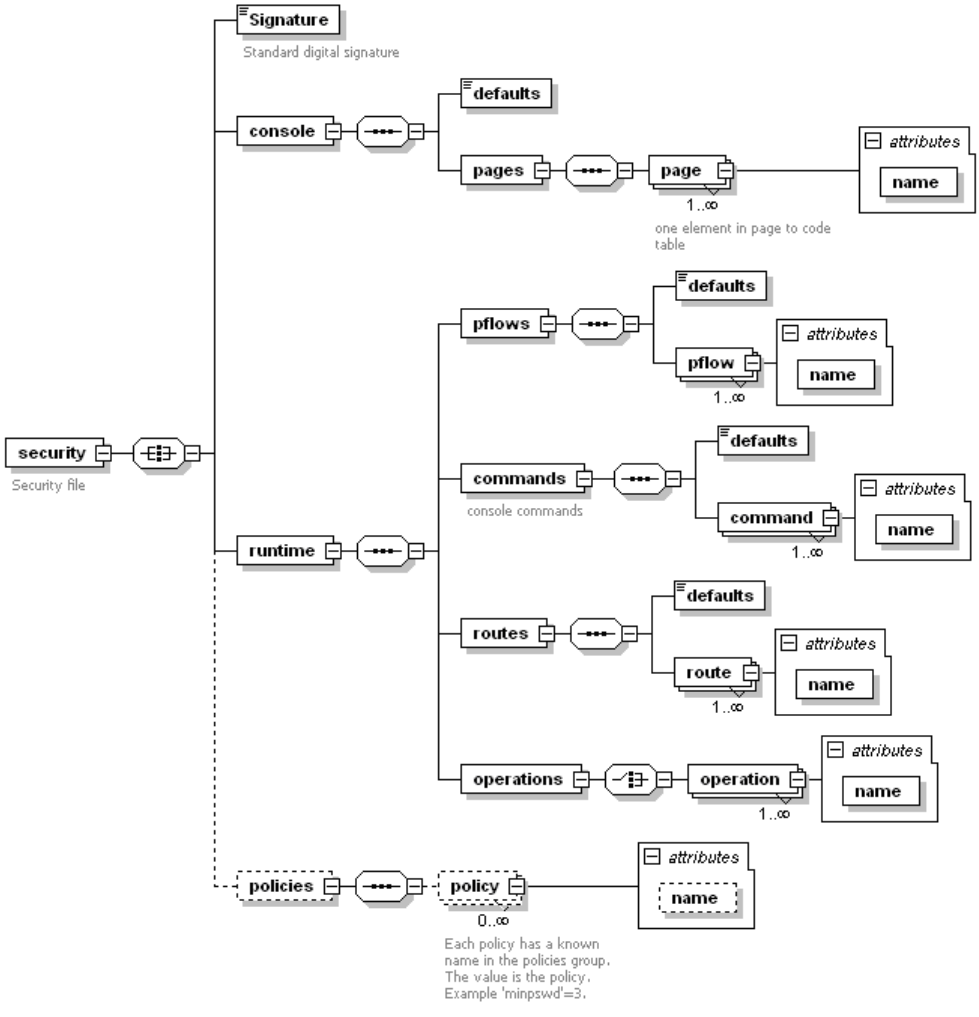
Security policies can be used to secure the server during run time. For example, if a security policy, such as dictionary and process flow signing is set when a runtime application is distributed, the server requires that the dictionary and any process flows to be run are signed using XML Digital Signatures. At startup, if the dictionary is not signed, the server will not start. An appropriate error message will be issued. Once running, any process flows to be run must be validly signed. Any attempt to load and run an unsigned or modified flow will be rejected. This protection applies to all channels. In addition, the iWay Service Manager console is disabled, as required. Application-specific consoles work through the standard HTTP channels, and are of course available.

Management of the dictionary signature is automatic. A validly signed dictionary must be distributed to customers. Doing this simply means taking the dictionary to be run from the development system. No further preparation or action is necessary.

Process flows need to be signed individually before they are packaged for distribution. The server manages signing keys and it considers two types of files: dictionaries and process flows. Each type uses a unique key pair. The server automatically selects the proper key for signing and validating configuration files based upon the type of file.

Prerequisites

The set of ACLs for the system are its policies. Policies are stored in a policy file in the config area. The candidate schema layout shown below is for discussion and clarification only.



The file stores the policies under which the server operates. It is not clear at this time whether the policies for run time differ by configuration. Should it be so decided, the policy layout will be changed. Each section has a default area in which the access role for any object of that section that is not named is stored.

Policies

Policies are values that control server actions. An example is minpswd, which controls the minimum length of passwords. Policies are carried in the file, and can be checked as needed in the server. The following table lists a selection of policies:

Name	Use	Comments
minpswd	Minimum password length	Does not apply to passwords recorded for other systems. For example, FTP.
signpflow	Should all pflows be signed	Move from license

Installing the iWay Security Extension

To install the iWay Security extension, you must add the Security Developers Tools extension to your iWay Service Manager instance during the iWay Service Manager installation. For more information on installing iWay Service Manager, see the *iWay Installation and Configuration Guide*.

Security Extensions - XDDevKit

The command SignTree is available as a part of the XDDevKit(Security) extension. This is a utility program for signing and verifying files.

Procedure: How to Signing a Process Flow Before Distribution to a Run-time Server

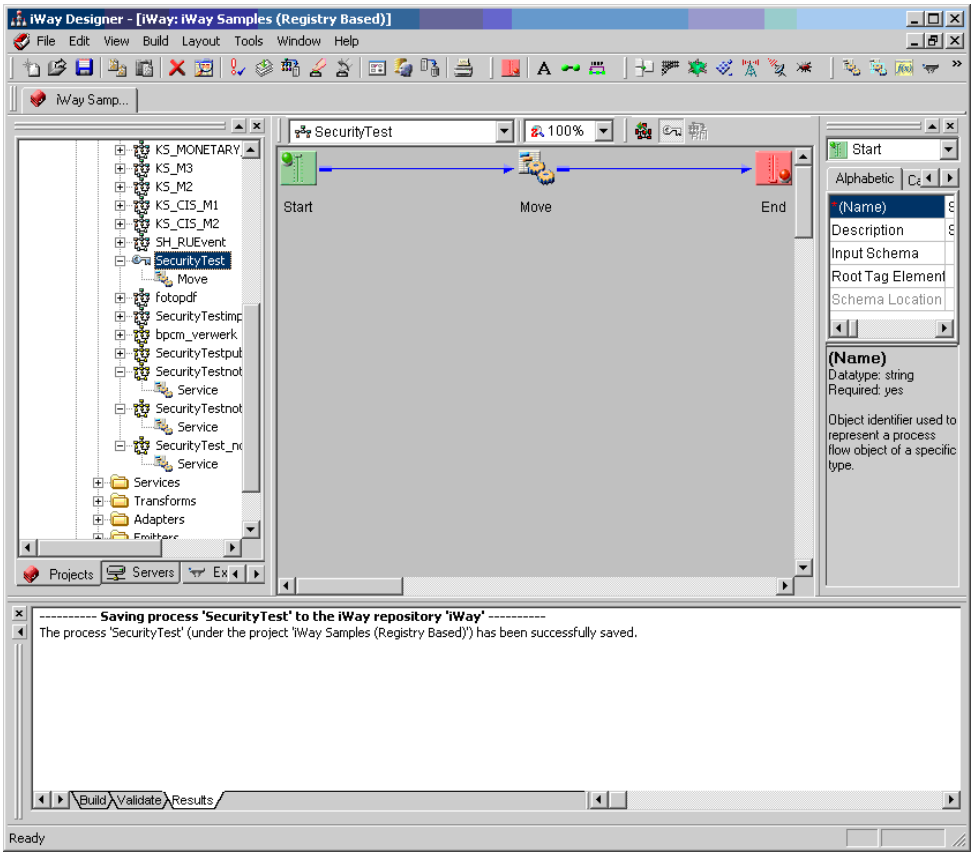
As a simple example, consider a case where the process flow is signed before the vendor distributes it to a customer. To test this case, use two different iWay servers for testing (for example, iWay Server A and iWay Server B).

1. Enable a policy for signing process flows on iWay Server A.

   The command is: `Set policy signpflow true`

   To enable a policy for signing process flows inside a specific configuration, for example, base, the command is: `Set policy base/signpflow true`

2. Create a simple process flow on iWay Server B, for example, *move*, which consists of a Move service.



3. Sign the process flows before distribution on iWay Server B.

```
Tool signtree -s SecurityTest.xml
```

where:

SecurityTest.xml

Is the process flow to be distributed.

4. Verify of the process flow on iWay Server A where distribution is done internally, provided the policy is enabled, as described in step 1. Import the process flow SecurityTest.xml on the run time iWay Server A. Name the process flow *SecurityTest*.

The process flow is successfully imported onto iWay Server A as the signature is set.

5. On iWay Server A, create a channel containing the process flow *SecurityTest* inside a route. Build and deploy the channel. Perform a test run on the channel to verify the move process.

Chapter 6

iWay Compatibility Extension

This section provides an overview of the iWay Compatibility extension.

In this chapter:

iWay Compatibility Extension Overview

In iWay Service Manager (iSM), the following set of obsolete services (agents) are now included in the iwxcompat iSM package for backward compatibility purposes.

- XDFieldAgent
- XDMarkAttachAgent
- XDStandardAgent

Note: For more information and a complete description of these services, see the *iWay Service Manager User's Guide, (Appendix C, iWay Services)*.

If there are any applications that have been upgraded from iSM Version 5.5 or earlier to iSM Version 6.1.6, then these applications may still be referring to these obsolete services. As a result, you can install the iWay Compatibility extension, which will add the ixcompat iSM package to the following directory:

iwayhome\etc\manager\extensions

where:

iwayhome

Is the location where iWay Service Manager is installed.

Installing the iWay Compatibility Extension

To install the iWay Compatibility extension, you must add the 5.x Compatibility extension to your iWay Service Manager instance during the iWay Service Manager installation. For more information on installing iWay Service Manager, see the *iWay Installation and Configuration Guide*.

Chapter 7

iWay Scheduler Extension

This section describes how to configure the iWay Scheduler extension, which is used to schedule iWay Service Manager (iSM) tasks to run at specific time(s) during the hour, day, or month.

In this chapter:

Installing the iWay Scheduler Extension

This section provides prerequisite information and describes how to install the iWay Scheduler extension.

Prerequisites

iWay Service Manager (iSM) Version 7.0 must be installed and configured correctly.

For more information on installing iSM, see the *iWay Installation and Configuration User's Guide*.

Installation Overview

Perform the following steps to install the iWay Scheduler extension:

1. Copy the iwxscheduler.jar file to the following directory:

   `<iWay_Home>\etc\manager\extensions`

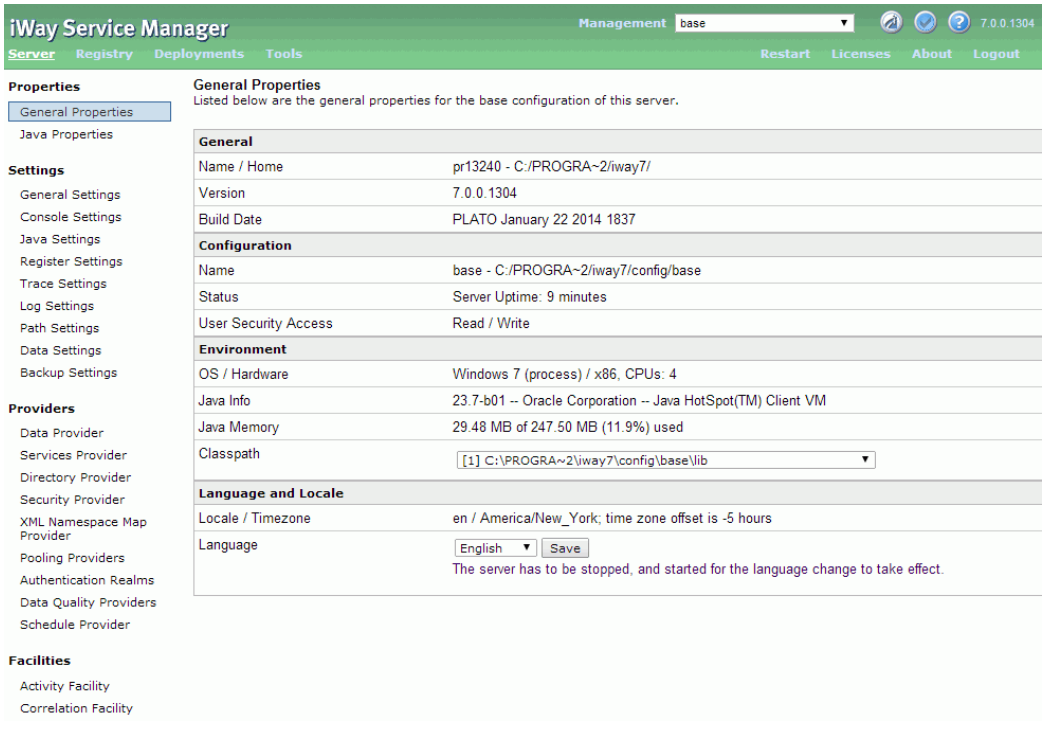
2. After copying the iwxscheduler.jar file into the extensions directory, iSM must be restarted especially if iSM was running prior to copying the .jar file.
3. Add a task to the schedule of iSM. This can be done using one of the following methods:
   - Using the iWay Service Manager Administration Console (web console interface).
   - Entering the task commands directly into the command line console of iWay Service Manager.

Configuring the iWay Service Manager Scheduler

This section provides an overview of using the iWay Service Manager Administration Console to schedule and configure Service Manager tasks.

iWay Service Manager Administration Console

The following image shows the default iWay Service Manager Administration Console open to the General Properties pane.



The Administrative Console is divided into the following sections:

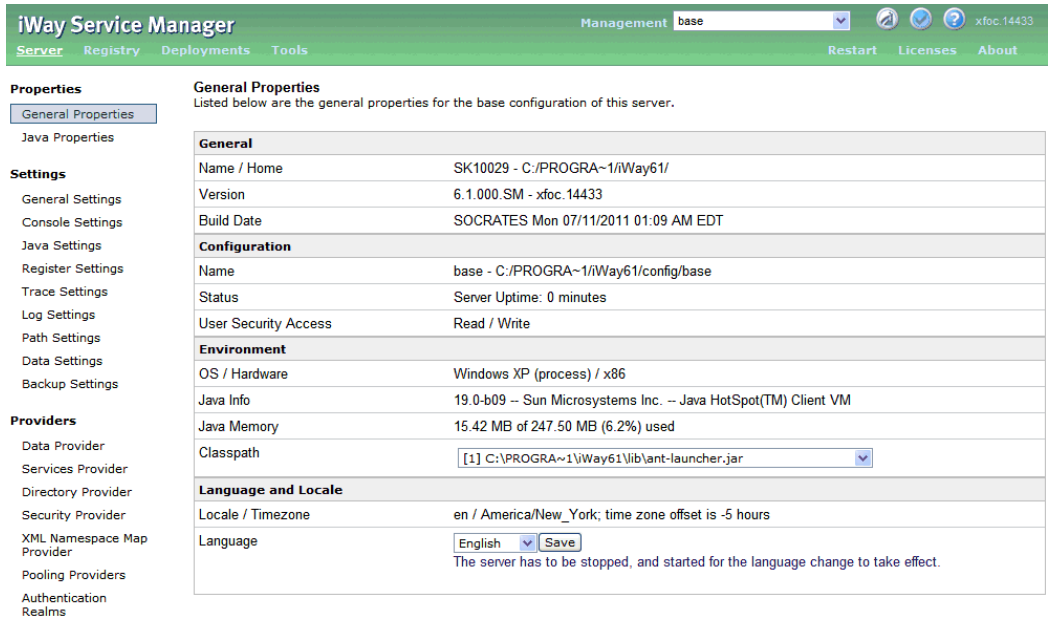
- Navigation menu (top pane of the console).
- Configuration menu (left pane of the console under the Navigation menu).
- Pane that displays information related to the option selected in the Configuration menu (right pane of the console under the Navigation menu).

If the scheduler extension of the Service Manager has been successfully installed, the Schedule Provider hyperlink is found in the Configuration menu.

iWay Service Manager Schedule Provider

This section describes how to access and use the Schedule Provider in iWay Service Manager.

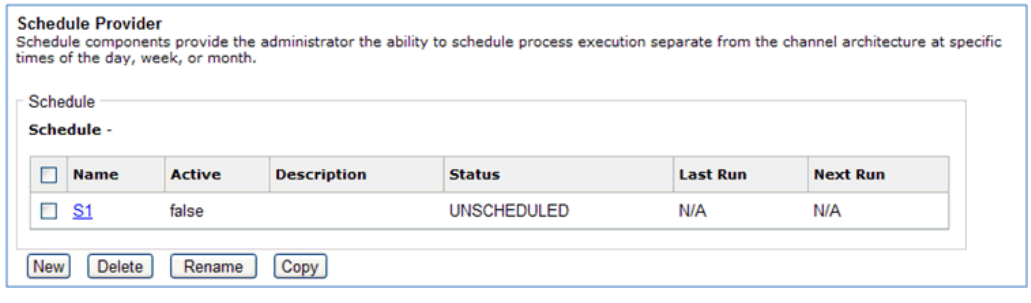
Accessing iWay Service Manager Schedule Provider



To access the Schedule Provider:

1. Click *Server* from the navigation menu in the top pane.
2. Select *Schedule Provider* from the configuration menu in the left pane.

The following image shows the Schedule Provider listing with several tasks scheduled.

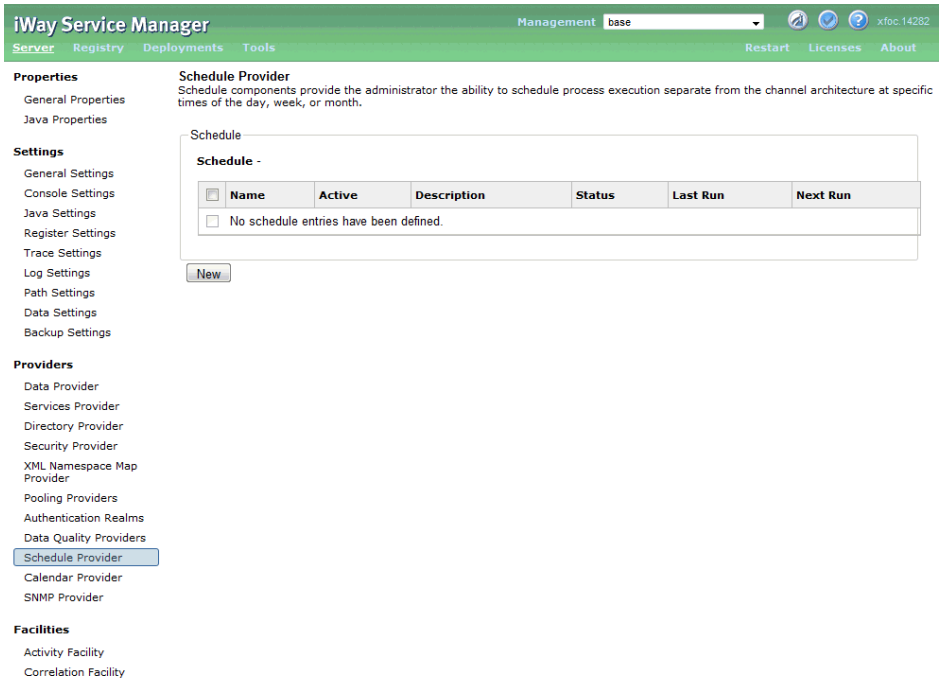


The Schedule Provider displays a list of already configured tasks. If this is the first time that the administrator is using the Schedule Provider or there are no tasks configured for this Managed server, then this list will be blank.

The following table shows the columns that a list of entries would be categorized by.

Column	Description
Name	Name associated with the task.
Active	Flag indicating whether or not the configured task is active. If True, the task is active and will be scheduled whenever iWay Service Manager is recycled. If False, the task is inactive and will not be scheduled whenever iWay Service Manager is recycled.
Description	Description of the task; this is a freeform field that is entered in the Schedule Configuration page, containing a description of what the task does.
Status	Last known status reported by the task.
Last Run	Last time that the task was run by the Service Manager.
Next Run	Next time that the task will be run by the Service Manager.

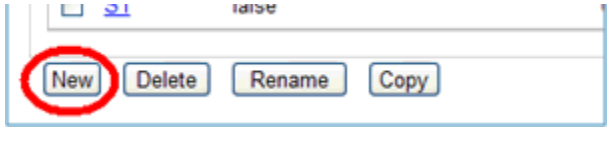
Procedure: How to Schedule a Listener to Start



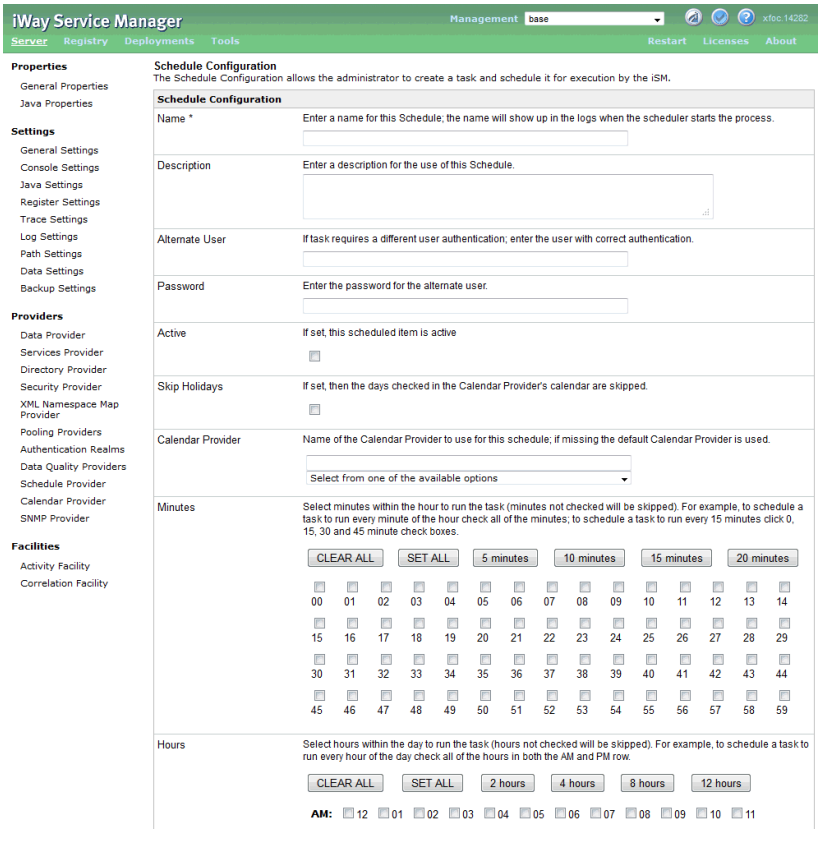
To create a new Schedule Provider:

1. Click *Server* from the navigation menu in the top pane.
2. Select *Schedule Provider* from the configuration menu in the left pane.

   A table appears that lists any existing Scheduled tasks and a short description for each.
3. Click *New* as shown in the following image.



The Schedule Configuration page opens, as shown in the following image.



4. Provide the appropriate property values for the task, as defined in the following table.

   Note: An asterisk indicates a required property.

Property	Definition
Name*	Name to associate to this task. Task names are case sensitive and can not contain punctuation or other special characters.
Description	A brief description of what this task does. This description should be descriptive enough to identify the task. This description is displayed on the Schedule Provider list of scheduled tasks.

Property	Definition
Alternate User	If this task is being executed with credentials of a different user, enter the User ID of that user. An alternate user must be a valid user defined on this iSM server. **Note:** The Alternate user can contain iFL commands (iSM functions like _SREG, _PROPERTY, and so on) that will be evaluated when the schedule provider is called.
Password	If this task is being executed with credentials from a different user, enter the password of that user. **Note:** The Password can contain iFL commands (iSM functions like _SREG, _PROPERTY, and so on) that will be evaluated when the schedule provider is called.
Active	If set to true, this task will be scheduled each and every time the Service Manager is recycled.

Property	Definition
Minutes	Checks the starting minute(s) of the hour to start the task at. Six short cut buttons are supplied to help in setting the minutes to start execution: ❏ Clicking the *CLEAR ALL* button sets all the minute check boxes off. ❏ Clicking the *SET ALL* button sets all the minute check boxes on. ❏ Clicking the *5 minutes* button starts at 0 and sets each 5 minute check box on. For example, 0, 5, 10, 15, and so on. ❏ Clicking the *10 minutes* button starts at 0 and sets each 10 minute check box on. For example, 0, 10, 20, 30, and so on. ❏ Clicking the *15 minutes* button starts at 0 and sets each 15 minute check box on. For example, 0, 15, 30, 45, and so on. ❏ Clicking the *20 minutes* button starts at 0 and sets each 20 minute check box on. For example, 0, 20, 40, 60, and so on.

Property	Definition
Hours	Checks the starting Hour(s) of the day to start the task at. Six shortcut buttons are supplied to help in setting the minutes to start execution. ❏ Clicking the *CLEAR ALL* button sets all the hour check boxes off. ❏ Clicking the *SET ALL* button sets all the hour check boxes on. ❏ Clicking the *2 hour* button starts at 12am and sets each 2 hour check box on. For example, 12 am, 2 am, 4 am, and so on. ❏ Clicking the *4 hour* button starts at 12am and sets each 4 hour check box on. For example, 12am, 4 am, 8 am, and so on. ❏ Clicking the *8 hour* button starts at 12am and sets each 8 hour check box on. For example, 12am, 8am, 4pm, and so on. ❏ Clicking the *12 hour* button starts at 12am and sets each 12 hour check box on. For example, 12am, 12pm, and so on.
Month	Checks the month that the task should run on. Two shortcut buttons are also provided: ❏ Clicking the *CLEAR ALL* button un-checks all of the Month check boxes. ❏ Clicking the *SET ALL* button checks all of the Month check boxes.

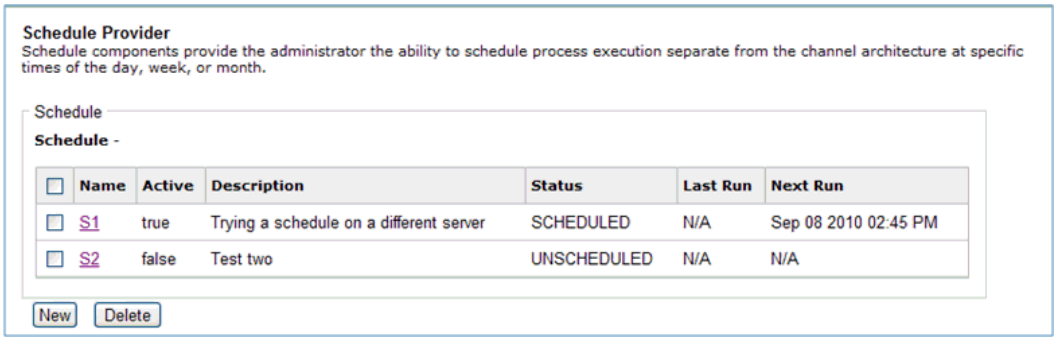
Property	Definition
Weekday	A number between 0 (Sunday) and 6 (Saturday) or 1 and 7 depending on the default Schedule Settings that represents what day of the week that the task should run on. In addition to the numerical representation, the weekday names or abbreviations can also be used: ❑ Clicking the *CLEAR ALL* button un-checks all of the Weekday check boxes. ❑ Clicking the *SET ALL* button checks all of the Weekday check boxes. ❑ Clicking the *Set Weekdays* button checks the Monday (Mon) through Friday (Fri) check boxes. ❑ Clicking the *Set Weekend* button checks the Saturday (Sat) and Sunday (Sun) check boxes.

Property	Definition
Day of Month	Checks the day of the month that the task should run on. In addition to the days, the following special values can also be checked: ❑ **@FMOM**: First Monday of the month, can be abbreviated as @FM. ❑ **@LFOM**: Last Friday of the month, can be abbreviated as @LF. ❑ **@LBDOM:** Last business day of the month. This value will return the last calendar workday (Monday through Friday) of the month. This value can be abbreviated as @LBD. ❑ **@LDOM**: Last day of the month; can be abbreviated as @LD. Clicking the CLEAR ALL button un-checks all of the Day of Month check boxes but does not clear the Special Value check boxes. Clicking the SET ALL button checks all of the Day of Month check boxes but does not check the Special Value check boxes.
Command*	The command that the task will execute when the scheduled time comes. Any Service Manager command will be executed. Some Service Manager commands make more sense than others to schedule as tasks.
Duration Timer	Length of time that the task will run prior to the Dependent Command. The format of duration [in seconds] is in the form [xxh][xxm]xx[s], as shown in the following example: `04h30m45` This creates a duration of 4 hours, 30 minutes, and 45 seconds.

Property	Definition
Dependent Command	Command to be executed after the Duration Timer of the task has expired. This can be the name of another configured scheduler task or be a separate iSM command (for example, stop listener, start listener, and so on). This task is entered into the Service Manager schedule as name.dependent, where name is the name assigned to this task. The dependent task will not be rescheduled after executing, but rather scheduled again only after the command of the primary task has started.

5. Click *Add*.

   The Schedule Provider page is displayed, as shown in the following image.



   The task BeginningOfMonth is added to the schedule list of iWay Service Manager and will be scheduled when iSM is recycled and a new instance is started.

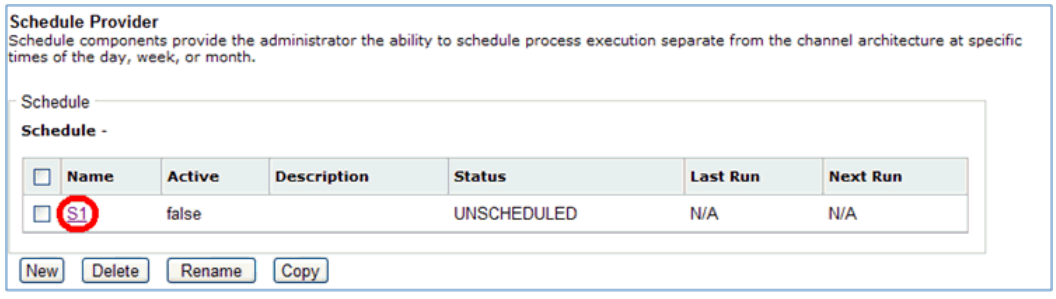
Procedure: How to Update a Task in Schedule Provider

From time to time it can become necessary to update the schedule of an existing task. To update a task in Schedule Provider:

1. Click *Server* from the navigation menu in the top pane.
2. Select *Schedule Provider* from the configuration menu in the left pane.

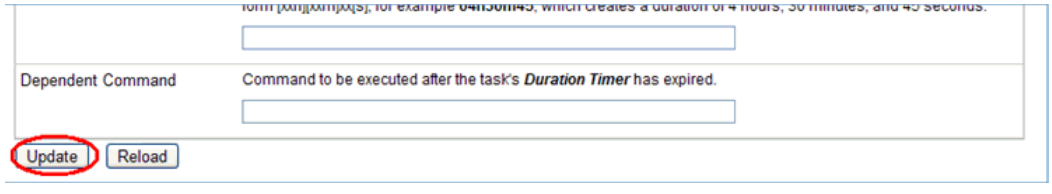
   A table appears that lists any existing Scheduled tasks and a short description for each.

3. Click a scheduled task you wish to update, for example, S1, as shown in the following image.



The Schedule Configuration page is displayed with the Schedule parameters for that task. Make whatever change is necessary to the schedule.

4. Click *Update*, as shown in the following image.



The Schedule Provider page is displayed and the task S1 has been updated in the schedule.

Changes that you have made to the Active and Description fields of the task will be reflected in the list, but schedule changes will not take place until either the Service Manager is recycled or a new instance is started.

Procedure: How to Reload a Task in Schedule Provider

Reloading a task in the schedule of iSM should be done when:

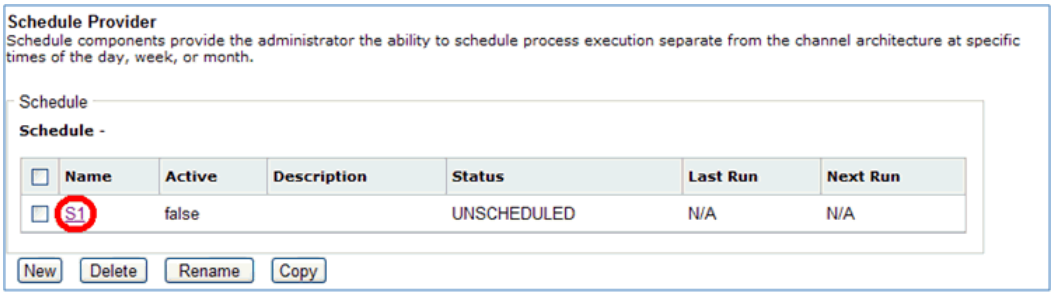
- A new task has been added to the schedule.
- An existing schedule of a task has been updated.

To reload a task in Schedule Provider:

1. Click *Server* from the navigation menu in the top pane.
2. Select *Schedule Provider* from the configuration menu in the left pane.

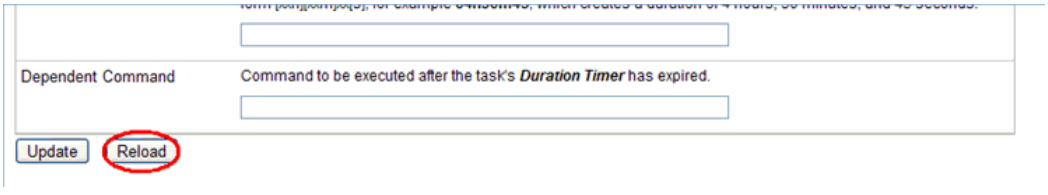
   A table appears that lists any existing Scheduled tasks and a short description for each.

3. Click a scheduled task you wish to update, for example, S1, as shown in the following image.



The Schedule Configuration page is displayed with the Schedule parameters for that task.

4. Click *Reload* as shown in the following image.



The Schedule Configuration page is displayed. If the task is successfully scheduled within this Service Manager instance, you will see the following message.

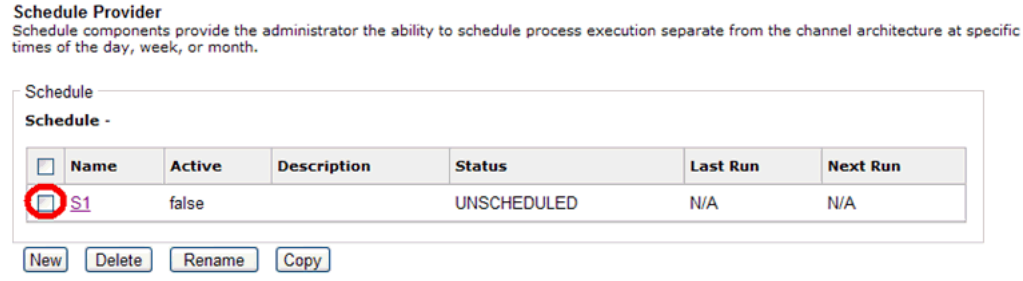
5. Click *Schedule Provider* on the configuration menu in the left pane to exit.

Procedure: How to Copy a Task in Schedule Provider

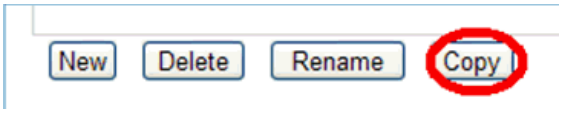
To copy a task in Schedule Provider:

1. Click *Server* from the navigation menu from the top pane.
2. Select *Schedule Provider* from the configuration menu in the left pane.

3. Select the check box next to the name of the Schedule Provider you wish to copy, for example, *S1*, as shown in the following image.



4. Click *Copy*, as shown below.

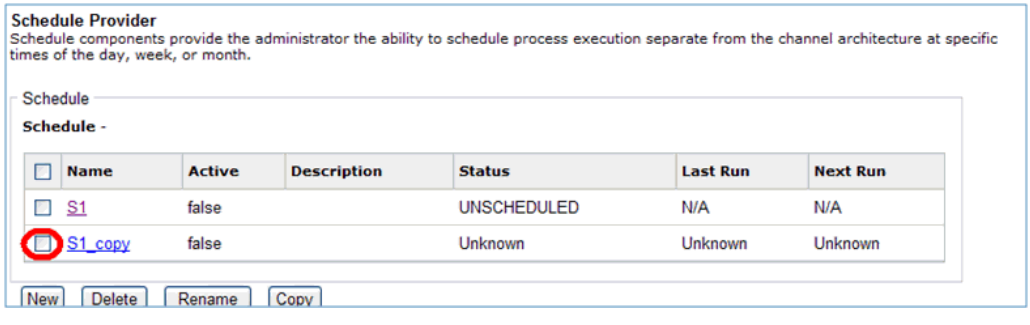


All schedules that are marked with the check are copied to a new Schedule Provider with the name pattern of oldProviderName_copy.

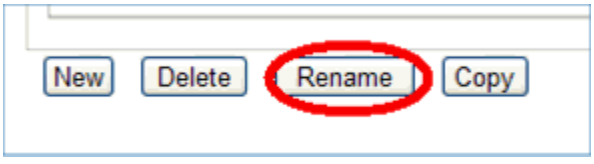
Procedure: How to Rename a Task in Schedule Provider

To rename a task in schedule provider:

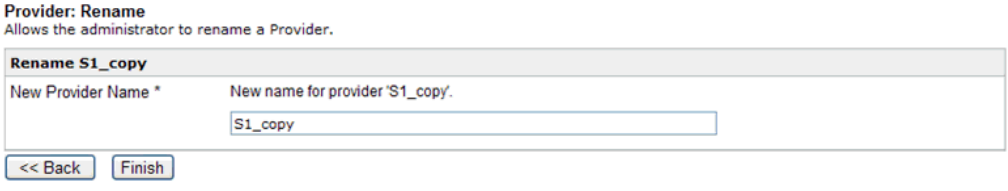
1. Click *Server* from the navigation menu from the top pane.
2. Select *Schedule Provider* from the configuration menu in the left pane.
3. Select the check box next to the name of the Schedule Provider you wish to rename, for example, *S1_copy*, as shown in the following image.



4. Click *Rename*, as shown below.



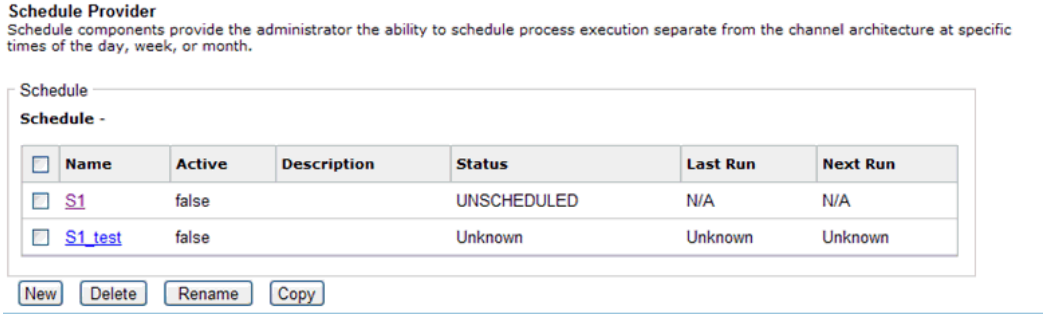
Clicking on the *Rename* button displays the Provider Rename frame with the current name of the provider displayed in the New Provider Name field, as shown in the following image.



To abort renaming the provider, click *Back*.

5. Enter a new name for the provider in the field provided and click *Finish*.

The provider will be renamed and the Schedule Provider screen will be displayed, as shown below.



Using iWay Service Manager Command Line Console

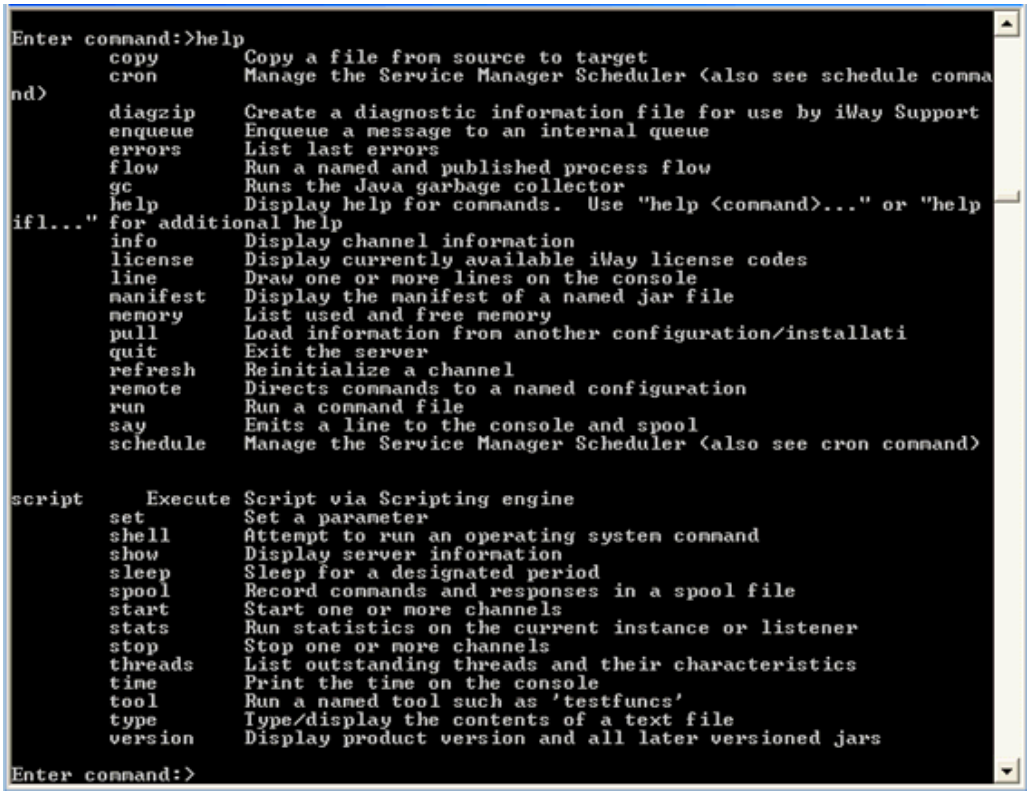
This section provides an overview of the Service Manager cron and schedule command line console.

Command Line Basics

The iSM command line console is available directly from the command line of the server when the Service Manager is started as a standalone Java application. If the Service Manager is running as a background task (either as a Windows Started Task or as a UNIX daemon), a telnet connection to the Service Manager can be used.

Command Line Help

Use the help system of the Command Line to see if the Service Manager Schedule is installed. Type *help* after the Enter command:> prompt. If the scheduler has been properly installed, you should see cron and schedule in the listing of commands and descriptions that follows.

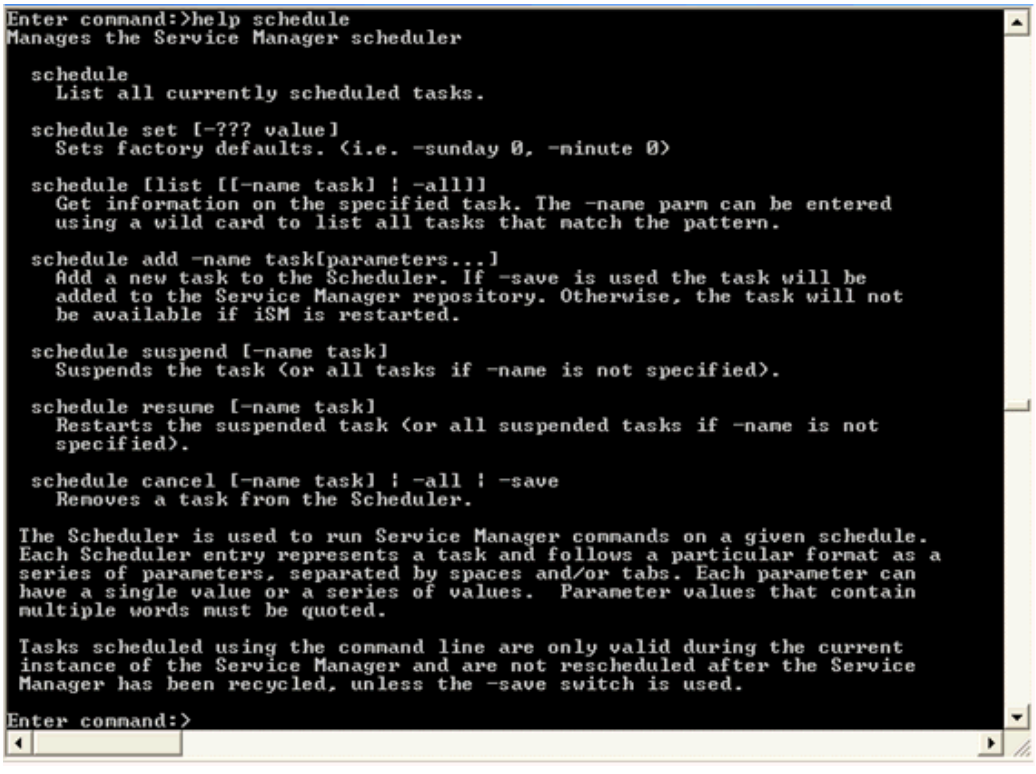


The schedule and cron commands also provide help for the user. To access either the schedule or cron help type either of the following commands after the command prompt:

```
help schedule
```

```
help cron
```

The following image shows the help schedule command screen:



The following image shows the cron help command screen:

iWay Service Manager Cron Command Console

The cron command allows you to add, suspend, resume or cancel tasks during the current instance of the Service Manager. Unless otherwise noted in the function, tasks that are scheduled using the command console will not be saved in the repository of Service Manager nor will they be carried over from one instance of the Service Manager to the next when recycled.

Note: The preferred way to Schedule a recurring task is to use the Schedule Provider found in the iWay Service Manager Administration Console.

The cron command console was modeled after the UNIX cron and crontab entry. Most UNIX operating systems have a cron utility that allows tasks to be automatically run in the background at regular intervals by a cron daemon. These tasks are often termed as cron jobs in UNIX. To manage those cron jobs, a file called crontab (short for CRON TABle) is used. This file contains one or more lines, each line a cron job entry to be run at specified times based on the parameters of the line.

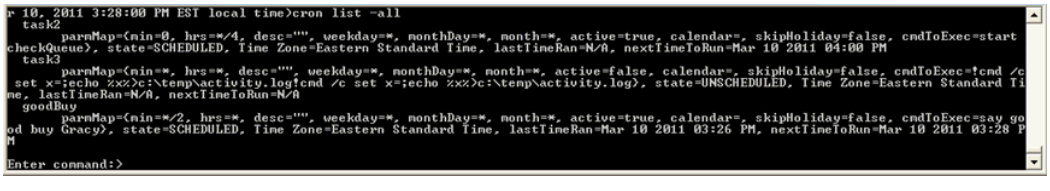
Listing a Task

Entering the command cron or cron list produces a listing of all currently scheduled iWay Service Manager tasks. To list one or all of the currently scheduled Service Manager tasks, the general command format is:

```
cron [list [-name task] | [-all]]
```

Parameter	Description
`-name`	Is the name or partial name of the task. The name can contain only part of a task name, for example, *t* to list all tasks that start with the letter *t*, *te* to list all tasks that start with the letters *te*, and so on.
`-all`	When the list command is issued without the all parameter, only active tasks are listed. Tasks that are suspended are not displayed. Using the -all parameter lists both active and suspended tasks.

The following image shows an example of the schedule list output.



Entering the command cron list -name <value> (in the following example case 'g*') produces a list of tasks that start with the value *g*. The following image shows an example of the schedule list -name function output.

Adding a Task

Unlike the schedule command of iSM, the cron command is very dependent on parameter positioning. Because the cron command emulates a crontab file entry, the first five fields following the cron add command specifies the day, date, and time followed by the command to be run at that interval. The table below shows the order (from top to bottom) and value of the first five fields that follow the cron add command.

Order	Value
Minutes	* or (0 - 59)
Hours	* or (0 - 23)
dayOfMonth	* or (1 - 31)
Month	* or (1 - 12)
Weekday	* or (0 - 6)
Command	Command to be executed.

Note:

- Day of week value of 0 indicates Sunday.
- An asterisk (*) in the value field above means all legal values as in braces for that column.
- The value column can have a *, a single value, or a list of values separated by commas. A value can be either a number in the ranges shown above or two numbers in the range separated by a hyphen. For example, 1-4 covers the values 1 through 4 inclusive of the numbers 1 and 4, 1-4,7-11 covers the values 1 through 4 and the values 7 through 11.
- Repeat pattern like (called a step) is supported. For example, /2 for every 2 minutes, /10 for every 10 minutes.

- The specification of days can be made in two fields: Day of Month and Weekday. If both are specified in an entry, they are cumulative meaning both of the entries will be executed.

Entering the command cron add will add a new task to the Scheduler. The general command format is:

```
cron add minutes hours dayOfMonth month weekday command -name taskName
   [[-active][-description ...][-user ... -password ...]
[-save]]
```

If -active flag is included in the command line, the task is immediately scheduled to run at the next scheduled time.

Parameters	Description
Minutes	A numeric value between 0 and 59 representing when within the hour the task should run. Optional * = all minutes in the hour (same as 0-59).
Hours	A numeric value between 0 and 23 (where 0=12am, 23=11pm) representing when within the day the task should run. Optional * = all hours in the day (same as 0-23).
dayOfMonth	A numeric value between 1 and 31 representing what day in the month the task should run. Optional * = all days in the month).
Month	A numeric value between 1 and 12 (where 1=January, 12=December) representing what month the task should run. Alternately the name of the months may also be used, for example: January[jan], February[feb], March[mar], and so on. Optional * = all months in the year (same as 1-12).

Parameters	Description
Weekday	A numeric value representing what day of the week the task should run, Sunday (0) through Saturday (6). Alternately the name of the weekday may also be used, for example: Sunday[sun], Monday[mon], Tuesday[tue], Wednesday[wed], and so on. Optional * = all days of the week (same as 0-6).
Command	The iSM command, for example, start listenerName, stop listenerName, ... that the task will execute. (For more information, see the *iWay Service Manager User's Guide*).
-name	Enter a unique name to associate to the task. This name will be used when looking up the task later.
-description (optional)	Enter a brief description of the task.
-active (optional)	If set to true, the task is scheduled to run immediately. If the value is false, then the task is not scheduled to run. If it is missing or set to false (the default), the status of the task is set to UNSCHEDULED.
-user (optional)	If the task must be run with an alternate user ID, enter the ID.
-password (optional)	If the task must be run with an alternate user ID, enter the password of the alternate user. (Required if -user is specified, ignored if -user is not included.)
-save (optional)	Save the added scheduled task in the Schedule repository. This ensures that the schedule and any changes are persisted when Service Manager is recycled.
-skipHoliday (optional)	If 'Skip Holidays' flag set ('true'), then the days checked in the Schedule's calendar are skipped.

For example, if you want to execute a shell command to clear the error_log file located in the directory c:wwapachelogs every day at midnight, issue the following:

```
cron add 0 0 * * * "!cmd /c set nada=;echo %nada% > c:logserror_log" -name
task1
```

To start up a Service Manager listener called MonthEnd on the last business day of the month (will not start the listener up on Saturday or Sunday) at 8am:

```
cron add 0 8 @LBDOM * * "start MonthEnd" -name task2
```

To start up a Service Manager listener called MonthEnd on the last day of the month at 8am:

```
cron add 0 8 @LD * * "start MonthEnd" -name task3
```

Note:

- Time parameters not entered (for example. -month, -day, -weekday, etc.) will default to the asterisk ('*') operator (meaning all possible values for that field).
- There are several ways of specifying multiple date/time values in a field:
  - The comma (,) operator specifies a list of values, for example: 1, 3, 4, 7, 8, and so on.
  - The dash (-) operator specifies a range of values, for example, 1-6 which is equivalent to 1, 2, 3, 4, 5, 6.
  - The values may also be combined as well, for example: 1-4, 6, 8-10 is equivalent to 1, 2, 3, 4, 6, 8, 9, 10.
- The asterisk (*) operator specifies all possible values for a field. For example, an asterisk in the hour time field would be equivalent to every hour (subject to matching other specified fields).
- The step operator (/) is valid only for the minutes and hours fields, which can be used to skip a given number of minutes or hours. For example, */3 in the minute time field is equivalent to 0, 3, 6, 9, and so on. While (*) specifies every minute, the */3 means only those minutes divisible by 3. When the step operator does not have an asterisk value (*) preceding the (/) specifier, this means starting now executes again in n minutes or hours rather than every n minutes or hours. For example, /2 in the minutes field indicates execute again in 2 minutes, /5 executes again in 5 minutes and so on.
- Commands and descriptions that consist of multiple words must be enclosed in double quotes ("). For example, start checkQueue or This task is run only at year's end.

Canceling a Task

The cron cancel command is used to remove a task or all tasks from the Scheduler of iSM. Any tasks that were canceled (but not saved) will resume as scheduled only when iSM Service Manager restarts.

To cancel a task from the schedule of iWay Service Manager, the general command format is:

```
cron cancel -name task [-save] | -all]
```

Parameter	Description
`-name`	Name of the task to cancel. The named task will be canceled and removed from the Service Managers schedule.
`-save`	The named task is removed from the Schedule repository of Service Manager and will not be available when Service Manager is restarted.
`-all`	All tasks currently in the Scheduler of Service Manager are immediately removed.

Suspending a Task

The cron suspend command is used to suspend the next scheduling a task (or all tasks) in the Service Manager's Scheduler.

Note: Any tasks that were currently executing will complete execution but will not be rescheduled. The suspend command only prevents the task from being scheduled in the future.

To suspend a task within the schedule of iSM, the general command format is:

```
cron suspend [-name task]
```

Parameter	Description
-name	Optional name of the task to suspend. The named task scheduling will be suspended; the task will remain on the Schedule list of iSM with a status of SUSPENDED. To restart the task at its next regularly scheduled time use the cron resume command. 1. If -name is not supplied ALL SCHEDULED tasks will be suspended. 2. If the suspend command is issued while a task is running, the running task is not interrupted but will complete normally and will not be rescheduled. 3. The suspend command only prevents the task (or tasks) from being scheduled in the future.

Resuming a Task

The cron resume function resumes the scheduling of a suspended task (or all tasks) in the scheduler of iSM.

To resume a SUSPENDED task in the schedule of iSM, the general command format is:

```
cron resume [-name task]
```

Parameter	Description
-name	Optional name of the task to resume scheduling. The named task execution will be scheduled to start at its next regularly scheduled time. **Notes:** If -name is not supplied, all SUSPENDED tasks will be scheduled to execute at their next regularly scheduled time.

iWay Service Manager Schedule Command Console

The schedule command allows you to add, suspend, resume or cancel tasks during the current instance of the Service Manager. Unless otherwise noted in the function, tasks that are scheduled through the command console will not be saved in the repository of iSM nor will they be carried over from one instance of the Service Manager to the next when recycled.

Unlike the cron interface named parameters may be entered in any order (for example -hour may proceed -minute, -name may follow -command, and so on). The command line input is evaluated from left to right following the schedule command verb (add, list, cancel, or delete). Each parameter is terminated by the start of another named parameter or by a carriage return.

Note: Named parameters are parameters that are proceeded by a dash ('-') then the parameter name, for example, -name, -hour, and so on.

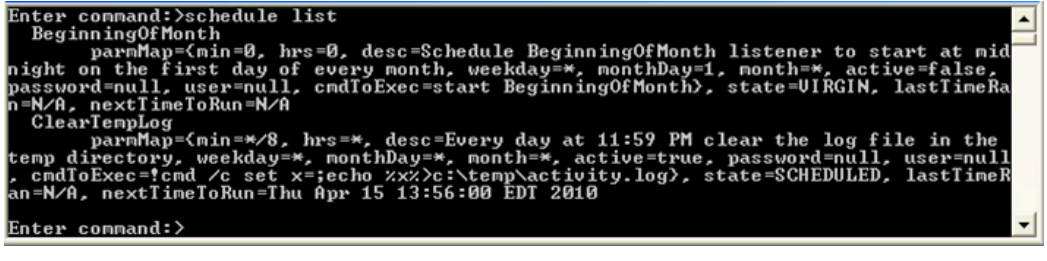
Listing a Schedule

Entering the command schedule or schedule list produces a listing of all currently scheduled Service Manager tasks. To list one or all of the currently scheduled Service Manager tasks the general command format is:

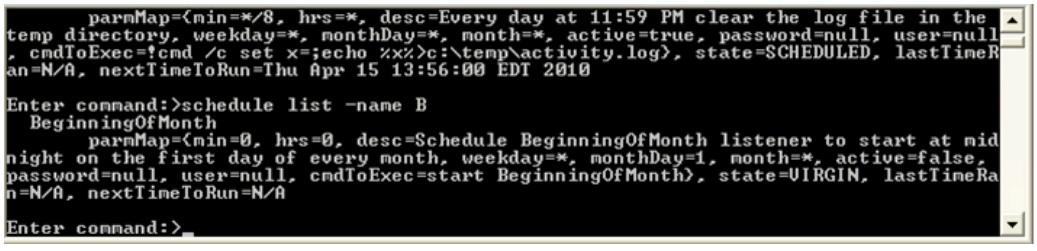
```
schedule [list [-name task]]
```

Parameter	Description
`-name`	Is the name or partial name of the task. The name can contain only part of a task name, for example, *t* to list all tasks that start with the letter *t*, *te* to list all tasks that start with the letters *te*, and so on.
`-all`	When the list command is issued without the all the -all parameter, only active tasks are listed. Tasks that are suspended are not displayed. Using the -all parameter lists both active and suspended tasks.

The following image shows an example of the schedule list output.



Entering the command schedule list -name <value> (in the following example case 'B*') produces a list of tasks that start with the value. The following image shows an example of the schedule list -name function output.



The schedule list displays the following information on the console.

Values	Description
name	Name of the task in the schedule.
parmMap	Parameter of the Task. Each Scheduler entry in the dictionary represents a job and follows a particular format as a series of fields, separated by spaces and/or tabs. Each field can have a single value or a series of values. ❑ **min.** A number between 0 and 59 that represents what minute of the hour the task should start execution. Zero (0) will start the task at the top of each hour. ❑ **hrs.** A number between 0 and 23 that represents what hour of the day the task should start execution. Zero (0) representing midnight (12 AM); 23 representing 11 PM. ❑ **desc.** Task description. ❑ **weekday.** A number between 0 (Sunday) and 6 (Saturday) that represents what day of the week that the task should run on. In addition to the numerical representation the weekday names or abbreviations may also be used, for example, Sun, Mon, Tue, and so on.

Values	Description
parmMap (continued)	❑ **month.** A number between 1 (January) and 12 (December) that represents what month that the task should run on. In addition to the numerical representation the month names or abbreviations may also be used, for example, Jan, Feb, Mar, and so on. ❑ **active.** If set to true, this task will be scheduled each and every time the Service Manager is recycled. ❑ **password.** If this task is being executed with the credentials of a different user, enter the password of the user. ❑ **user.** If this task must be executed under the credentials of a different user, enter the User ID to use when executing this task. ❑ **cmdToExecute.** The command that the task will execute when the scheduled time comes. Any Service Manager command may be executed. Some Service Manager commands make more sense than others to schedule as tasks. ❑ **dependent.** Dependent command to run when duration timer expires. ❑ **monthDay.** A number between 1 and 31 that represents what day of the month that the task should run on. In addition to the numbers the following special values may also be entered: ❑ **@FMOM.** First Monday of the month, can be abbreviated as @FM. ❑ **@LFOM.** Last Friday of the month, can be abbreviated as @LF. ❑ **@LBDOM.** Last business day of the month. This value will return the last calendar workday (Monday through Friday) of the month. This value can be abbreviated as @LBD. ❑ **@LDOM.** Last day of the month, can be abbreviated as @LD.

Values	Description
`parmMap` (continued)	❑ **duration.** Time between the start of the cmdToExecute and when to start the dependent command. There are several ways of specifying multiple date and time values in a field: ❑ The comma (,) operator specifies a list of values, for example: 1, 3, 4, 7, 8. ❑ The dash (-) operator specifies a range of values, for example: 1 - 6, which is equivalent to 1, 2, 3, 4, 5, 6. ❑ The asterisk (*) operator specifies all possible values for a field. For example, an asterisk in the hour time field would be equivalent to every hour (subject to matching other specified fields). ❑ The slash (/) operator (called step), which can be used to skip a given number of values. For example: */3 in the hour time field is equivalent to 0, 3, 6, 9, 12, 15, 18, 21. So * specifies *every hour*, but the */3 means only those hours divisible by 3. The meaning of / specifier, however, means *when the modulo is zero* rather than *every*. If an * does not proceed the / (for example, /2, /5, and so on) it directs the scheduler to execute the command every n cycles where is the number that follows the step.
`state`	Current state of the task. This can be one of the following values: ❑ `VIRGIN`. Task has not been scheduled. This state is an indication that the active flag of the task is set to false. ❑ `SCHEDULED`. Task is scheduled to run. ❑ `RUNNING`. Task is currently running. ❑ `COMPLETE`. Task is complete but has not been rescheduled. ❑ `CANCELED`. Task has been canceled and not rescheduled. ❑ `ERROR`. Task ended in error.

Values	Description
lastTimeRan	Time of day that the task was last ran. A value of 'N/A' indicates that the task has not been run during this instance of the Service Manager.
nextTimeToRun	When the task is scheduled to run again.

Adding a Task

The add function adds a task to the schedule of iSM. Tasks added using the command line are immediately added to the schedule. Tasks added using the command line will not be persisted to the repository of iSM unless the -save option is specified in the command line.

The general command line format is:

```
schedule add [parameters]
```

Parameters	Description
-name	Enter a unique name to associate to the task. This name will be used when looking up the task later. If -name is missing the Service Manager will generate a name for the task.
-active	A value of either true or false. If true, the task is scheduled to run immediately. If the value is false then the task is not scheduled to run. Optional, if omitted, a false value is assumed.
-minute	A numeric value between 0 and 59 representing when within the hour the task should run. Optional, if omitted, an * is assumed (all minutes in the hour).
-hour	A numeric value between 0 and 23 (where 0 = 12am, 23 = 11pm) representing when within the day the task should run. Optional, if omitted, an * is assumed (all hours in the day).

Parameters	Description
`-day`	A numeric value between 1 and 31 representing what day in the month the task should run. Optional, if omitted, an * is assumed (all days in the month).
`-month`	A numeric value between 1 and 12 (where 1=January, 12=December) representing what month the task should run. Alternately, the text name of the month, (for example, January[jan], February[feb], March[mar],...) can also be used. Optional, if omitted, an * is assumed (all months in the year).
`-weekday`	A numeric value between 0 and 6 (where 0=Sunday and 6=Saturday) representing what day of the week the task should run. Alternately the text name of the weekdays (for example, Sunday[sun], Monday[mon], Tuesday[tue],...) can also be used. Optional, if omitted, an * is assumed (all days in the week).
`-command`	The Service Manager command (for example, start listener, stop listener, and so on) that the task will execute. (For more information, see the *iWay Service Manager User's Guide*).
`-duration`	Length of time that the task will run prior to the Dependent Command. The format of duration [in seconds] is in the form [xxh][xxm]xx[s]. For example 04h30m45, which creates a duration of 4 hours, 30 minutes, and 45 seconds.
`-dependent`	The Service Manager command to be executed after the Duration Timer of the task has expired. (For example, start listener, stop listener, and so on) that the task will execute. (For more information, see the *iWay Service Manager User's Guide*).
`-user`	If the task must be ran with an alternate user ID, enter the ID of the alternate user for the value of this parameter.

Parameters	Description
-password	If the task must be ran with an alternate user ID, enter the password of the alternate user for the value of this parameter.
-save	Save the added scheduled task in the repository of iSM. This allows the Service Manager to reschedule this task when the current instance is recycled.

Note:

1. Adding a task with the same name as a currently scheduled task will cause the previously scheduled task (with the same name) to be canceled and this new task to be scheduled in place of the old task.
2. Time parameters not entered (for example, -minute, -hour, -day, -month, -weekday) will default to the asterisk (*) operator (meaning all valid values for that field).
3. There are several ways of specifying multiple date/time values in a field: The comma (,) operator specifies a list of values, for example: 1, 3, 4, 7, 8. The dash (-) operator specifies a range of values, for example, 1 - 6, which is equivalent to 1, 2, 3, 4, 5, 6.
4. The asterisk (*) operator specifies all possible values for a field. For example, an asterisk in the hour time field would be equivalent to every hour (subject to matching other specified fields).
5. The slash (/) operator (called step), which can be used to skip a given number of values. For example, */3 in the hour time field is equivalent to 0, 3, 6, 9, 12, 15, 18, 21. So * specifies every hour, but the */3 means only those hours divisible by 3. The meaning of */ specifier, however, means '*when the modulo is zero* rather than *every*.

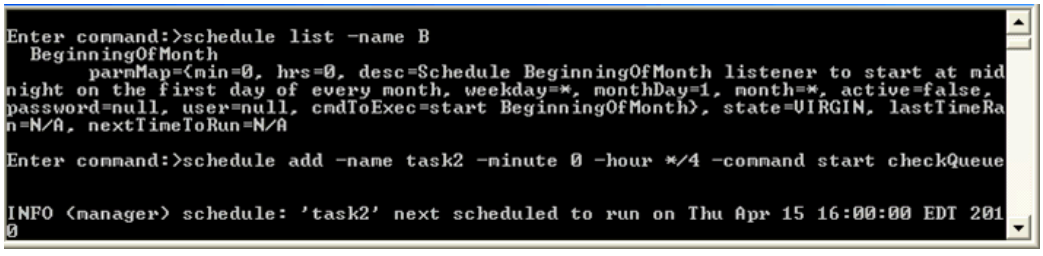
   It the * does not proceed the step (/) (for example, /2, /5, and so on) it directs the scheduler to execute the command every n cycles (minutes, hours, and so on) where n is the number that follows the step (/) character.

Adding a Task to Start a Listener

To add a Service Manager task named task2 that will start a listener named checkQueue at the top of the hour, every four hours every day of the week, every month of the year, enter the following command (all one line) following the Enter command:> prompt:

```
schedule add -name task2 -minute 0 -hour */4 -command start checkQueue
```

The following image shows the results of the command.

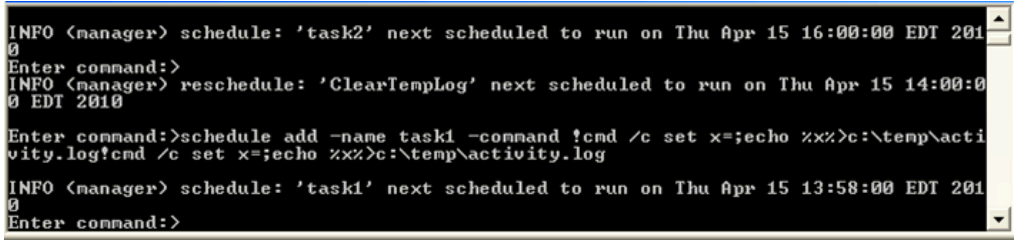


Adding a Task to Execute an External Command

To add a Service Manager task named task1 that will clear the file activity.log found in the subdirectory c:\temp every minute of every hour of every day of the week every month of the year, enter the following command (all one line) following the Enter command:> prompt:

```
schedule add -name task1 -command !cmd /c set x=;echo
 %x%>c:\temp\activity.log!cmd /c set x=;echo %x%>c:\temp\activity.log
```

The following image shows the results of the command.



Canceling a Task

The cancel function removes the named task from the schedule of iSM. The task, if currently processing, completes its processing cycle prior to being canceled.

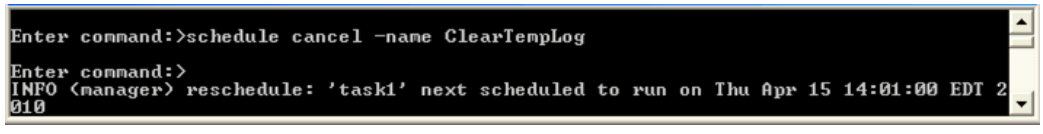
Canceling a task only removes it from the current instance of the schedule. When iSM is recycled, the task (if persisted in the repository of Service Manager) will be rescheduled.

The general command line format is:

```
schedule cancel -name task
```

Parameter	Description
-name	Name of the task to cancel. The named task will be canceled and removed from the Service Managers schedule.

In the image that follows, the iWay Service Manager task ClearTempLog has been canceled.



If the task that is canceled has a dependent task configured, the dependent task is canceled from the Service Managers schedule.

Suspending a Task

The suspend function suspends the scheduling of the next invocation of the task by the scheduler of iSM. The task, if currently processing, completes its processing cycle but will not be rescheduled.

The general command line format is:

```
schedule suspend [-name task]
```

Parameter	Description
`-name`	Optional name of the task to suspend. The named task execution will be suspended. The task will remain on the Schedule list of iSM with a status of SUSPENDED. To restart the task at its next regularly scheduled time, use schedule resume. **Notes:** 1. If -name is not supplied, ALL SCHEDULED tasks will be suspended. 2. If the suspend command is issued while a task is running, the running task is not interrupted but will complete normally; and will not be rescheduled. 3. The suspend command only prevents the task (or tasks) from being scheduled in the future.

If the task that is suspended has a dependent task configured, the dependent task is canceled from the Service Managers schedule.

Resuming a Task

The resume function resumes the scheduling of a suspended task in the scheduler of iSM. The named task will be scheduled to execute at its next regularly scheduled time.

The general command line format is:

```
schedule resume [-name task]
```

Parameter	Description
-name	Optional name of the task to resume scheduling. The named task execution will be scheduled to start at its next regularly scheduled time. If the -name parameter is missing, all tasks in the Service Manager schedule that are suspended will resume scheduled execution on their next regularly scheduled time.

If the task that is resumed has a dependent task configured, the dependent task is only scheduled by the Service Managers immediately following the start of the configured primary task.

Command Line Schedule Examples

This section provides examples on how to configure various schedules using the command line.

Once a Year

To schedule the same task using the command line, use the following command:

```
schedule add -name RunOnNewYear -description Run once a year on New Year's
Day at
   12:01am -minute 01 -hour 0 -day 1 -month January -command run
script.xyz.scr -save
```

Once a Month

To schedule the same task using the command line, use the following command:

```
schedule add -name LDOBScript -description "On the last day of business run
script
   scr.xyz.scr" -minute 0 -hour 0 -day @LBDOM -command run script.xyz.scr -
save
```

Once a Week

To schedule the same task using the command line, use the following command:

```
schedule add -name RunOnMonday -description "Run the script script.xyz.scr
on
   Monday evening at 11:59pm" -minute 59 -hour 23 -weekday Monday -command
run script.xyz.scr -save
```

Daily

To schedule the same task using the command line, use the following command:

```
schedule add -name RunDaily -description "Run the script script.xyz twice
daily;
   once at 12pm (noon) and again at 6pm" -minute 0 -hour 12,18 -weekday
Mon,Tue,Wed,Thu,Fri
   -command run script.xyz.scr -save
```

Chapter 8

iWay Migration Extension

This section provides an overview of the iWay Migration extension.

In this chapter:

iWay Migration Extension Overview

iWay Software has made every effort to ensure that iWay Service Manager (iSM) Version 7.0 is compatible with earlier versions. Overall, this effort has been successful. Specifically, all channels and process flows that were designed in earlier iSM versions (for example, 6.6 and 6.5) are expected to function properly in iSM Version 7.0. The exception is any application that depends on timing or "bugs." Applications that follow best practices should port over to iSM Version 7.0 seamlessly.

The only major exception are the cases where applications rely on situations where strict adherence to RFC recommendations have not been applied.

Since iSM 7.0 is a major version, many new features including improved performance and security are available. As a result, some incompatibilities may arise which can be handled by upgrading component configurations.

The iWay Migration extension provides tools to migrate an iSM 6.x Version to a 7.0 Version including providers, special registers, schemas, and tools to compare IFL functions and XPaths.

For more information on the iWay Migration extension, see the *iWay Service Manager Migration Guide*.

Installing the iWay Migration Extension

To install the iWay Migration extension, you must add the Migration Tools extension to your iWay Service Manager instance during the iWay Service Manager installation. For more information on installing iWay Service Manager, see the *iWay Installation and Configuration Guide*.

Chapter 9

iWay Real Time Data Replication Extension

This section describes how to configure the iWay Real Time Data Replication (RTDR) extension.

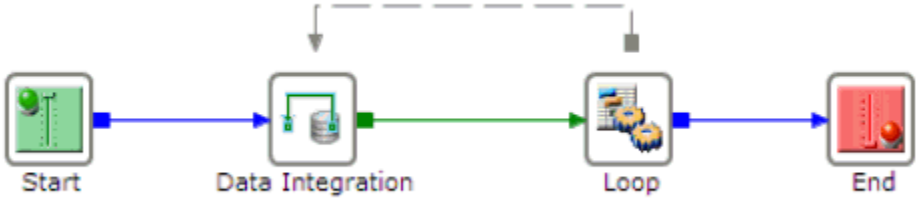
In this chapter:

iWay Real Time Data Replication Extension Overview

The iWay Real Time Data Replication (RTDR) extension is composed of the SQL Batch Insert Iterator object and the Data Integration object. The Data Integration object is supported as of iWay Service Manager (iSM) Version 6.1 and higher.

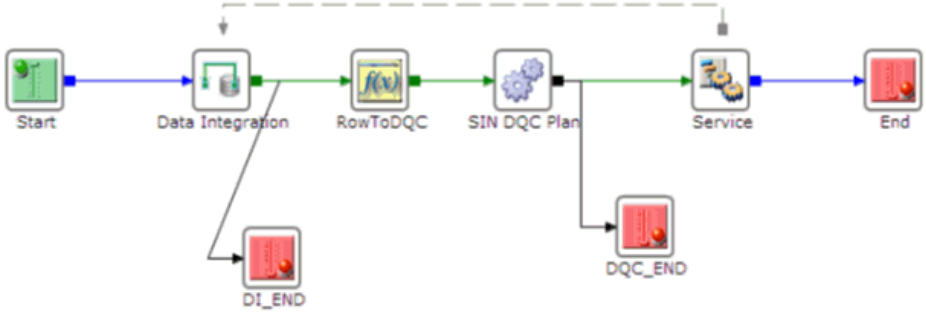
Data Replication Use Case

In the simplest use case, the iWay RTDR extension can be used to replicate data from a source database to a target database. The source and target databases can be the same or different types. Both databases must exist prior to replication.



Data Cleansing or Transformation Use Case

In a more complex use case, the iWay RTDR extension can be used to extract data one row at a time from a source database, cleanse or transform the data, and then insert the row into another database. The source and target databases must exist prior to execution.



Adding Data Integration Object Support for Process Flows

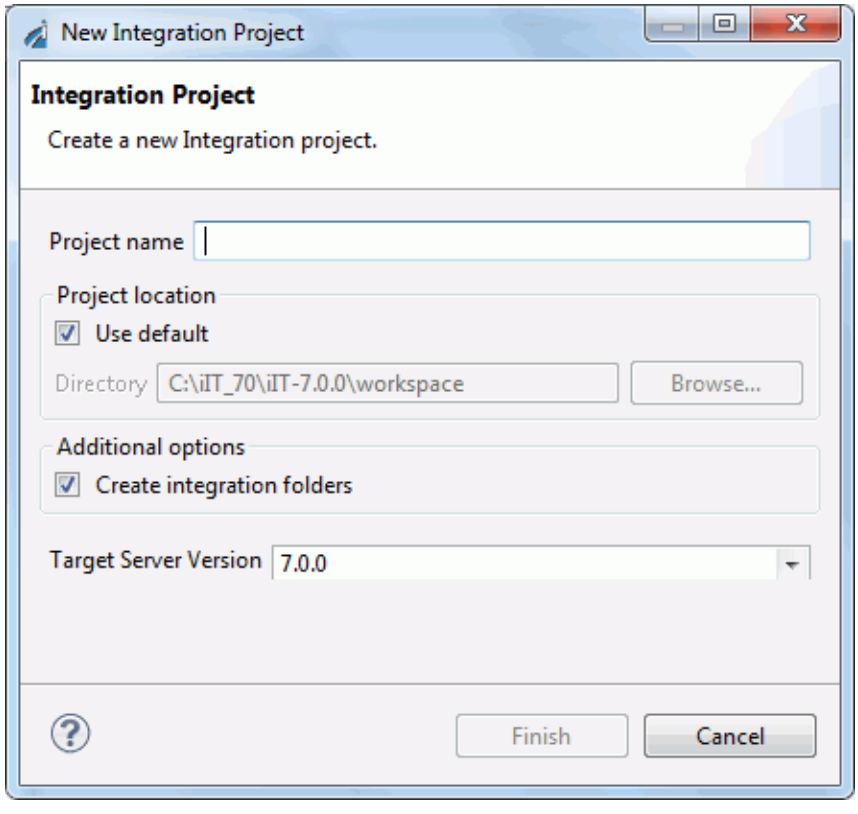
If you are creating process flows using iSM Version 7.0 that is patched to support the iWay Real Time Data Replication (RTDR) extension, you must add the SQL Batch Insert Iterator to the Additional Components customization project preference before the Data Integration object is available for selection from the palette.

Procedure: How to Add Data Integration Object Support for Process Flows

To add Data Integration object support for process flows:

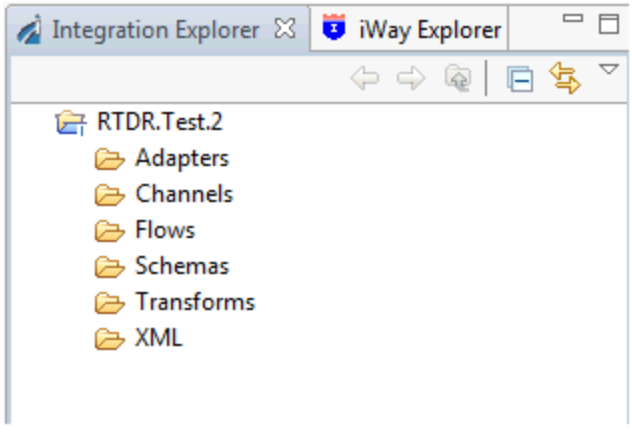
1. Start iIT and create a new Integration project by clicking *File*, selecting *New*, *Integration*, and then clicking *Project*.

   The New Integration Project dialog opens, as shown in the following image.



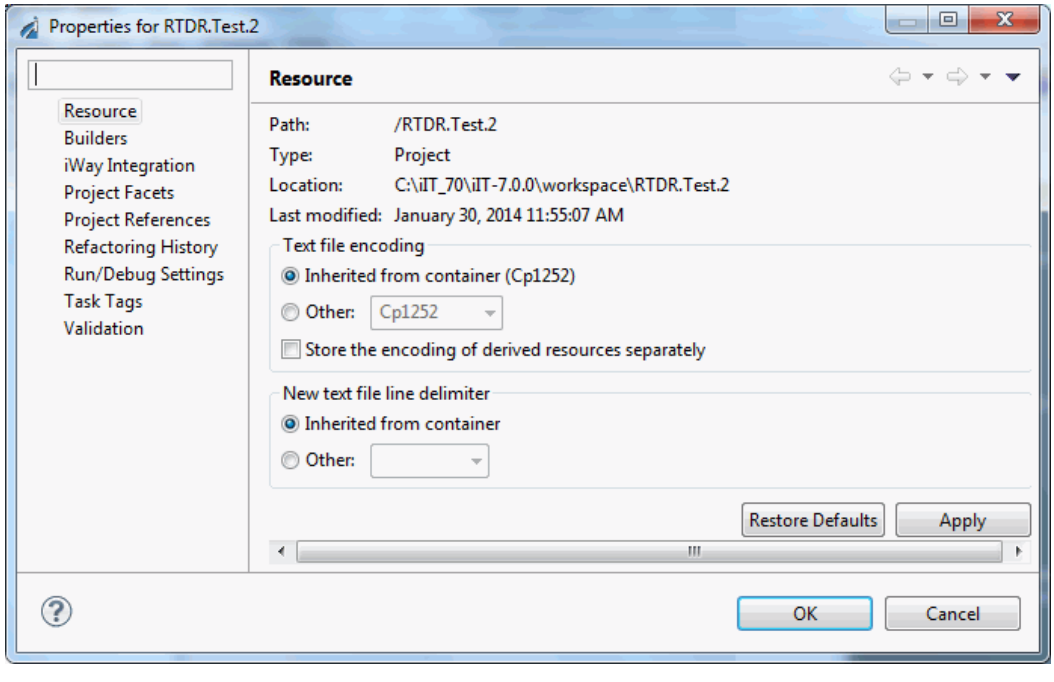
2. Select *7.0.0* from the Target Server Version drop-down list.
3. Enter a new project name and click *Finish*.

The new Integration project is created in the Integration Explorer tab, as shown in the following image.

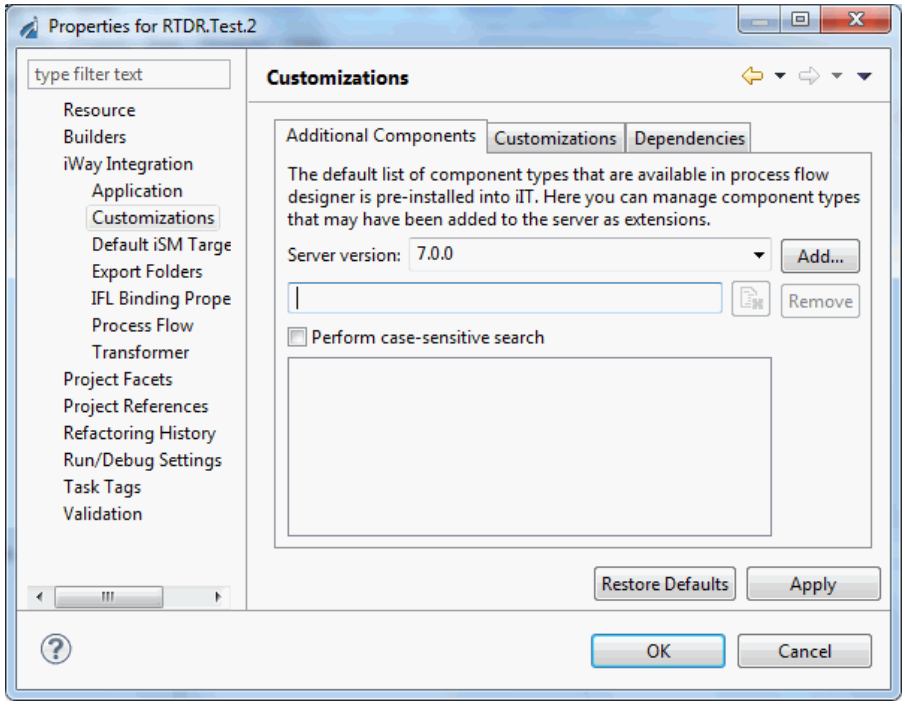


4. Right-click the new Integration project and select *Properties*.

   The Properties dialog for the Integration project opens, as shown in the following image.



5. Expand the *iWay Integration* category and select *Customizations*, as shown in the following image.



6. Select *7.0.0* from the Server version drop-down list.
7. Click the *Add* button to the right of the Server version drop-down list.

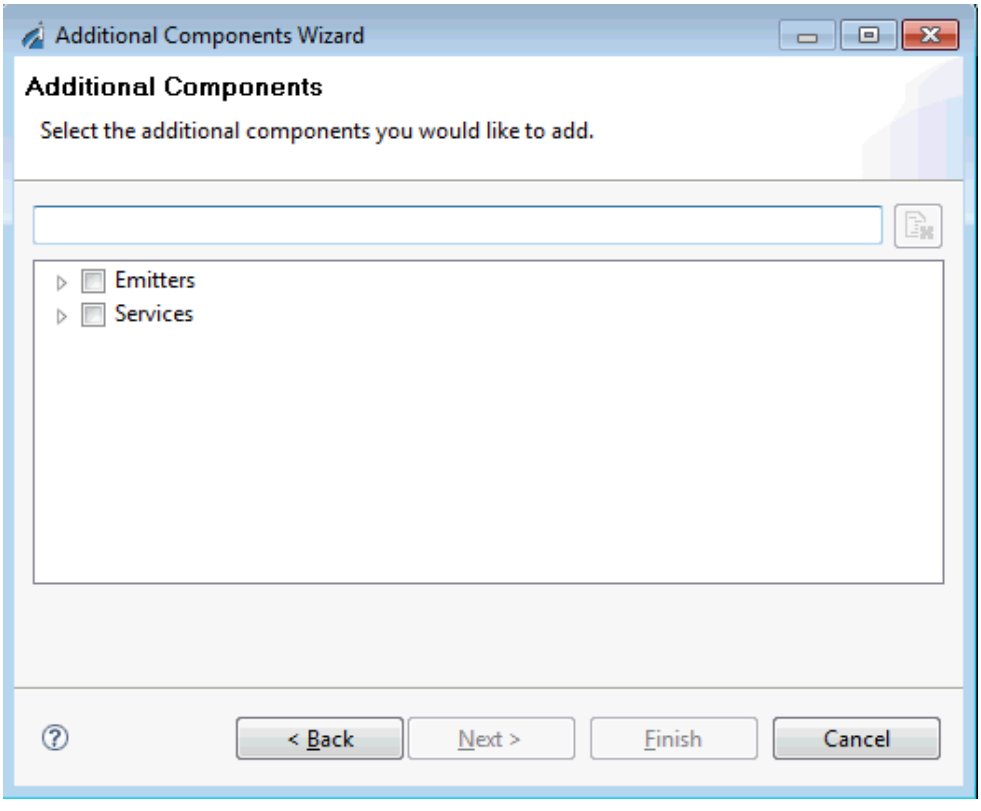
   The Additional Components Wizard opens.
8. In the Server URL field, select an available server from the drop-down list or enter the URL and SOAP port of a server directly in the field.

   For example, if you are using a default iSM installation on your local system, then the URL would be as follows:

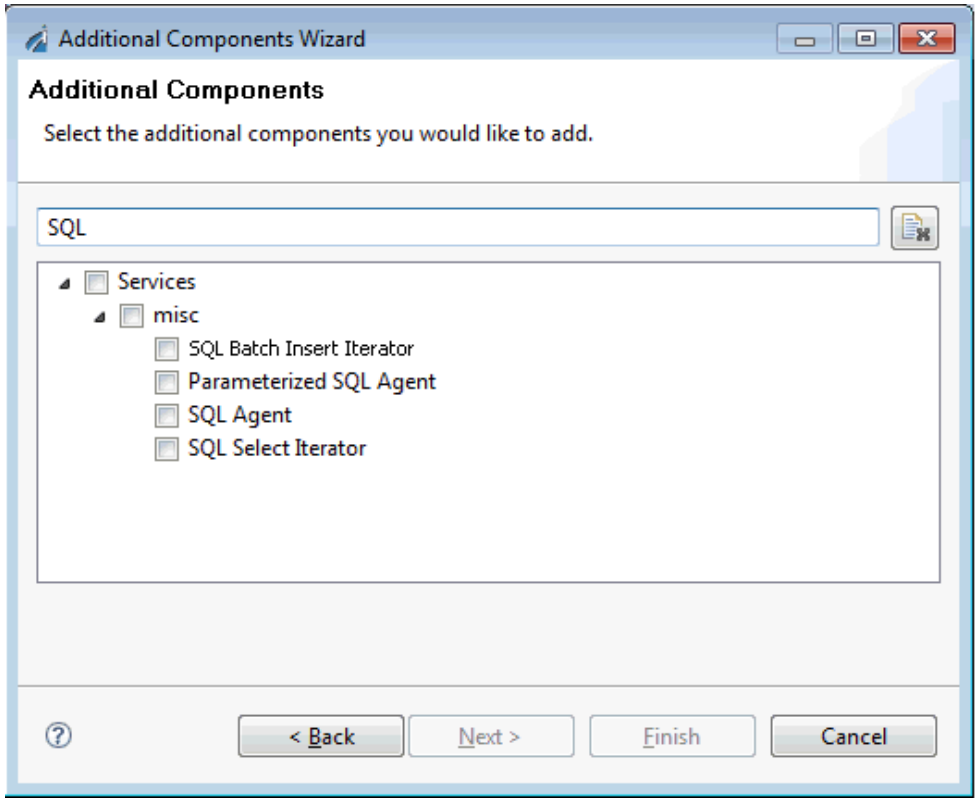
   ```
   http://localhost:9000
   ```

   Note: The Server URL field will only be populated with available servers if you have added connections using iWay Explorer. For more information, see the *iWay Integration Tools User Guide*.
9. Click *Next*.

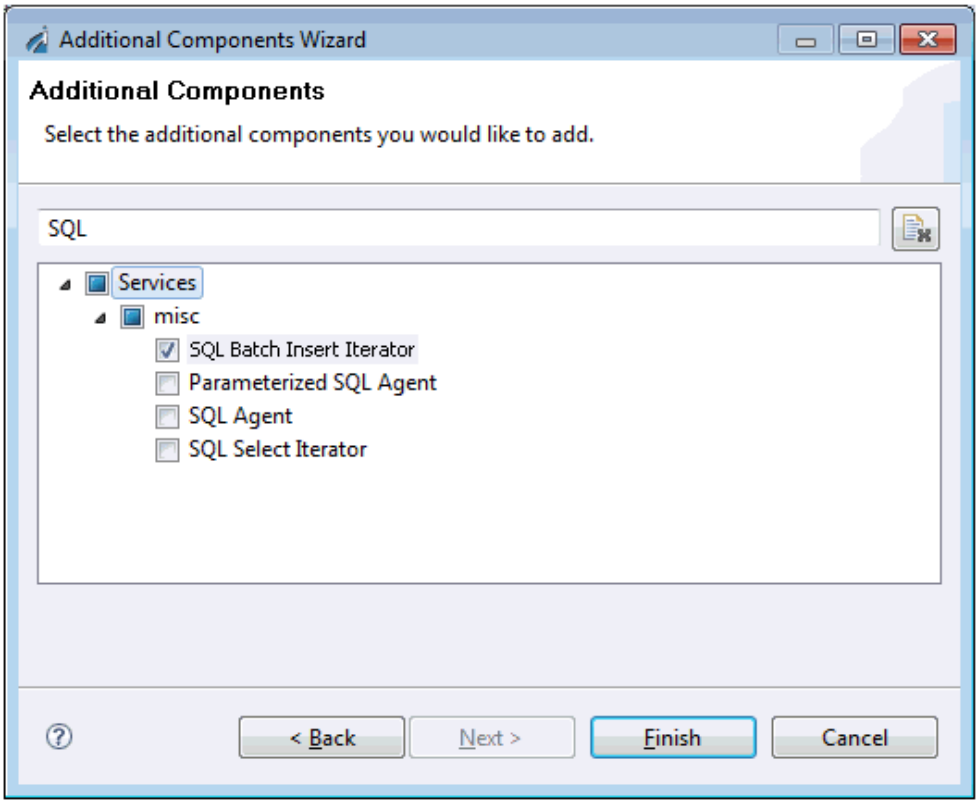
The Additional Components pane opens, as shown in the following image.



10. In the filter area, type *SQL*, as shown in the following image.

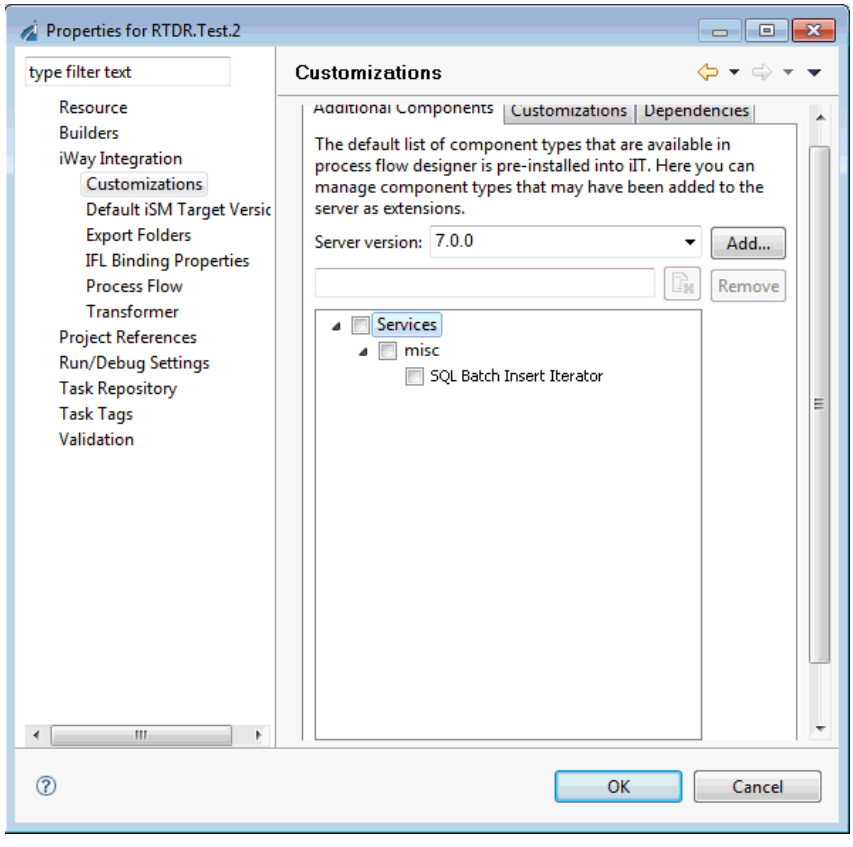


11. Select the *SQL Batch Insert Iterator* check box, as shown in the following image.



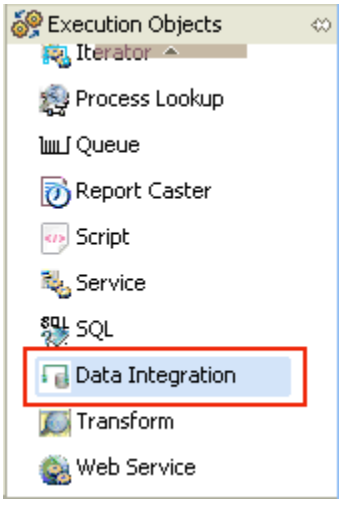
12. Click *Finish*.

The service is now added to the Customizations pane in the Additional Components tab, as shown in the following image.



13. Click *OK* to close the Properties dialog.

The Data Integration object is now available for selection from the Execution Objects palette when you create a new process flow in the current Integration project with the Target Server Version set to 7.0.0.



If there are any open process flows with the Target Server Version set to 7.0.0 in the current project, they will need to be closed and reopened for the Data Integration object to be available on the palette.

Note: As customizations are specific to individual projects, you must follow this procedure for any new project that requires use of the Data Integration object.

Creating a Connection Using the Data Source Explorer

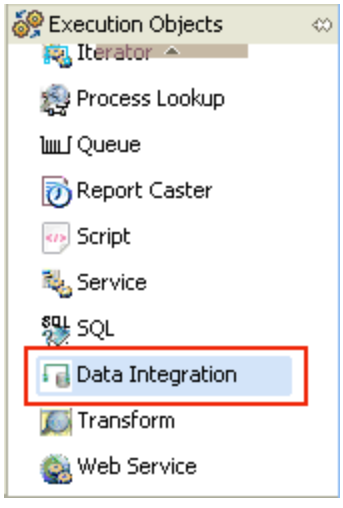
iWay Integration Tools (iIT) can be used to browse and work with data sources using the Eclipse Data Tools Project (DTP). The DTP provides a large amount of functionality for working with data sources. The full set of available functionality for DTP can be found in the Help Contents of iIT, which can be accessed by clicking *Help* and then clicking *Help Contents* from the context menu.

Your first step should be to create a connection to the required data source, which will allow you to use the SQL Builder in the Data Integration object. This can be accomplished by using the Data Source Explorer. For more information on creating a connection profile and a driver definition, see the DTP Help.

Configuring a Data Integration Object

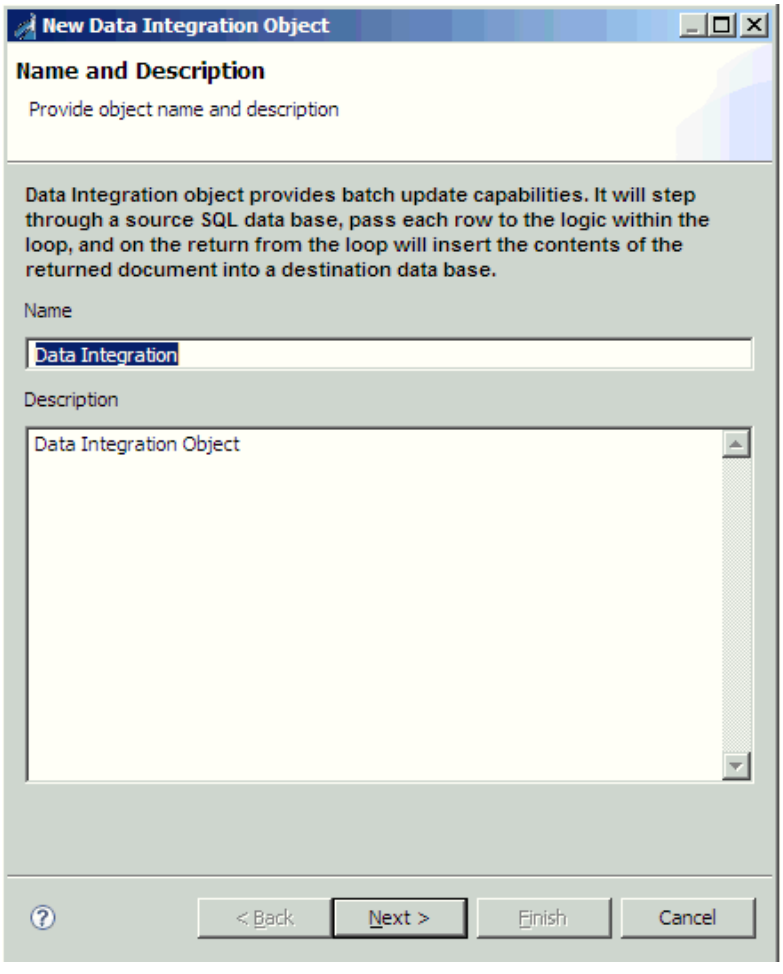
This section describes how to configure a Data Integration object in a process flow.

To open the New Data Integration Object wizard, drag the Data Integration object from the Execution Objects palette to a process flow.



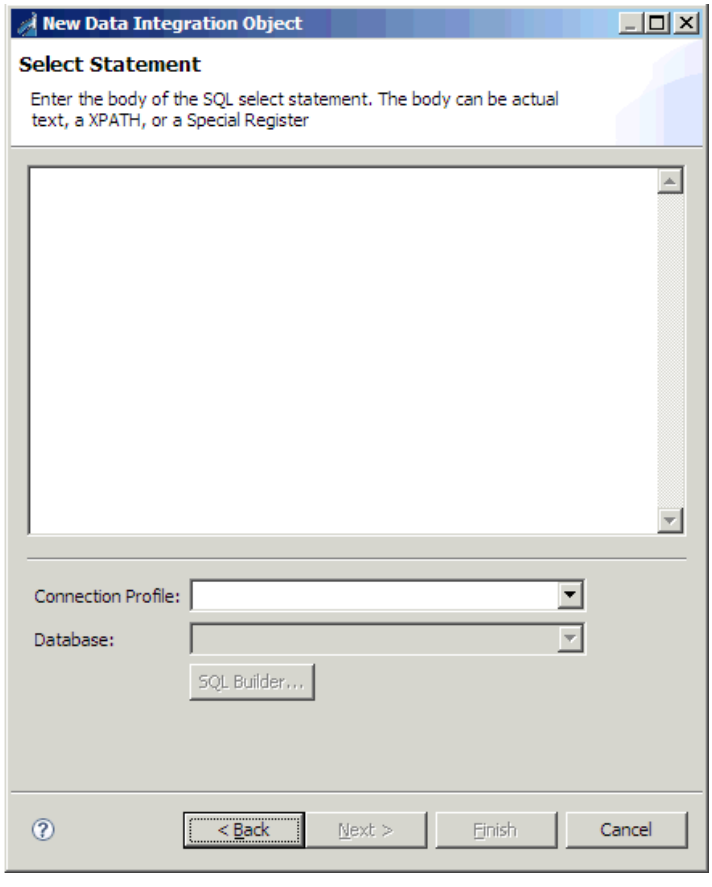
Name and Description Pane

The following image shows the Name and Description pane, where you can enter a name and description for an instance of the Data Integration object. This name will display in your process flow.



Select Statement Pane

The Select Statement pane allows you to enter an SQL statement to retrieve the records for which you are interested. The SQL Query Builder provided by the DTP can be used to assist in this task.



Procedure: How to Access the SQL Query Builder

To access the SQL Query Builder:

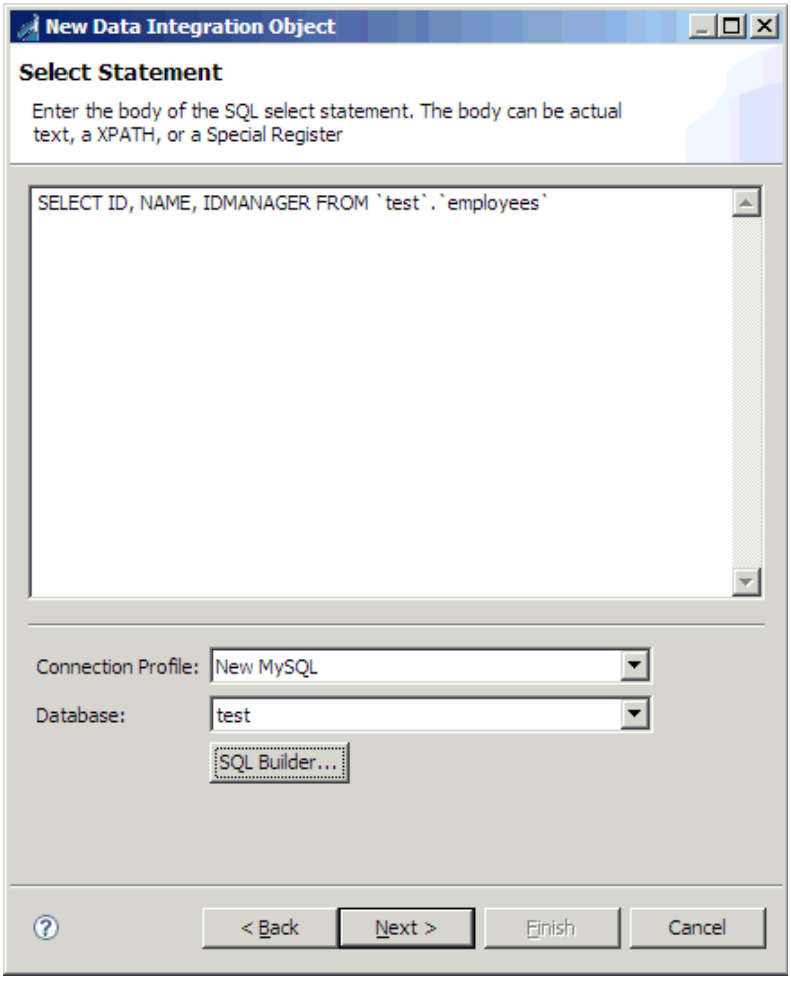
1. Select the connection profile from the *Connection Profile* drop-down list.

   If you have not created a connection profile, exit the wizard by clicking *Cancel*, and follow the steps in the DTP help.

2. Select the database from the *Database* drop-down list.
3. Click *SQL Builder*.

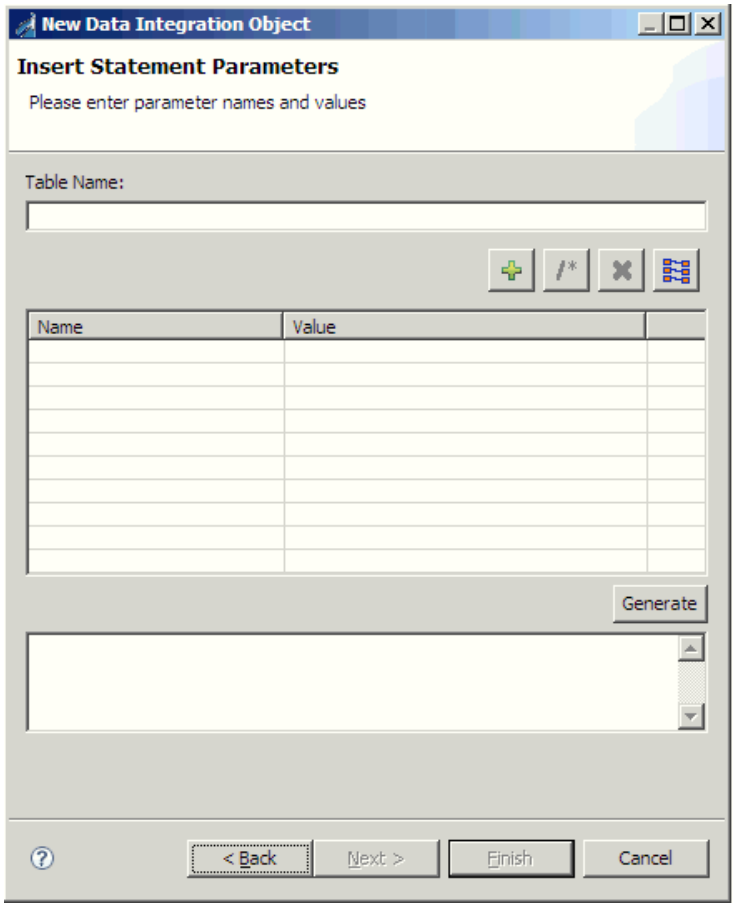
For more information on using the SQL Query Builder, see the DTP help.

4. Once you have completed entering the SELECT statement, click *Next*, as shown in the following image.



Insert Statement Parameters Pane

This section describes the Insert Statement Parameters pane, as shown in the following image.



Parameterized SQL

The destination SQL is expected to take the following form:

```
INSERT INTO <table name> (<column name>*) VALUES (<?name>*)
```

where:

?name

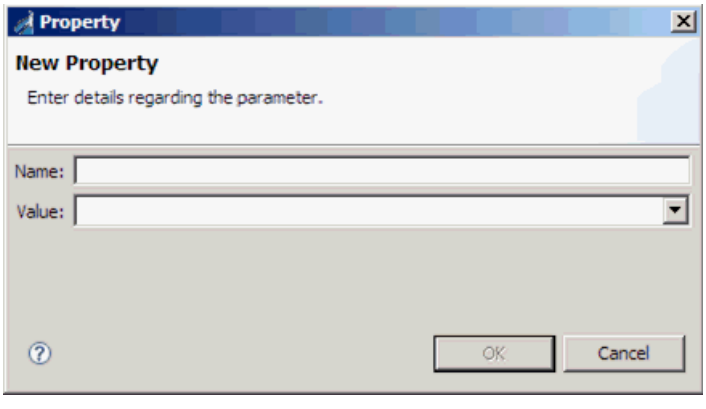
Is the name of a user parameter, the value of which is inserted for the specific row. It is anticipated that this will be an _iwxpath() expression to locate the value in the document that reaches the iterator on the loop edge. It is not required for the name to be the actual column name in the destination table.

Procedure: How to Generate Parameterized SQL

To generate the required parameterized SQL:

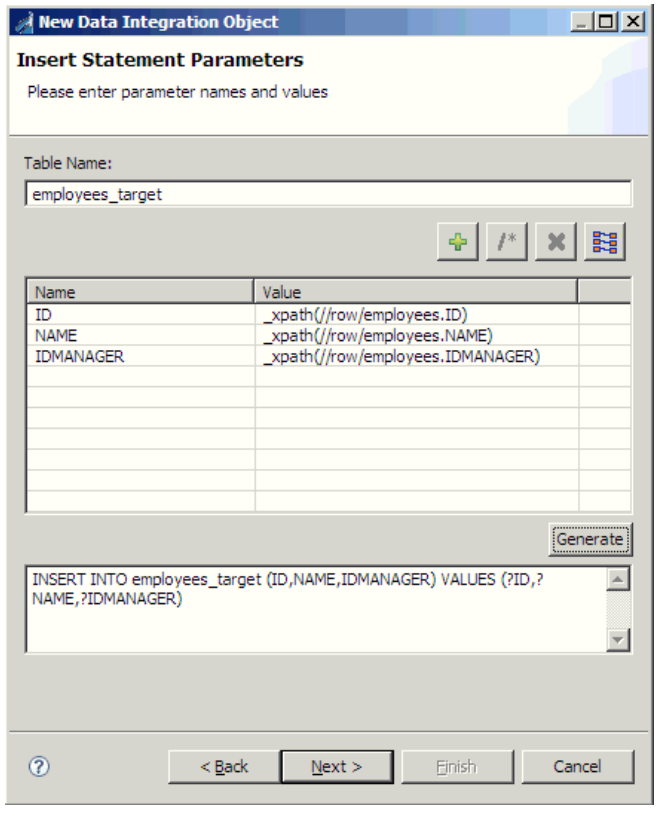
1. Enter a destination table name in the Table Name field.
2. Click the plus (+) icon.

   The Property dialog opens, as shown in the following image.



3. Enter the name of your parameter and a corresponding value. The value can be a constant, XPath, IFL, or SREG.
4. Click *OK*.

5. Once you have entered your parameters, click *Generate* to create the INSERT statement, as shown in the following image.



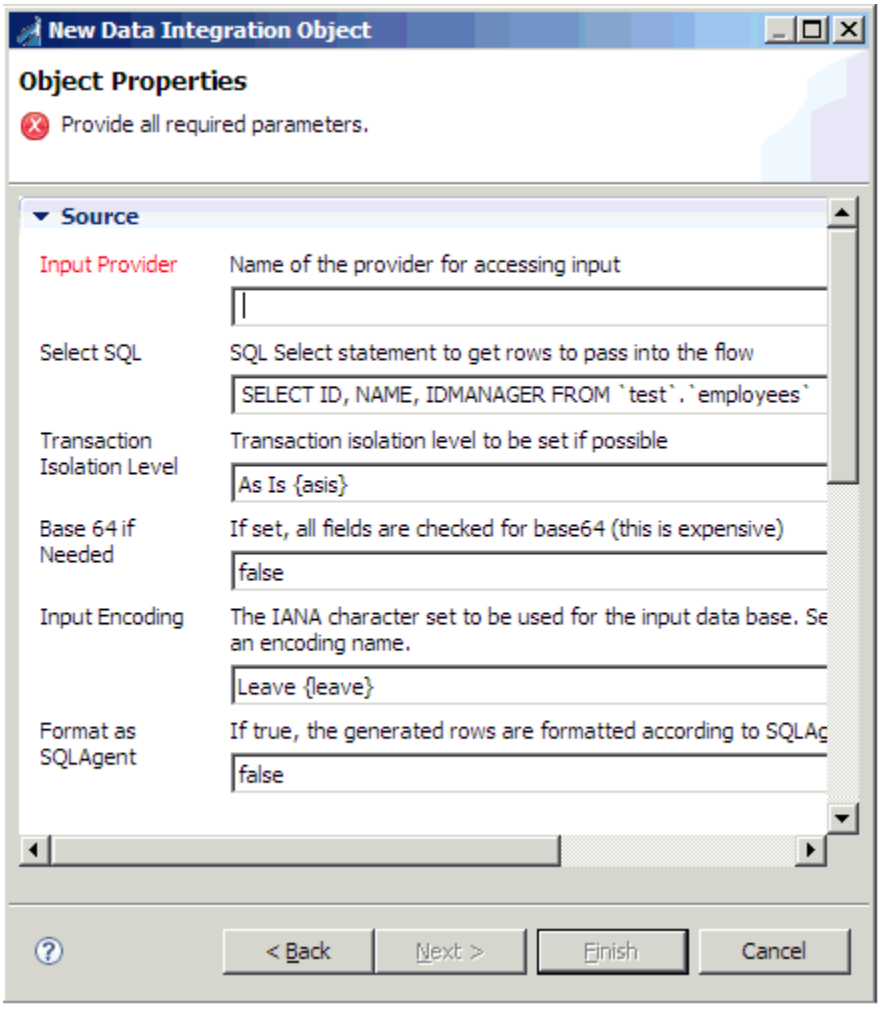
If the target table has the same schema as the source database, you can click the following icon:

This allows you to create parameters and XPath statements to extract the data from an XML document that has the same format as when it left the Data Integration object.

This extracted data is then used to populate the parameters. Clicking *Generate* will then create the INSERT statement using the parameters. You can leverage this feature if your target table has the same number of columns but different names by using aliases in your SELECT statement.

Object Properties Pane

This section describes the Object Properties pane, which is shown in the following image.



The following table lists and describes the available parameters for the Data Integration object.

Name	Description
Source	

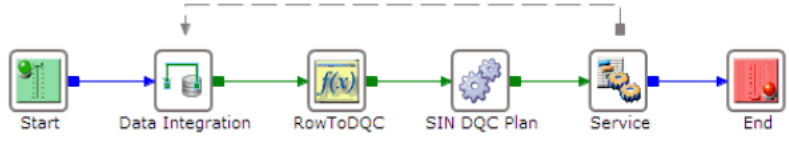
Name	Description
Input Provider (required)	Name of the JDBC provider used to access the input (source) database. This parameter must be provided with the name of the data provider created in iSM for the source database. For more information on creating data providers, see the *iWay Service Manager User's Guide*.
Select SQL	The SQL SELECT statement used to retrieve rows from the source database.
Transaction Isolation Level	Determine if the transactional control of the input should be set to a specific value. The following list shows the available options that can be specified: ❏ **As is.** As set for the database. This value is set by default. ❏ **Read Uncommitted.** Allows reading of a record that may be rolled back later. ❏ **Read Committed.** Will never read data that another application has changed and not yet committed, but does not ensure that the data will not be changed before the end of the transaction. ❏ **Repeatable Read.** Dirty reads and non-repeatable reads can not occur. All data used in the query is locked, and other transactions cannot update the data. ❏ **Serializable.** Most restrictive isolation level. Phantom values cannot occur. This prevents other users from updating or inserting rows into the data set until the transaction is completed.
Base 64 if Needed	Determines if binary data should be passed to the loop as base 64 if binary data is read. This value is set to *false* by default.

Name	Description
Input Encoding	Determines what IANA encoding should be assumed for binary data during conversion to base 64. The default is the current system default.
Format as SQLAgent	If set to *true*, then the generated rows are formatted according to the SQL service field schema. This value is set to *false* by default.
Three Part Name	If set to *true*, then the names will be presented as full three part names if supported by the database. This value is set to *false* by default.
Destination	
Output Provider (required)	Name of the JDBC provider used to access the output (destination) database. This parameter must be provided with the name of the data provider created in iSM for the destination database. For more information on creating data providers, see the *iWay Service Manager User's Guide*.
Output Insert	The SQL INSERT statement used to insert data into the target database. This SQL uses a special format to delineate the columns to receive data from the loop return.
Out Encoding	Determines what IANA encoding should be assumed for binary data during conversion from base 64. The default is the current system default.
Batch Size	Determines how many inserts constitute a batch. Each sub-batch is executed to the destination. The use of sub-batches may reduce memory depending on the characteristics of the destination database and its drivers.
Commit Sub-batches	If sub-batches are requested, determine whether the batches should be committed or if all of the commits should be held for EOS/process flow end or a transactional commit. This value is set to *false* by default.

Name	Description
Fail First	Determine if an insert failure on the first row should be considered as a catastrophic failure. The default is to treat such a failure as a normal row insert failure. This value is set to *false* by default.
Omit Test	If present, this is an iFL test that is evaluated for each candidate destination record. If it evaluates to true, then the candidate record is omitted.
Main	
Output document type	Determines whether the output document that is emitted should be the original input document (input) or a status document (status).

Looping

The Data Integration object can be a connection to any number of objects to perform data cleansing, transformations, lookups, and so on. If any of these tasks are performed, the output must be returned to the Data Integration object. This can be accomplished by adding a Service object that uses a Move service to the process flow, as shown in the following image.



For more information about configuring the Move service, see the *iWay Service Manager Component Reference Guide*.

Connection Options

In addition to the standard events (OnError, OnSuccess, and OnFailure) the Data Integration object includes custom output events, as listed and described in the following table. These can be used by creating an OnCustom connection and selecting the type of edge.

Edge	Description
success	The operation is successful and the document on this edge is the next row.
cancelled	The operation has been cancelled. The cancel status is checked during each iteration.
fail_parse	An iFL expression is not well-formed.
fail_connect	Cannot connect to either the source or the destination database.
fail_connect_source	Cannot connect to the source database.
fail_connect_destination	Cannot connect to the destination database.
fail_insert	An insert failed. Either the interaction with the destination source resulted in an error or the failure tolerance (including the first row test) was reached.
fail_nullability	Attempt to set NULL on a non-nullable column.
fail_operation	Another operation within the iterator failed.
xxxxx (XOpen code)	The XOpen code for a failure when known.

Sample Real Time Data Replication Extension Documents

The table that is being accessed in this sample has four fields. Each row appears one at a time from the iterator. Two rows are shown. The first and the ninth, and the type codes are JDBC. This is the standard *field* form output documented in the RDBMS listener and SQL service.

The first row is shown below:

```
<root>
   <row row="1">
      <Name type="12">Mr. One</Name>
      <Id type="2">2</Id>
      <Dept type="12">ss</Dept>
      <Company type="12">ss</Company>
   </row>
</root>
```

The following instance shows the ninth row:

```
<root>
   <row row="9">
      <Name type="12">Mr. Jay</Name>
      <Id type="2">99</Id>
      <Dept type="12">jj</Dept>
      <Company type="12">jj</Company>
   </row>
</root>
```

If the user requested that the SQL service format be used, then the first row would have looked as follows:

```
<iway>
   <response>
      <cncresult>
         <result format="field">
            <row row="1">
               <Name type="12">ss</Name>
               <Id type="2">2</Id>
               <Dept type="12">ss</Dept>
               <Company type="12">ss</Company>
            </row>
         </result>
      </cncresult>
      <timestamp>2009-11-27T18:48:26Z</timestamp>
   </response>
</iway>
```

On EOS (end of select), the status document is emitted to the EOS service (the one past the bottom of the loop). Note that the number of batches may be greater than one, if sub-batches are used. This example has no failures.

```
<batcheos rowsread="13" batches="1" failures="0" omits="0" />
```

If failures were encountered, then these would have been denoted in a failure section, as shown in the following example:

```
<batcheos rowsread="13" batches="1" failures="2" omits="3">
     <failures>
        <failure row='5' key='Mr. Eee'/>
        <failure row='7' key='Mr. Gee'/>
     </failures>
  </batcheos>
```

An actual error (possibly because of the inability to reach a destination) will result in an error document.

Real Time Data Replication Extension Tips and Tricks

This section describes some tips and tricks that can be used.

Omitting Destination Records

Records for the destination table can be filtered by including an iFL test. If the test is present, it is evaluated on each row. If the test evaluates as *true*, then the record is not written to the destination.

Some methods to indicate that a specific record should be omitted, might be to include an attribute in the root of the XML that is returned to the iterator, which would normally become the next destination record. For example, the test might be:

```
_iwxpath(/root/@omit)=true
```

A special register (SREG) can also be set with a predetermined name, such as *omit*, and the test would then be:

```
_sreg('omit', 'false')='true'
```

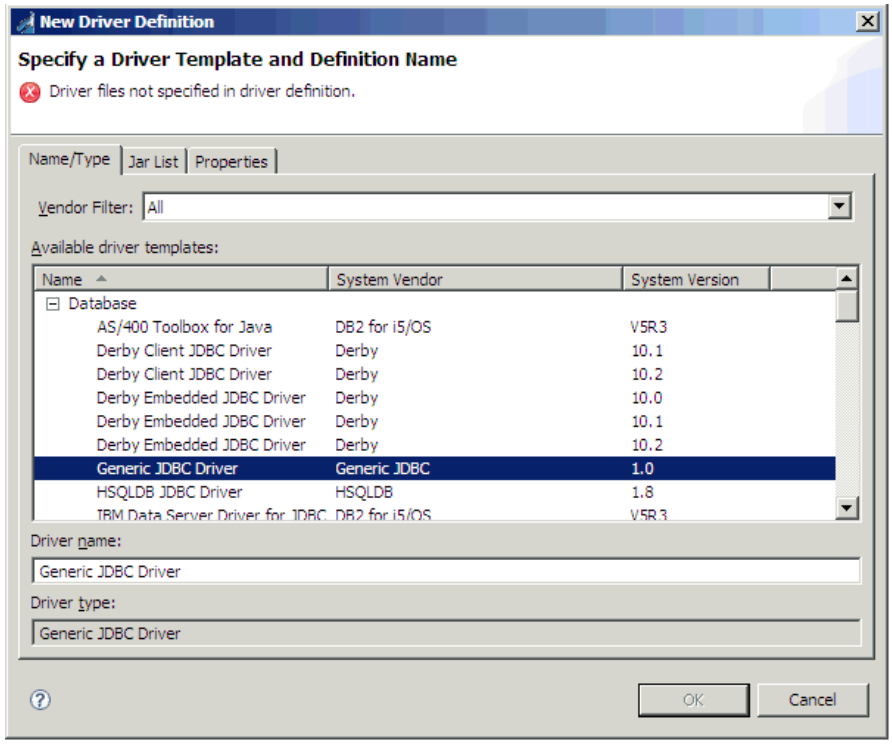
Using the Generic JDBC Driver Definition

The Data Tools Project (DTP) does not provide a predefined JDBC driver template for SQL Server Version 2008. However, the generic JDBC driver definition can be used.

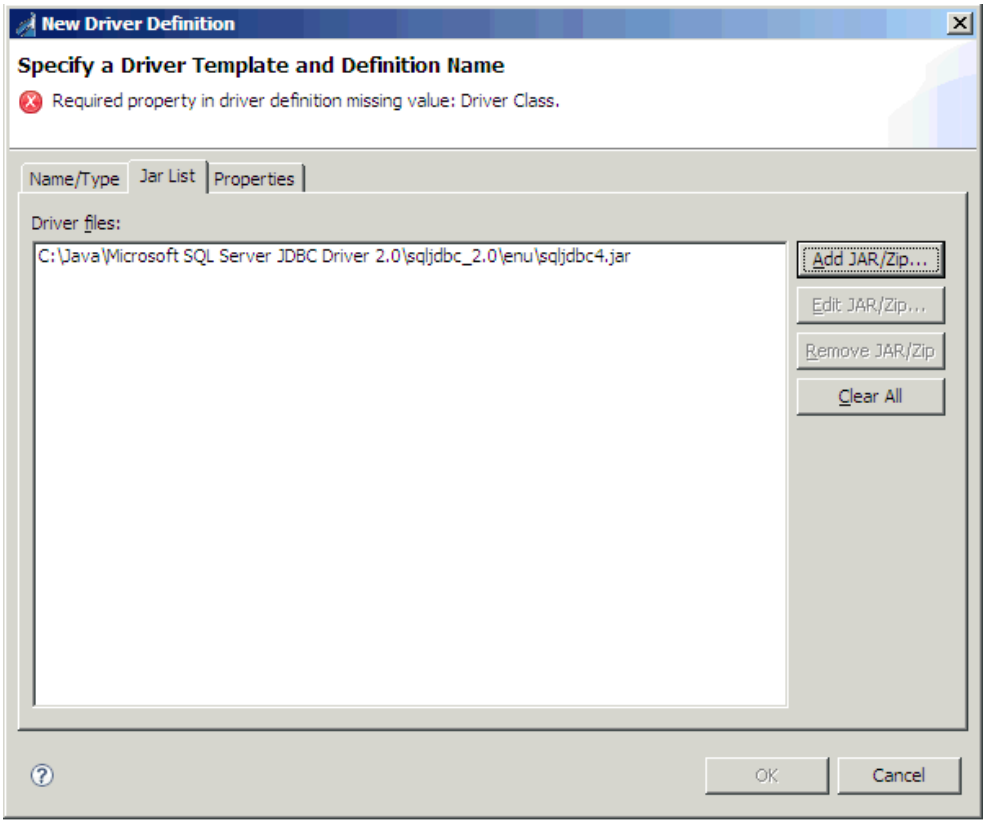
Procedure: How to Use the Generic JDBC Driver Definition

To use the generic JDBC driver definition:

1. Using the New Driver Definition wizard, select the *Generic JDBC Driver*, as shown in the following image.



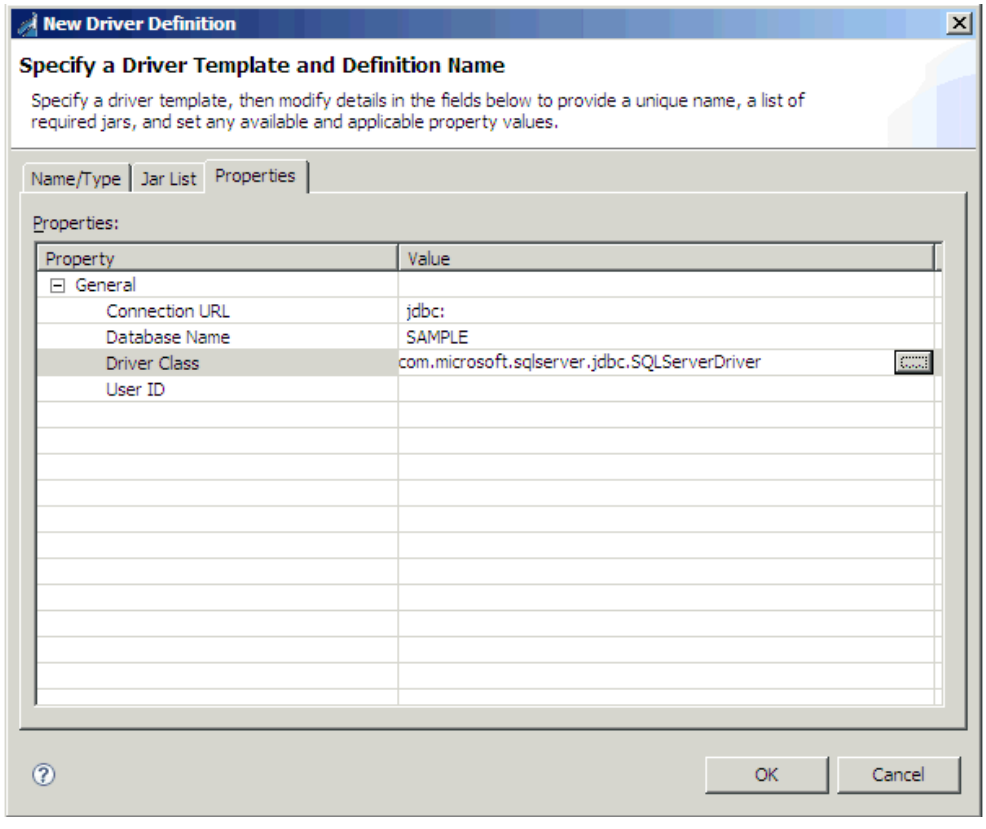
2. Add the sqljdbc4.jar file, as shown in the following image.



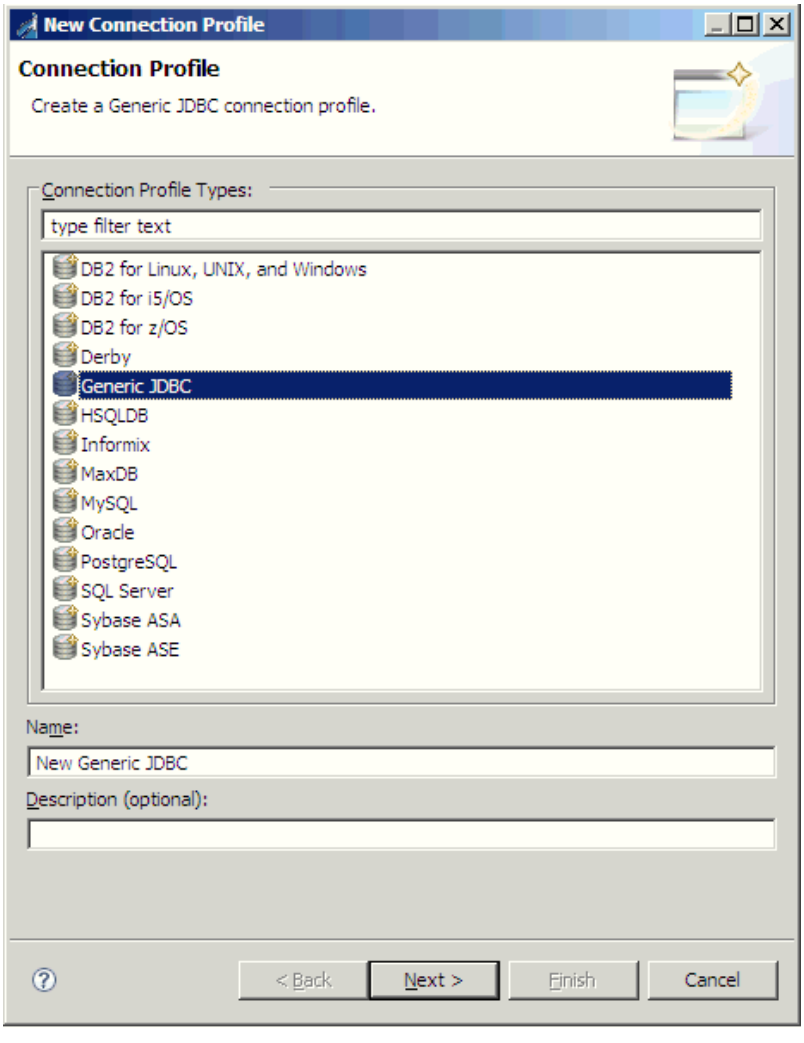
3. In the Properties tab, set the Driver Class property to:

   `com.microsoft.sqlserver.jdbc.SQLServerDriver`

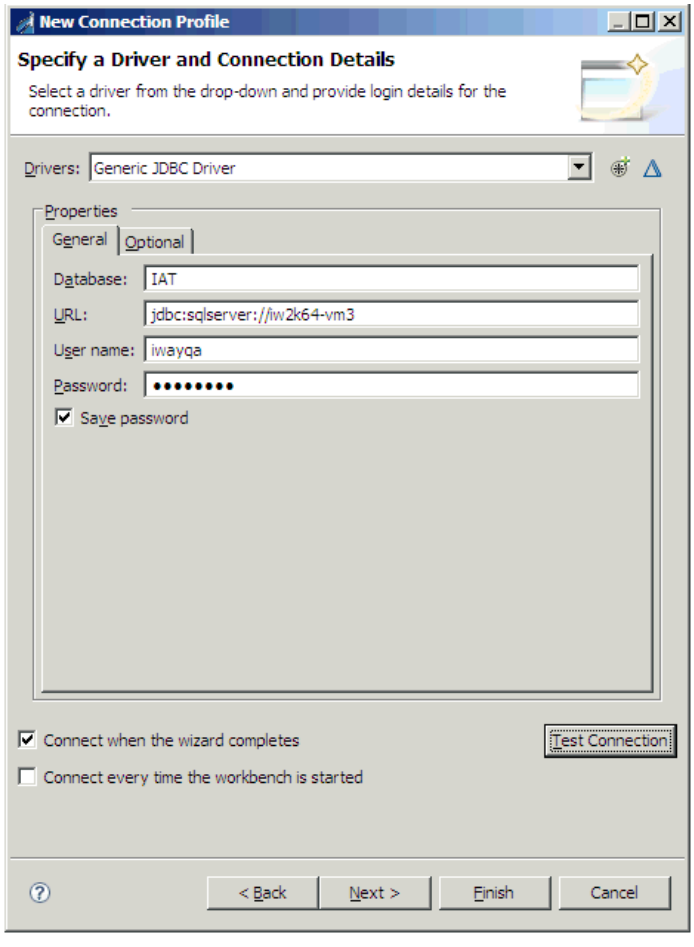
For example:


New Driver Definition
Specify a Driver Template and Definition Name
Specify a driver template, then modify details in the fields below to provide a unique name, a list of required jars, and set any available and applicable property values.
Name/Type
Jar List
Properties
Properties:
Property
Value
General
Connection URL
jdbc:
Database Name
SAMPLE
Driver Class
com.microsoft.sqlserver.jdbc.SQLServerDriver
User ID
OK
Cancel

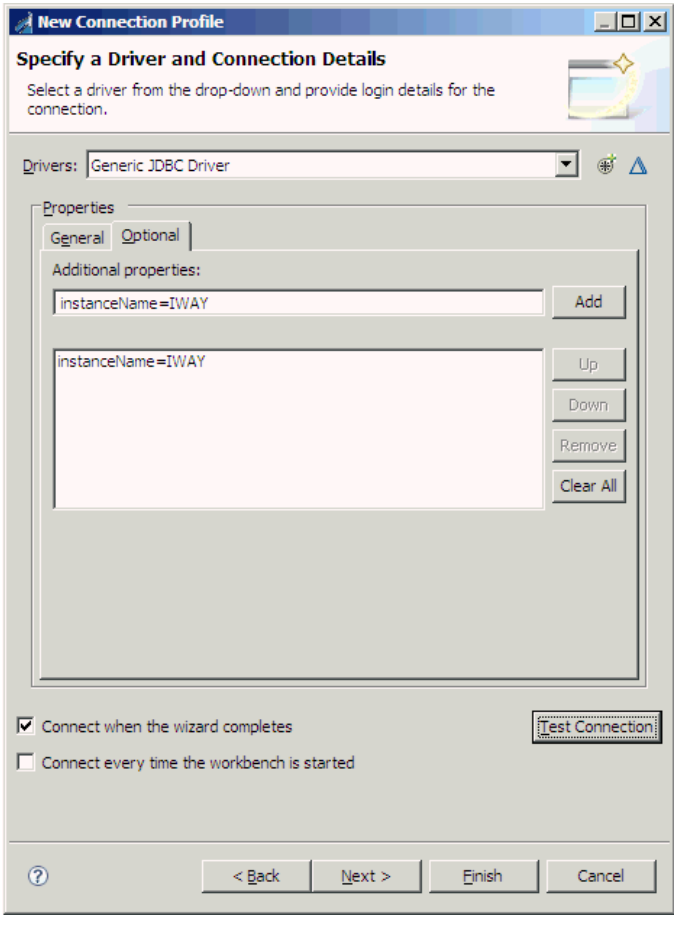
4. To use the definition, select *Generic JDBC* in the New Connection Profile wizard, as shown in the following image.



5. Enter the connection information for the database, as shown in the following image.



6. If any properties are required, such as the instance name, enter them using the Optional tab in the Properties section, as shown in the following image.



7. Click the *Test Connection* button to verify that your connection is working.

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

TIBCO, the TIBCO logo, the TIBCO O logo, FOCUS, iWay, Omni-Gen, Omni-HealthData, and WebFOCUS are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle Corporation and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.