

HP Connect-It

Software Version: 9.51

Connector Guide

Document Release Date: November 2012

Software Release Date: November 2012



Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

© Copyright 1994-2012 Hewlett-Packard Development Company, L.P.

Acknowledgements

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

This product includes code licensed from RSA Data Security.

This product includes ANTLR.

Support

Visit the HP Software Support Online web site at:

<http://www.hp.com/go/hpsoftwaresupport>

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Contents

Connector Guide.....	1
Contents.....	4
The Connector Guide.....	12
Introducing Connectors.....	12
Configuring Connectors.....	13
Launching a Connector's Configuration Wizard.....	13
Selecting a Connection Type.....	14
Naming and Describing a Connector.....	14
Configuring Database Connections.....	15
Selecting Files or Folders.....	16
Turning on Advanced Configuration.....	18
Configuring Advanced Mode Settings.....	18
Configuring the Reconnection Parameters.....	19
Determining Server Delay.....	20
Configuring Schedule Pointers.....	22
Managing Transactions.....	24
Configuring the Cache.....	24
Configuring Additional Advanced Options.....	26
Defining Document Types.....	28
Choosing a DTD or XSD.....	29
Configuring the JVM.....	32
Enabling Parallel Consumption.....	32
Configuring the Proxy Server.....	32
Configuring SSL.....	33
Connector Directives.....	38
Defining Production Directives.....	38
WHERE and ORDERBY Clauses.....	39
Defining Consumption Directives.....	40

Reconciliation	41
Configuring Basic Reconciliation Settings	41
Configuring Advanced Reconciliation Settings	46
Using Reconciliation Scripts (for the Asset Manager connector only)	47
Two-pass Reconciliation	49
RESTful Exposure	51
Configuring Restful Exposure	52
Consumed Document Types Design	55
Runtime	58
Discovering What Collections are Available	58
Retrieving the Entries in a Collection	59
Managing Entries	61
OpenSearch Descriptor	65
Resources Metadata	66
Asset Manager Exposure	67
Hewlett-Packard Connectors	68
HP Asset Manager Connector	68
Configuring the HP Asset Manager Connector	72
HP Business Service Management Dashboard / HP Universal CMDB Connectors	75
HP Client Automation Inventory Manager Connector	75
Configuring the HP Client Automation Inventory Manager Connector	76
HP Client Automation Management Portal Connector	76
Configuring the HP Client Automation Management Portal Connector	76
HP Client Automation Service Events Connector	77
Configuring the HP Client Automation Service Events Connector	77
HP Client Automation Usage Manager Connector	78
Configuring the HP Client Automation Usage Manager Connector	78
HP Cloud Service Automation Connector	78
Configuring the HP Cloud Service Automation Connector	79
HP Discovery and Dependency Mapping Inventory Connector	79
Configuring the HP Discovery and Dependency Mapping Inventory Connector	79
OpenView Configuration Management Connectors	80

Configuring OpenView Configuration Management Connectors.....	81
HP DecisionCenter Connector.....	82
Configuring the HP DecisionCenter Connector.....	82
HP Network Node Manager i-series Web Service Connector.....	82
HP Operations Manager for Unix Connector.....	84
Configuring the HP Operations Manager for Unix Connector.....	85
HP Project and Portfolio Management Center Connector.....	85
Configuring HP Project the Project and Portfolio Management Center Connector.....	85
HP Service Desk (Inbound) Connector.....	86
Configuring the HP Service Desk (Inbound) Connector.....	86
HP Service Desk (Outbound) Connector.....	88
Configuring the HP Service Desk (Outbound) Connector.....	88
HP ServiceCenter and Service Manager Connector.....	88
Configuring the HP ServiceCenter and Service Manager Connector.....	90
Configuring Advanced Mode Settings for the HP ServiceCenter and Service Manager Connector.....	91
Additional Information for the ServiceCenter and Service Manager Connector.....	93
HP ServiceCenter and Service Manager Web Service Connector.....	101
Configuring the HP ServiceCenter and Service Manager Web Service Connector.....	102
HP Service Anywhere Connector.....	103
Configuring the HP Service Anywhere Connector.....	103
HP SCAuto Listening Connector.....	104
Configuring the SCAuto Listening Connector.....	105
HP Universal CMDB Connector 8.....	108
Configuring the HP Universal CMDB Connector 8.....	108
HP Universal CMDB Connector.....	110
Configuring the HP Universal CMDB Connector.....	110
HP Universal CMDB Connector (XML).....	113
Configuring the HP Universal CMDB Connector (XML)- Read.....	114
Configuring the HP Universal CMDB Connector (XML)- Write.....	117
Consumption Directive of the uCMDB (XML) Connector (write) - FTP Mode.....	120
Additional Information - uCMDB (XML) Connector.....	120

Published Document Types.....	123
Inventory Connectors.....	124
Altiris Connectors.....	124
Configuring the Altiris Connector.....	125
Aperture VISTA Connectors.....	126
Configuring Aperture VISTA Web Service Connectors.....	126
CA Unicenter DSM 11 Connector.....	127
Configuring the CA Unicenter DSM 11 Connector.....	127
LANDesk for Inventory Connectors.....	128
Configuring LANDesk for Inventory Connectors.....	128
Production Directives of the LANDesk Connector.....	128
LANDesk Software Distribution Connector.....	129
Configuring the LANDesk Software Distribution Connector.....	129
Microsoft System Center Configuration Manager (SCCM) Connectors.....	130
Configuring Microsoft System Center Configuration Manager (SCCM) Connectors...	131
Tally Systems TS.Census 3 Connector.....	131
Configuring the Tally Systems TS.Census 3 Connector.....	131
Tivoli Connectors.....	132
Tivoli Inventory Connector (version 4.0).....	132
Tivoli CM for Inventory 4.2 Connector.....	132
Tivoli CM for Software Distribution Status 4.2 Connector.....	135
Tivoli CM for Software Distribution 4.2 Connector.....	135
Application Connectors.....	136
Action Request System Connector.....	136
Configuring the Action Request System Connector.....	138
IBM Websphere MQ Connector.....	139
Configuring the Websphere MQ Connector (read).....	140
Configuring the Websphere MQ Connector (write).....	141
Production Directives of the Websphere MQ Connector.....	142
Consumption Directives of the Websphere MQ Connector.....	143
Additional Information About the Websphere MQ Connector.....	145
Lotus Notes Connector.....	146

Configuring the Lotus Notes Connector.....	147
Production Directives of the Lotus Notes Connector.....	150
Consumption Directives of the Lotus Notes Connector.....	150
NT Security Connector.....	151
Configuring the NT Security Connector.....	152
Tivoli Enterprise Console Connector (sending).....	154
Configuring the Tivoli Enterprise Console Connector (sending).....	154
Tivoli Enterprise Console Connector (receiving).....	156
Configuring the Tivoli Enterprise Console Connector (receiving).....	157
JMS for IBM WebSphere MQ Connector.....	162
Configuring the JMS for IBM WebSphere MQ Connector.....	163
JMS for IBM WebSphere MQ Connector Published and Consumed Documents.....	164
ERP Connectors.....	165
SAP ALE Connector.....	166
Configuring the SAP ALE Connector.....	166
SAP BAPI Connector.....	166
Configuring the SAP BAPI Connector.....	167
SAP IDoc Connector.....	168
Configuring the SAP IDoc Connector.....	169
SAP Web Service Connector.....	170
Configuring the SAP Web Service Connector.....	170
Protocol Connectors.....	172
Command-line Connector.....	172
Configuring the Command-line Connector.....	172
Command-line Connector Published Document-Types.....	173
BIOS Commands.....	174
Command-line Connector Consumption Directives.....	174
Database Connector.....	174
Configuring the Database Connector.....	180
Delimited-text Connector.....	182
Configuring the Delimited-text Connector (read).....	183
Configuring the Delimited-text Connector (write).....	187

E-mail Connectors.....	195
Configuring the E-mail connector (fetching).....	196
Document Types Produced by the E-mail (fetching) Connector.....	199
Configuring the E-mail Connector (sending).....	201
Document Types Produced by the E-mail (sending) Connector.....	203
HTTP Service Connector.....	204
Configuring the HTTP Service Connector.....	206
JMS Connector.....	211
Configuring the JMS Connector.....	211
LDAP Connector.....	213
Configuring the LDAP Connector.....	214
LDAP Connector Consumption and Production Directives.....	216
Additional Information for the LDAP Connector.....	219
Web Service Connectors.....	222
RESTful (Deprecated) Client Connector.....	222
RESTful Client Connector.....	223
SOAP Connector.....	228
XML Connector.....	234
Configuring the XML Connector (read).....	234
Configuring the XML Connector (write).....	235
Consumption Directive of the XML Connector (write) - FTP mode.....	235
Additional Information for the XML Connector.....	235
XML Listening Connector.....	235
Configuring the XML Listening Connector.....	237
Using the XML Listening Connector.....	238
XML Listening Connector Example- Tomcat Web Server.....	242
Internal Tools.....	244
Status Report Connector.....	244
Configuring the Status Report Connector.....	246
Out-of-Box Scenarios.....	248
HP Asset Manager Connector Scenarios.....	248

HP Business Service Management Dashboard / HP Universal CMDB Connector Scenarios.....	250
HP Cloud Service Automation Connector Scenarios.....	251
HP Discovery and Dependency Mapping Inventory Connector Scenarios.....	251
Configuration Management 5.1 / Client Automation 7.x Scenarios.....	257
Customizing the Asset Manager Database.....	261
HP DecisionCenter Connector Scenarios.....	264
HP Network Node Manager i-series Web Service Scenarios.....	265
HP Operations Manager Scenarios.....	265
HP Project and Portfolio Management Center Scenarios.....	267
HP Service Desk (Outbound) Connector Scenarios.....	268
HP ServiceCenter and Service Manager Connector Scenarios.....	268
HP ServiceCenter and Service Manager Web Service Scenarios.....	270
HP SCAuto Listening Connector Scenarios.....	270
HP Service Anywhere Connector Scenarios.....	271
HP ServiceCenter uCMDB Scenarios.....	272
Altiris Scenarios.....	272
Aperture Vista Scenarios.....	273
CA Unicenter DSM Scenarios.....	273
LANDesk Scenarios.....	273
Microsoft SCCM Scenarios.....	274
TS Census Scenarios.....	275
Tivoli Scenarios.....	276
Action Request System Connector Scenarios.....	277
HP Network Node Manager Scenarios.....	277
IBM Websphere MQ Connector Scenarios.....	280
Lotus Notes Connector Scenarios.....	282
NT Security Connector Scenarios.....	282
SAP BAPI and Web Service scenarios.....	282
SAP IDOC Scenarios.....	283
Email Connectors Scenarios.....	284
HTTP Service Connector Scenarios.....	284

LDAP Connector Scenarios.....	285
RESTful Client Connector Scenarios.....	286
SOAP Scenarios.....	286
AQL Queries.....	288
Recommendations for Writing AQL Queries.....	289
AQL Sorts and Index.....	295
The AQL Query Editor.....	296
AQL Syntax.....	300
AQL Clauses.....	304
AQL Function References.....	309
Appendix A Sample Scenarios of HTTP Service Connector.....	314
Scenarios.....	314
Folders.....	314
Scenario description.....	314
Index.....	320

Chapter 1

The Connector Guide

The Connector Guide explains how to configure and use all connectors and the out-of-box scenarios included with Connect-It.

Introducing Connectors

Connectors communicate with external applications and enable these applications to exchange data. Connectors can be divided into three categories:

- **Base connectors**
These connectors are available to all users of the Scenario Builder.
- **Optional connectors**
These connectors are provided with the Scenario Builder. Depending on your license, however, you may or may not have the right to use them.
- **Additional connectors**
These connectors are not provided with the Scenario Builder and must be requested from Hewlett-Packard.

Compatibility

For compatibility-related information, refer to the Compatibility Matrix available on the Hewlett-Packard software support site:

www.hp.com/go/hpsupport

You will require a valid login and password to access this site.

Obsolete Connectors

Certain connectors have been made obsolete in this version of Connect-It. For a list of obsolete connectors, see the Release Notes.

Chapter 2

Configuring Connectors

Configuring a connector in a scenario enables you to:

- Name and describe it.
- Specify the connection parameters that enable it to communicate with an external application (database, messaging system, queue manager, etc.).
- Specify a multitude of options enabling it to process the document types it publishes, produces or consumes.

Configuration options

The following options are available for connectors. The options differ depending on what type of connector you are configuring:


Connector options

Standard Configuration	
Selecting a Connection Type	
Naming and Describing a Connector	
Configuring Database Connections	
Selecting Files or Folders	
Turning on Advanced Configuration	

Advanced Configuration	
Configuring the Reconnection Parameters	Defining Document Types
Determining Server Delay	Choosing a DTD or XSD
Using Time Zones	Configuring the JVM
Configuring Schedule Pointers	Enabling Parallel Consumption
Managing Transactions	Configuring the Proxy Server
Configuring the Cache	Configuring SSL
Configuring Additional Advanced Options	

Launching a Connector's Configuration Wizard

To launch a connector's configuration wizard:

- If the connector is not in the Scenario diagram pane.
 - a. Double-click the connector in the Toolbox.
 - b. Drag and drop the connector from the Toolbox into the Scenario diagram.
- If the connector is already in the Scenario diagram pane.
 - a. Select the connector and choose the **Tools > Configure** menu.
 - b. Select the connector and press **F2** on the keyboard.
 - c. Select the connector, right-click and choose **Configure connector** from the shortcut menu.
 - d. Click the configure connector button .

Selecting a Connection Type

This page enables you to select the type of connection for a database application. You can choose from:

- An ODBC data source. In this case, the ODBC layers are used.
- A native Oracle connection.
- A native Sybase connection.
- A native MySQL connection.
- A native DB2 connection.

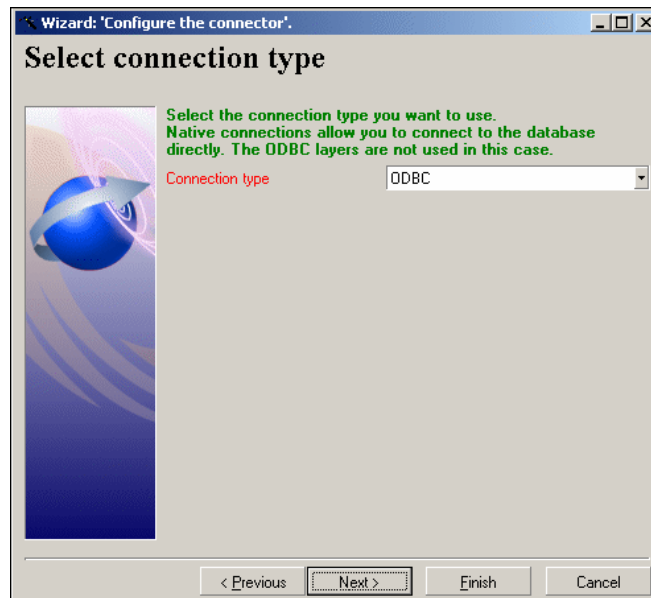


Figure 2-1: Selecting a connection type

Naming and Describing a Connector

In the **Name** field, enter a descriptive name for the connector. By default, the value of the field is the name of the connector as it appears in the Toolbox. If you have another connector of the same kind in the same scenario, Connect-It will add a number to the connector name: NameOfConnector, NameOfConnector1, NameOfConnector2, etc.

This name identifies the connector in the scenario, and each connector displays its name in the Scenario diagram.

Caution: If you use the same connector more than one time in a scenario, you need to assign them each a different name. Example: a different name for each Asset Manager connector in a scenario to transfer data between two Asset Manager databases.

In the **Description** field, enter text to describe the connector's role in the scenario. (Example: Reading or write such and such a data source.) This field is not mandatory.

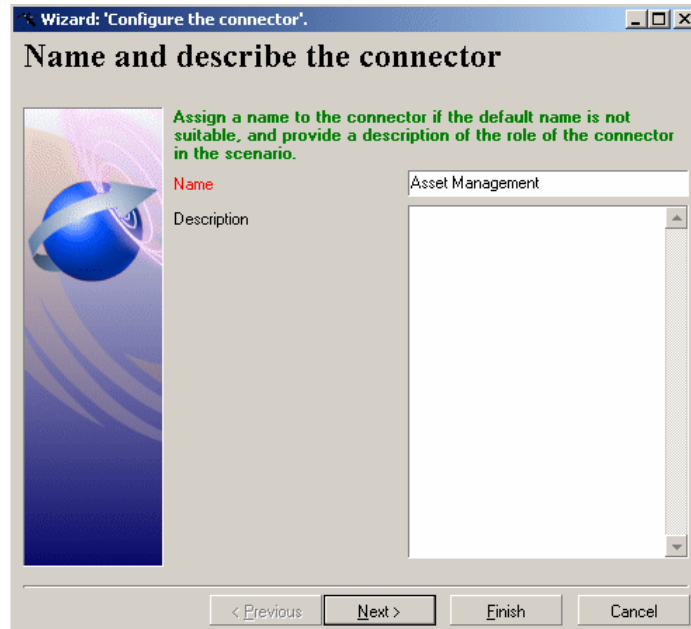


Figure 2-2: Naming and describing the connector

Configuring Database Connections

The configuration screen you see depends on the [connection type](#) that you selected. For your connection type, configure the relevant settings:

Connection Type	ODBC	Oracle	Sybase	MySQL	DB2
User Enter the login that enables you to access your database.	Yes	Yes	Yes	Yes	Yes
Password Enter the password associated with the login. This password is not visible when being entered and is stored in an encrypted format.	Yes	Yes	Yes	Yes	Yes
Table owner Enter the name of the table owner if different from the value you entered in the User box. Separate multiple owners with a comma. For example: owner1, owner2.	Yes	Yes	Yes	No	Yes

Connection Type	ODBC	Oracle	Sybase	MySQL	DB2
<p>Test This button enables you to test your connection. To test your connection:</p> <ol style="list-style-type: none"> 1. Enter your connection parameters. 2. Click Test. The Test the connection window appears and tells you whether the connection succeeded or failed. If it failed, there will be messages explaining the cause. 3. Click Close to return to the configuration wizard. 	Yes	Yes	Yes	Yes	Yes
<p>Server Enter the name of the server. For the Oracle database, you can use the alias which is defined in file <code>tnsname.ora</code>.</p>	No	Yes	Yes	Yes	No
<p>Server database Indicate the name of the database to which you want to connect.</p>	No	No	Yes	Yes	Yes
<p>ODBC data source Indicate the name of the ODBC connection by selecting a value from the drop-down list. The drop-down list contains the ODBC connections available on your computer. Because this list cannot be modified, the ODBC data source must already be configured in the ODBC Administrator before creating your connector.</p>	Yes	No	No	No	Yes

Selecting Files or Folders

Use this screen to specify the locations of the files and folders to be read.

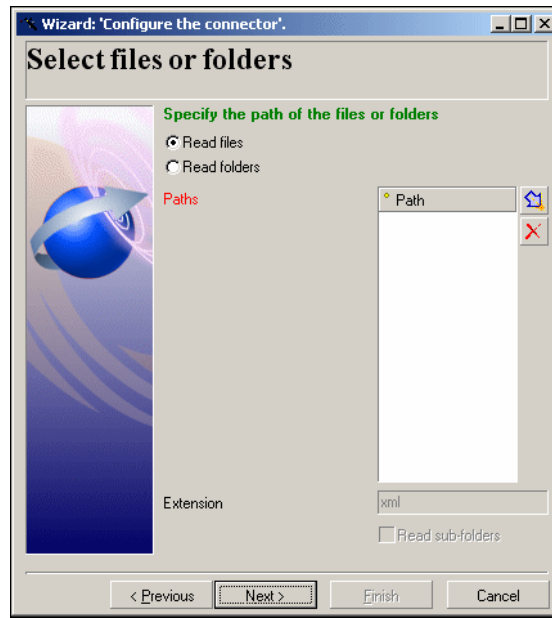


Figure 2-3: Select files or folders

If you select the **Read folders** option, you must specify:



- The path of the folder
- The file extension name.
The default value of the field depends on the connector: .xml, .gz, .xsf. The extension name is free. You can define multiple extensions by making a list separated by semi-colons.

You must also select the **Read sub-folders** option.

Note: When you indicate the path of a file or folder located in a folder associated with a network drive on your computer, you must never indicate the letter of this drive; only indicate the folder associated with this letter. Using a network drive in a path will stop the scenario from working properly when it is associated with a service.

Read files

To define multiple files to be read, proceed as follows:

1. Click .
2. In the field that appears, click  and then select the appropriate file.
3. Repeat this operation as many times as required.



If you select this option, the wizard will ask you to enter the path or paths of the XML files on your computer or your network.

Read folders

If you select this option, the wizard will ask you to populate the following fields:

- **Folder name:** Enter the path of the folder that contains your files.
- **Extension:** Indicate the file name extension that the connector must read. For example, .xml.

To define multiple folders to be read, proceed as follows:

1. Click .
2. In the field that appears, click  and then select the appropriate folder.
3. Repeat this operation as many times as necessary.

Read the sub-folders

If you select this option, the connector will also read the files in the sub-folders of the selected folder.

Turning on Advanced Configuration

The number of pages displayed in the configuration wizard depends on this option.

To enable this option, click the  icon. The following pages are displayed in advanced mode:

- Advanced configuration
- Configure the reconnection parameters
- Determine server delay
- Configure schedule pointers
- Manage transactions
- Configure the cache
- Define document types
- Parallelize consumption

Configuring Advanced Mode Settings

This page enables you to specify the advanced configuration parameters.

SQL92 supported

By default, this option is selected. It indicates the SQL queries sent to the database by the connector respect the SQL92 syntax. If this database does not support this syntax, you must clear this option. Example: Informix.

Archival

When a connector uses post-processing actions, you can archive the files that are moved. Archival is enabled when the **Move it to folder** option is selected. This occurs both in the event of a failure or upon success. Archival is performed for a given period of time or for a given number of versions of a file you wish to keep. Archival is enabled when you select the **Keep archive** option. Each archived file is suffixed according to the defined configuration.

Archival settings

Configuring the archival means defining one of the following:

- A time limit to keep a file (time, day, week, month, year). This duration is defined in the **Date** frame.

- A maximum number of versions of a file to keep. Populate the **Number of versions of a file to keep** field.
- Indefinite archival. Select the **Keep all** option.

Archival using the date

When you choose to archive using the date, each file name is suffixed with the date. You must specify the date format used. The following date formats are available:

```
yyyy-MM-dd-HH-mm-ss
yyyyMMddHHmms
dd-mm-yyyy hh:nn:ss
ddmmyyyhhnss
```

For example, ex_2003-09-02-16-20-05.

Note: Only the suffixed date in the file name is taken into account and not the creation date of the file itself.

Archival using a number

When you choose to archive using a number instead of the date, each file will be suffixed with a number. You must specify the way in which the files will be incremented. This incrementation is defined in the **Numbering** frame.

Incrementation is either by:

- Ascending sort: Each file keep is added to the end of the queue. For example, for the files ex_1, ex_2, and ex_3, the most recent file is ex_3.
- Descending sort: Each file keep is added to the start of the queue. For example, for the files ex_1, ex_2, and ex_3, the most recent file is ex_1.

The number of digits used to suffix the file name and the separator that is used can be configured.

Configuring the Reconnection Parameters

This screen enables you to configure your reconnection parameters in case the initial connection is lost.

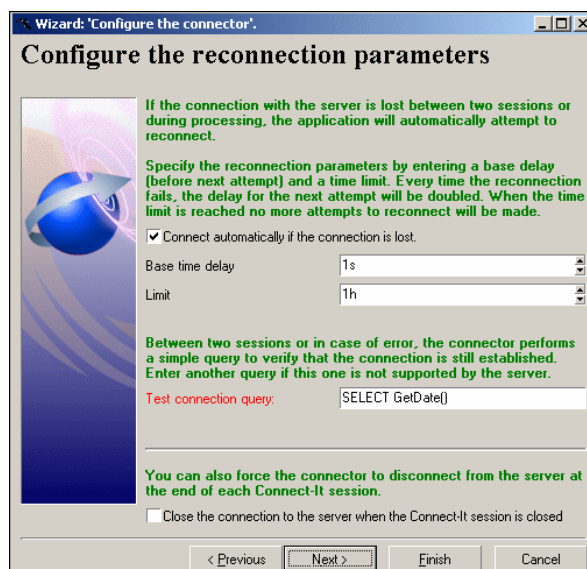


Figure 2-4: Configuring the reconnection parameters**Connect automatically if the connection is lost**

If you select this option, you must specify how Connect-It should reconnect in case the connection to the server fails or is interrupted. If it is a database, then it's the server on which this database is located.

Note: *This option is selected by default. If the database server does not support reconnections, you must clear this option.*

These are the following reconnection parameters:

Base time delay

The base time delay enables you to specify how long Connect-It should wait before trying to reconnect to the server. If reconnection is not successful, Connect-It doubles this period before trying again. Example: With a base time delay of 2 seconds, the second attempt to reconnect will take place after 4 seconds, the third after 8 seconds, etc.

Time limit

The time limit enables you to specify the maximum length of time after which Connect-It should give up trying to reconnect to the server.

Note:

After successful reconnection:

While writing data, the document whose processing was interrupted is processed again, and the session can continue normally.

While reading data, the interrupted session is restarted in full.

Test connection query

Each time you open a new session, or in case of a processing error, a query is sent to the server in order to verify that the database connection is still open. The request must be supported by the server, and must be efficient, in order for optimal document processing. The query emitted by default is **SELECT GetDate()**. If this query is not supported by the database, enter a valid query.

Note: *Each time it is open, the connector emits the test connection query. If this query is not supported by the server, a warning message will tell you. In this case, automatic reconnection is not possible.*

When losing a connection is not an issue, you can deactivate this option so that you don't obtain a warning message when opening the connector. Example: An ODBC connection to an Excel file.

Closing the connection to the server when the Connect-It session is closed

Do not select this option unless told to do so by Connect-It support technicians.

Determining Server Delay

This page enables you to determine the delay with the server to which the connector connects when it reads or receives data.

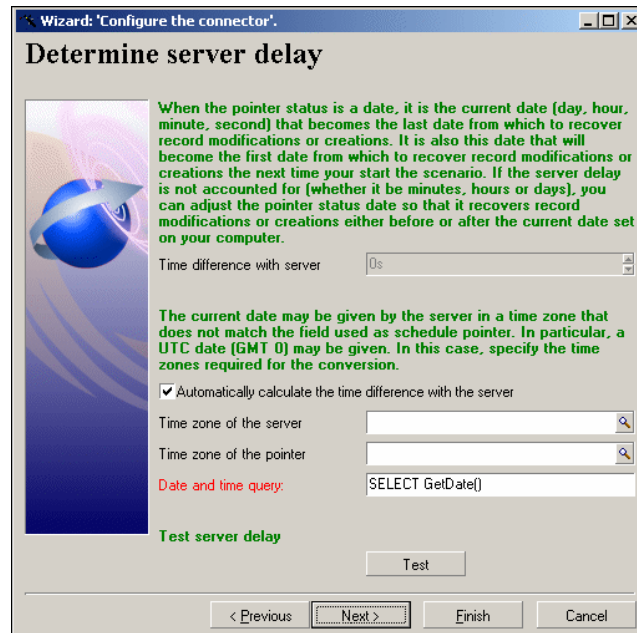


Figure 2-5: Determining the server delay

Automatically calculate the time difference with the server selected

This option is selected by default. In this case, Connect-It queries the server at the start of each session to automatically adjust the time different between Connect-It's clock and the server's clock.

Automatically calculate the time difference with the server cleared

If you do not select the **Automatically calculate the time difference with the server** option, you must manually enter the time difference. You must manually enter a value when the connector cannot automatically determine this time difference.

Date and time query

This field enables you to specify a query used to return the time of the server in order to compensate the time used locally by the scheduling pointers.

A query is performed:


- when the connector is opened
- each time a session is opened, in order to update the delay between the server and local time

Using Time Zones

You must use the **Time zone of the server** and **Time zone of the pointer** fields when the date returned by the server does not correspond to the time zone expected by the external application to perform filtering operations. For example:

- The server returns a UTC timestamp (Universal Time Coordinated) when, in fact, the external application saves this timestamp under a different time zone.
- The external application returns the current timestamps in a given time zone when, in fact, the operation to filter these timestamps corresponds to the time zone defined in Connect-It server's 'Date and time' parameters.

To specify a time zone:

1. Verify that the **Automatically calculate the time difference with the server** option is selected.
2. Click .
3. Select a time zone in the window that appears.

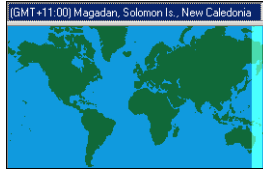


Figure 2-6: Time zones


4. Select the **Apply daylight savings time** option if your connector accounts for daylight savings time.
5. Click **OK**.

To display the current timestamp of the external application:

To the right of the title **Test time difference with server**, click **Test**. This action triggers the following operations, which you can view in a contextual window:

- Connection to the external application.
- Recovery and display of the external application's current timestamp.
- Display of the current timestamp after having taken into account the time server delay (if time zones have been specified).

To test the time zone parameters:

1. Modify a record in the external application.
2. Create a document type that corresponds to this record.
3. Apply a filter on the field indicating the record's last modification date. The value of this field in the syntax of the filter must correspond to the external application's dates and times, obtained using the **Test** button.
4. In the edit window of the document types produced by the connector, click  to verify that the modified record has been recovered.

Configuring Schedule Pointers

This screen enables you to configure the schedule pointers used by the connectors.

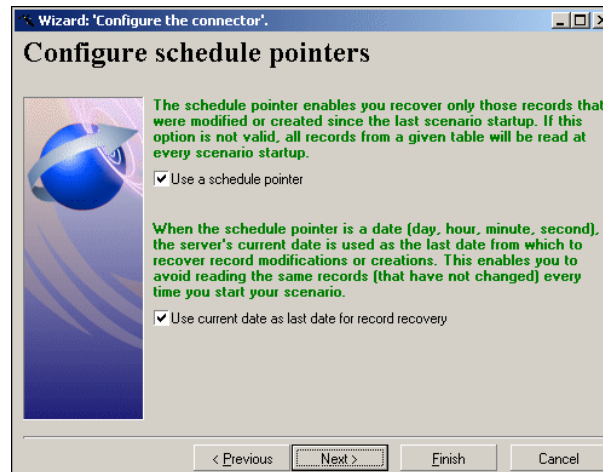


Figure 2-7: Configuring schedule pointers

When a scenario is launched in scheduled mode, the status of a document-type's pointer tells the connector to process only those documents that have not been processed since the last session.

For example, in its last session, an Asset Manager connector took until 9h00mn00s to read all the records in the Assets table. Thus, in the next session, it will read all the records created after 9h00mn00s.

Use a schedule pointer

This option is selected by default. When this option is selected, the connector only processes the data that was modified since the last session. For example, a database-type connector only processes the records that were created or updated since the last session.

If this option is not selected, the connector processes all the data in the source application. For example, if you use a database-type connector to save a copy of the database, the record-modification date is not used.

Use current date as last date for record recovery

This option is selected by default. To illustrate this option, we'll use the case of a scheduler that wakes up a connector whose job is to read source data every hour on the hour. The last session was at 9h00m00s, and this session is at 10h00m00s.

Use current date as last date for record recovery option selected

In this case, during the new session, the connector only processes the data whose modification date is:

- Greater than or equal to the date of the last session.
Example: all data whose modification date is greater than or equal to 9h00m00s. Data modified at exactly 9h00m00s will thus be processed by the connector.
- Less than the current date (adjusted to the server delay if this can be automatically calculated).
Example: all data whose modification date is less than 10h00m00s. All the data modified after 10h00m00s is not processed by the connector.

Use current date as last date for record recovery option cleared

During the new session, the connector only processes the data whose modification date is greater than or equal to the date of the last session. For example, all data modification dates greater than or equal to 9h00m00s. A record created exactly 9h00m00s will be processed by the connector. All the data created after 10h00m00s will be processed by the connector.

This is useful when the connector cannot automatically determine the time difference with the server.

Note: The date of the schedule pointer uses the format defined for the local computer. This format is saved in the `[scenario name].ini` file of the folder of the current scenario.

Managing Transactions

This screen enables you to define the mode you use to transmit documents used by the connector.

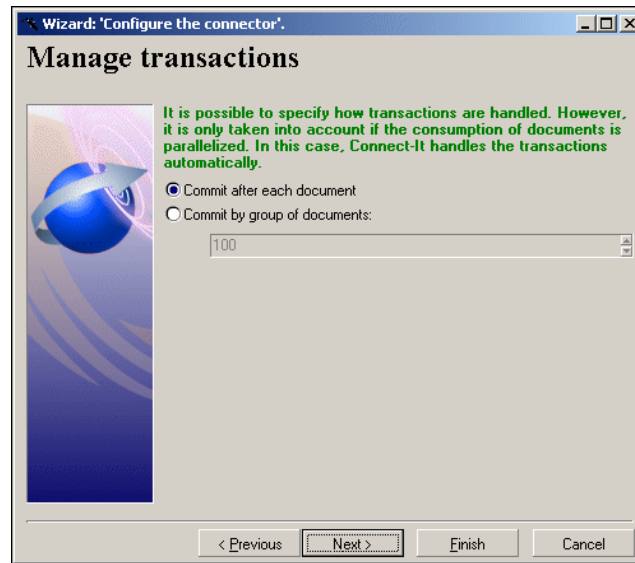


Figure 2-8: Managing transactions

The connector can:

- **Commit after each document:** after each document is processed, the Commit command is performed. This validates the insert, update or delete queries emitted while processing the document.
- **Commit by group of documents:** after processing the indicated number of documents, the Commit command is performed. This validates the insert, update or delete queries emitted while processing the documents. The number of documents to process is set to 100 by default.

Note: If you're using this mode and an error appears while processing documents, Connect-It will insert one document at a time until it comes to the erroneous document. For optimal results, we recommend using the **Commit by group of documents** option.

Configuring the Cache

This screen enables you to use a cache file containing the description of document types published by the connector. The file extension for such a file is `.cch`.

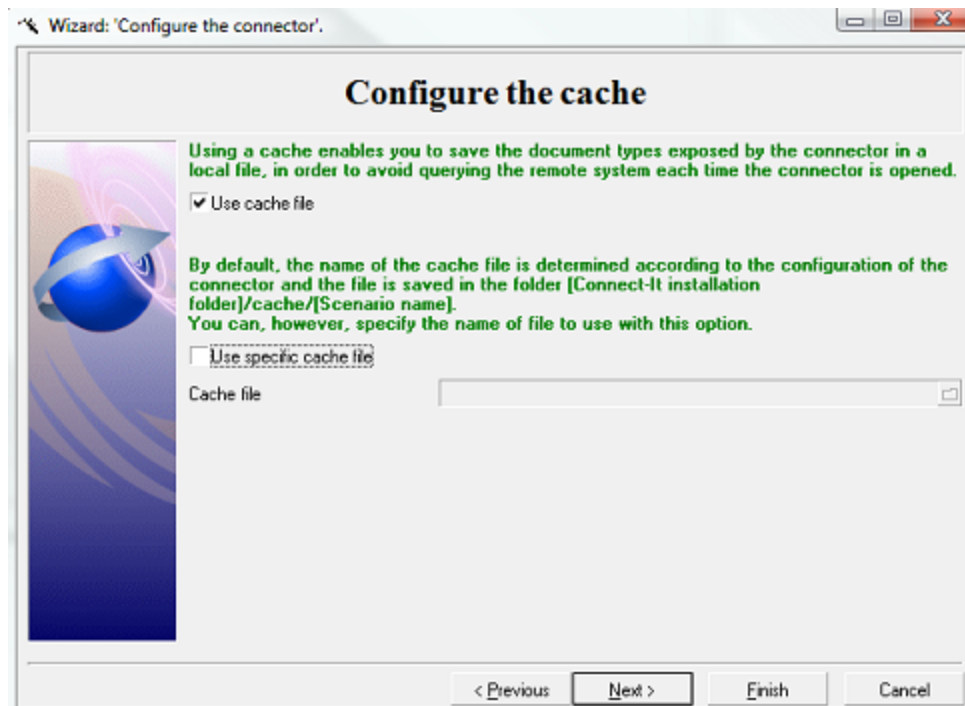


Figure 2-9: Configuring the cache

In the case where you use an application that is not located on your local network, a cache enables you to quickly work with the connector without having to wait for the retrieval of the document types.

Note: We recommend that you use a cache file to be able to access data faster and to enable you to work offline.

You can choose from two options:

- **Use cache file**
If you select this option, a cache file is created in a folder named after your scenario in the cache folder of the Connect-It installation folder. The name of each cache file CCH is unique. In addition, the cache files of each instance of the same connector have different names. Generally, the cache file takes its name from the name of the server.
- **Use specific cache file**
If you select this option, you need to indicate the path and file name of the cache that you want to create or reuse. If this cache file does not exist, Connect-It will create it when you open your connector for the first time. If this file already exists Connect-It suggests that you either overwrite it or rename your new cache file.

Note: Use this option when you want to use a cache file that was created in a previous scenario.

The cache must be synchronized each time that the external application's structure is modified: creation of a new table for database applications, creation of an output event for ServiceCenter, etc.

To synchronize a connector's cache:


1. Select the connector in the Scenario diagram.
2. Right-click.


3. Select **Cache > Synchronize** from the shortcut menu (**Tool > Cache > Synchronize the cache**).

Configuring Additional Advanced Options

This editable zone enables you to enter advanced options that are different depending on your connector.

To enter an advanced option:

1. Click .
2. Enter the name of the option in the **Name** column.
3. Enter the value of the option in the **Value** column.

To modify the name or the value of an option, double-click directly on this name or value and enter a new one. To delete an option, select the row corresponding to this option and click .

Advanced options for the inventory and database connectors

The following table presents the advanced options that can be entered for the Inventory and Database connectors.

Caution: *The default values of the advanced options should not be modified unless HP support or the accompanying documentation specifies otherwise.*

Inventory and database connectors - Advanced configuration options

Name	Default value	Description	Data type	Examples
StmtCache	30	Cache size	Long integer Number of instances	Enter 0 to: <ul style="list-style-type: none"> • DB2 • Microsoft Visual Studio
ConstAsText	0	Constant format using the text format.	Boolean: 0 = False 1 = True	Enter 1 to: <ul style="list-style-type: none"> • ODBC Novell driver
NoPrepare	0	Prevents use of dynamic SQL.	Boolean: 0 = False 1 = True	Enter 1 to: <ul style="list-style-type: none"> • ODBC Novell driver
AutoCommit	1	Indicates the automatic mode (1) or manual mode (0) of the commit command.	Boolean: 0 = False 1 = True	Enter 0 to: ODBC Novell driver
RowSetSize	1	Enables to specify the number of rows to return.	Boolean: 0 = False	Enter 0 to: ODBC Novell driver

Name	Default value	Description	Data type	Examples
			1 = True	
ExtendedFetch	1	Uses extended fetching.	Boolean: 0 = False 1 = True	Enter 0 to: ODBC Novell driver
AddIndexDesc	1	Enables the autodescription of indexes.	Boolean: 0 = False 1 = True	Enter 0 to: ODBC Novell driver
FetchingArray- Size	Depends on the database engine.	Enables to specify the number of rows to link.	Long integer	Enter 1 to: ODBC Novell driver
OdbcSelectCursorType	0 for DB2 and Oracle 1 for all other engines	Enables to specify the type of cursor.	Long integer: 0 FORWARD_ONLY 1 KEYSET_DRIVEN 2 KEYSET_DRIVEN 3 STATIC	Enter 0 to: DB2 NT
IsODBC3Compliant	1			Enter 1 to: Oracle native
OwnerDisplayname.owner	owner	Set the display name for the duplicate table when you set multiple owners. OwnerDisplayname is a key word and owner is the table owner that you entered when you configured the connector.	String	Enter OwnerDisplayName.orcl as the option name and cit as the option value.

Name	Default value	Description	Data type	Examples
QuoteChar ¹	empty	<p>Two conditions to use the option:</p> <ul style="list-style-type: none"> When the error ORA-01747 occurs in the mapping, it indicates that a field name in the Oracle database is identical to an Oracle reserved word. Table or field names in the Oracle database are in lower case. 	String	Enter <code>QuoteChar</code> as the option name and " (a straight quotation mark) as the option value .

1

The limitations of the QuoteChar option:

- Note that table and field names are case sensitive if the option is used.
- It is not supported if two field names in a table are the same regardless of case sensitivity, for example, `EmpName` and `EMPNAME`.
- It is not supported if a table name is identical to any Oracle reserved word.
- It is not supported if a field name is identical to one of these reserved words (in upper or lower case letters): `DISTINCT`, `FIRST_ROWS`, `FROM`, `HAVING`, `INTO`, `NOT`, `NULL`, `SELECT`, `VALUES`, and `WHERE`.

Defining Document Types


This screen enables you to define the extension file for the document types published by the connector. This file contains:

- The definition of the joins between the different tables of the database.
- Fields used as schedule pointers.

The original file must not be modified. For further information on editing an extension file, refer to the User Guide,

Caution: *Certain extension files cannot be modified using the configuration editor; In this case, you must use the extension file of the Database connector (`config\database\config\database.cfg` file located in the Connect-It installation folder.)*

Choosing a DTD or XSD

To create or process an XML file, the connector must use a DTD (Document Type Definition) or XSD (Extended Schema Definition). On the **Select a DTD/XSD** screen, indicate the DTD or XSD used in the **DTD/XSD** field. The document types published by the XML connector correspond to this DTD or XSD. A DTD is provided and populated with default values for this connector. By clicking , the **File location** screen is displayed. This screen enables you to enter the path of your DTD or XSD depending on your connection protocol.

You have the choice of three options:

- **Local/Network**

Enter the full path of your DTD or XSD file located on the computer or network on which Connect-It is installed.

- **FTP**

Enter the FTP parameters in the **Server**, **Login**, **Password**, and **Path** fields. The **Path** field enables you to specify the location of your DTD or XSD file on the server. For example, if your DTD is located in the `myfolder/dtd` folder on the `http://mycompany.com` server, you must populate the **Server** and **Path** fields with `http://mycompany.com` and `/myfolder/dtd/mydtd.dtd`.

Note: Do not include a slash mark (/) at the end of the server name; instead include it at the start of the path of the DTD or XSD file.

- **HTTP**

Enter the HTTP parameters in the **Server**, **Login**, **Password**, and **Path** fields. Note the items above regarding the **Path** field.

Determine the root elements

A DTD or XSD is composed of elements that can contain other elements. (For example: In a DTD intended for book publishing, **Section** elements are included in the **Chapter** elements, which are in turn included in the **Book** element.) The only element that cannot be included in another element is a root element. A DTD can contain any number of root elements. To determine the root elements, which correspond to as many document types as published by the XML connector, Connect-It gives you different options:

- Publish a document type for each root element selected by the user (values are separated by commas)
- Publish a document type for each root element found in the DTD/XSD (recommended)

Publish a document type for each root element found in the DTD/XSD (recommended)

If you choose this option, the XML connector searches all the root elements of the DTD or XSD and publishes a document type for each root element found. Example: In a DTD intended for book publishing, only one root element (**Book**) is found. In this case, the XML connector will only publish one document type.

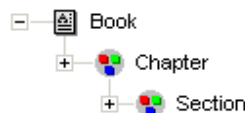


Figure 2-10: Example document types

Note: When no root elements can be found (all the elements that can include other elements), Connect-It will take the first element found in the DTD or XSD as its only root element.
 Example: A DTD corresponds to the organization of a relational database in which all the elements (each one representing a table) include one another: The Assets table is linked to the Users table, which is linked to the Assets table, etc.

Publish a document type for each root element selected by the user (values are separated by commas)

If you select this option, you can indicate the root elements of your choice, separating each one with a comma. Example: In a DTD intended for book publishing, the user decides to choose the elements **Book**, **Chapter**, and **Section**. In this case, the XML connector will publish one document type per root element chosen by the user.

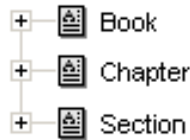


Figure 2-11: Example document types

For a DTD corresponding to the organization of a relational database in which all the elements (each representing a table) include one another, this option enables you to obtain a published document-type for each one of these tables.

Publish a document type for each first-level collection

By choosing this option, the XML connector publishes as many document types as there are first-level collections declared in the selected DTD or XSD. For example, a DTD, whose root element is **Company**, contains three first-level collections: **Employees**, **Suppliers**, and **Localizations**. If you do not select this option, the connector will publish only one document type having **Company** as its root element.

Note:

The first-level collections are those that appear directly under the root element of a document type published by a connector.



Figure 2-12: Published document types for first-level collections

For example, there will be one single "Company" XML document produced containing all employees, suppliers and locations in your database. Using the **Publish a document type for each first-level collection** option, the uCMDB (XML) connector publishes one document type per first-level collection. In this case, the root element of the DTD (the Company element) no longer appears in the tab of the document types published by the XML connector.

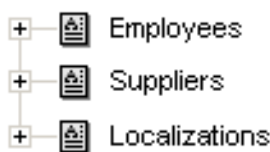


Figure 2-13: Published document types for first-level collections

For example, there will be as many XML documents produced as there are employees, suppliers and locations in your database.

Support for qualified names

This option is available when an XSD has been selected. This option is used to take into account qualified names in XML documents, as defined in the W3C standard. Support for this option is limited to XML schemas as in a document instance:

- All elements must be qualified whether they are declared globally or locally in the associated schema.
- Only attributes declared globally in the associated schema need to be qualified.

Learning mode (no documents produced)

The learning mode enables the XML connector to create a DTD according to the XML file or files specified on the **Select a file or folder** page.

Note: *The DTD obtained in learning mode must be saved in a local or network file. In this case, it is not possible to save it to an HTTP or FTP site.*

To start learning mode:


1. Configure the XML connector by selecting the **Learning mode (no documents produced)** option.
2. Select your XML connector in your scenario's diagram.
3. Do one of the following:
 - Select **Produce now** in the **Tools** menu.
 - Right-click and choose **Produce** from the menu that appears.
 - Press **F5**.

To disable learning mode, reconfigure the XML connector by clearing this option on the **Choose a DTD** page.

In case of conflict between XML and DTD/XSD documents

In principle, the XML elements of the files read by the uCMDB (XML) connector must match those defined in your DTD. Nevertheless, it is always possible that new elements appear. This type of conflict is frequently encountered when the DTD used is obtained in learning mode using a limited number of XML files containing a small number of elements.

To handle these conflicts, Connect-It provides two different options:

- **Reject file and continue**
In case of conflicts, the read XML files containing non-defined elements in the DTD or XSD are rejected. (Whether these documents are saved or not depends on the option selected on the **Action after processing** screen.) In order that these files be properly processed, you must restart the uCMDB (XML) connector in learning mode.
- **Issue warning and continue**
In the case of a conflict, the XML files are processed normally. A warning in the document log, recognizable by the  icon, appears in the tracking lines of the XML connector in read mode.

The types of conflicts handled by the application are:

- element names
- attribute names
- ? and * wildcard characters for occurrences

Ignore the XML files whose root element does not correspond to the current document type (do not apply post-processing actions)

This option enables you to process XML documents corresponding to different document types, and therefore linked to different DTDs. Selecting this option enables you to take current DTD into account alone, and as a consequence, avoid processing errors for XML documents linked to other document types. For example, if you have specified a folder in which there are several XML documents, select this option to only take into account the DTD defined in the **DTD/XSD** field.

Configuring the JVM

Enabling Parallel Consumption

Parallelization is applied to all connectors consuming a document type. It consists of duplicating a connector, making multiple processes, in order to process the consumption of documents in parallel.

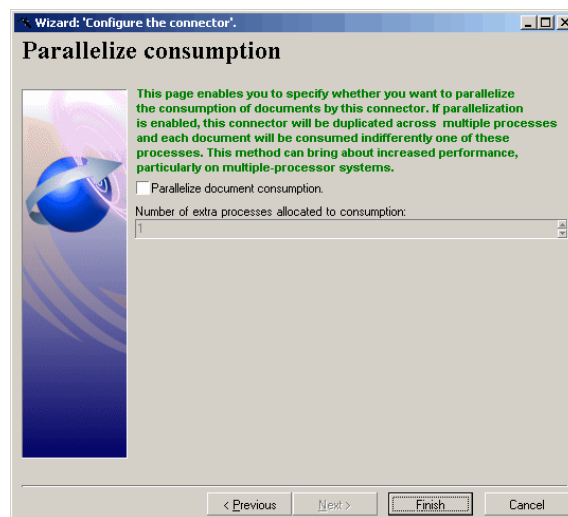


Figure 2-14: Parallelize consumption

- **Parallelize the consumption of the documents**
Select this option to enable the parallel processing of your consumed document types.
- **Number of additional processes allocated to consumption**
Populate this field with the number of processes running simultaneously and allocated to the consumption of documents.

Configuring the Proxy Server

This page lets you define the connection type that is used:

- **Direct connection**
- **HTTP**
Enter the name or IP address of the server, and the user and password if required.

- **SOCKS**

Enter the name or IP address of the server, and the user and password if required.

For HTTP and SOCKS type connections, you can specify conditions when the proxy server should be bypassed (**Do not use a proxy server for** field). You must enter an HTTP or SOCKS type address in this field depending on the connection that is used. Names must be separated by a space.

Configuring SSL

HP Connect-It supports Secure Hypertext Transfer Protocol (HTTPS), which encrypts and decrypts message requests and responses. The primary reason to enable Secure Sockets Layer (SSL) on Connect-It is to restrict access to the sever to clients that are known and identified by the server. You may select to enable SSL in the connector's configuration wizard, but the configurations may differ depending on the connector group.

Note: Only the configurations associated with SSL are listed below. For the complete configuration settings for each connector, see relevant information in this guide.

For Delimited-text and XML connectors

Provide values for the following fields on corresponding wizard pages.

- On the **Select a connection protocol** page:
 - Select **HTTP Web site** as the connection protocol.
- On the **Connect to the HTTP Web site** page:
 - **Use a secure connection (HTTPS protocol)**
 - Select the check box to indicate a secured connection is needed

- **Client certificate**

Select an HTTPS client certificate on your computer.

Note: This option is for two-way SSL. You can keep the default value as Nothing, if you do not need the server to require Connect-It to present a client certificate.

- **Do not check the server identity**

This is an optional field. Select the check box if you do not need the client to validate the server's certificate (the client to verify that the name in the certificate is the name of the server).

For E-Mail connectors

The SSL configuration for E-Mail connectors is on the wizard page of **Define the connection of parameters** and depends on the **Messaging type** you select. Connect-It supports SSL configuration for protocols of POP3, IMAP and SMTP.

E-Mail (fetching)

- **Messaging type**

Select **POP3** or **IMAP** as needed. Note: The number of fields in this wizard page varies according to the protocol selected in the Messaging type.
- **Server**

Enter the name of your server.

- **Connection security**
Select **SSL** or **STARTTLS** to provide encrypted email communication.
- **Connection port**
Indicate the connection port used by the server on your computer. By default, the value of this field is **110** when no security or when **STARTTLS** is selected, and **995** when **SSL** is selected.

E-Mail (sending)

- **Messaging type**
Select **Internet:SMTP** as the messaging protocol for SSL.
Note: SMTP is the only messaging protocol that E-Mail (sending) connector supports for SSL configuration.
- **Server**
Enter the name of your SMTP server.
- **Connection security**
Select **SSL** or **STARTTLS** to provide encrypted email communication.
- **Connection port**
Enter the connection port. The default port is **25**.

For Java connectors

In Connect-It, Java connectors that support SSL share the same wizard pages for SSL configuration (the HTTP Service connector is an exception to require additional configuration). See the [Configuring the JVM](#) for a list of Java connectors.

On the wizard page of **SSL configuration**, provide values for the following fields:

- **Use an SSL connection**
Select this option to use SSL. This will activate the following fields that are optional to fill.
- **CA certificates file (truststore)**
Select the Java certificate file for the secure connection. A CA certificate file is issued by a CA used for the client to verify the server. By default, Connect-It searches for information concerning the Java certificate file in the JRE, in the following files:
`/lib/security/jsse.cacerts` and `/lib/security/cacerts`.
- **Truststore password**
Enter the password to access the truststore.
- **Truststore type**
Select **JKS** or **PKCS12** as the truststore type.
- **Certificates and client key file (keystore)**
Select private key file to be used. This file is required if the private key is stored in a database.
- **Keystore password**
Enter the password to access the keystore.
- **Keystore type**
Select **JKS** or **PKCS12** as the keystore type.

- **Verify the server's identity**

This is an optional field. Select the option if you want Connect-It to verify the server's name.

For connectors using MySQL

In Connect-It, you can enable SSL for secure communication on all connectors that use MySQL as the connection type. On the wizard page of SSL configuration, provide values for the following fields:

- **Use an SSL connection**

Select this option to use SSL, and choose the following files:

- **Client key file**

- **Certificate file**

- **CA certificate file**

Note: The Client key file and Certificate file fields are for two-way SSL configuration. Leave the file fields empty if you are configuring one-way SSL.

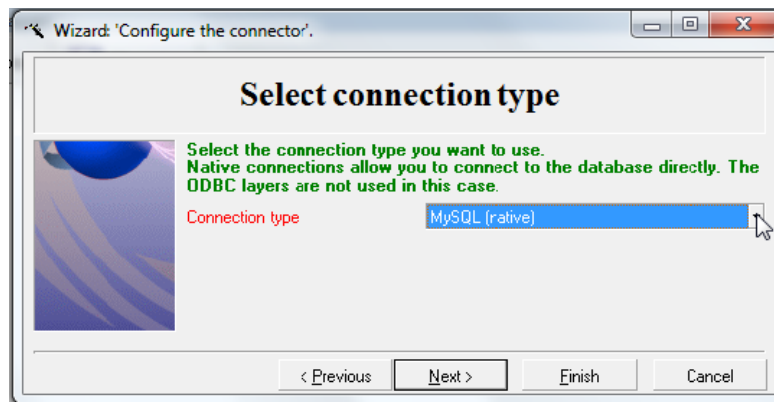


Figure 2-15: Selecting MySQL as connection type

For connectors supporting RESTful Exposure

RESTful Exposure exposes data as XML resources using a RESTful design, and this functionality is available for some connectors. When a connector is set to expose RESTful data, it can be regarded as a "listener" connector. See the topic of [RESTful Exposure](#) for the list of connectors supporting RESTful Exposure and more information about the feature.

The RESTful exposure requires the following SSL configuration:

- On the wizard page of **RESTful Webservices exposure**, provide information for fields below:
 - **RESTful exposure:** select the check box.
 - **Port:** port of the embedded HTTP server that will run.
 - **User/Password:** provide Basic Authentication support to restrict the usage to allowed clients.



Figure 2-16: SSL configuration for RESTful Exposure

- Add the following to the JVM options field on the Configure the JVM wizard screen:
-Dconfig.xml=<JETTY_XML_PATH>/<JETTY_CONFIG_XML_FILE>
For example: -Dconfig.xml=C:/SSL/jetty.xml
- Run the connector that use RESTful Web Services exposure in scheduler mode. Note that the default port used in the connector wizards is 8080. You can use another port number in the Jetty configuration file to override this setting (the example below uses port 8443).

The following is an example:

```
<Configure class="org.mortbay.jetty.Server">  
<Call name="addConnector">  
<Arg>  
<New class="org.mortbay.jetty.security.SslSocketConnector">  
<Set name="Port">8443</Set>  
<Set name="needClientAuth">>true</Set>  
<Set name="maxIdleTime">30000</Set>  
<Set name="keystore">file:///C:/SSL/server/keystore</Set>  
<Set name="keyPassword">password</Set>  
<Set name="truststore">file:///C:/SSL/server/truststore</Set>  
<Set name="trustPassword">password</Set>  
</New>  
</Arg>  
</Call>  
</Configure>
```

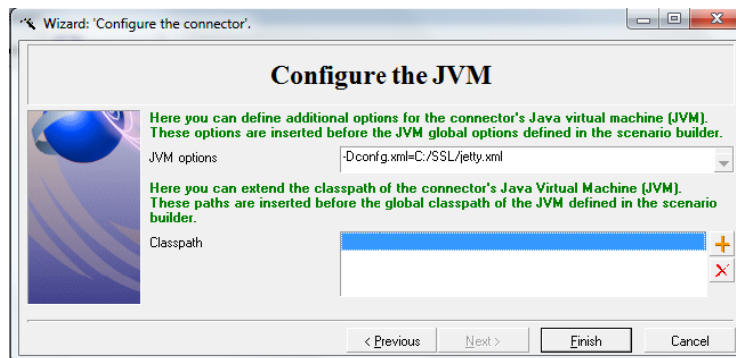


Figure 2-17: Adding JVM options

For HTTP Service connector

The SSL configuration for the HTTP Service connector is a combination of settings for Java connectors and connectors supporting RESTful exposure. For more information about how the HTTP Service connector works and how to configure the connector, see the topic [HTTP Service Connector](#).

- On the **SSL Configuration** page
See the configuration for Java connectors.
- On the **Configure the JVM** page
Provide JVM options. See the example in the "For connectors supporting RESTful Exposure" section.

Chapter 3

Connector Directives

Directives are special instructions for the connector that are followed when producing or consuming documents. You can create directives when you configure:

- Produced or consumed document types
- Mappings between document types

Directives vary depending on the connector. For example, for database type connectors, the production directives correspond to WHERE and ORDERBY clauses that filter the values of fields recovered in the source database. On the other hand, the consumption directives correspond to the reconciliation options that let you write data to a destination database.

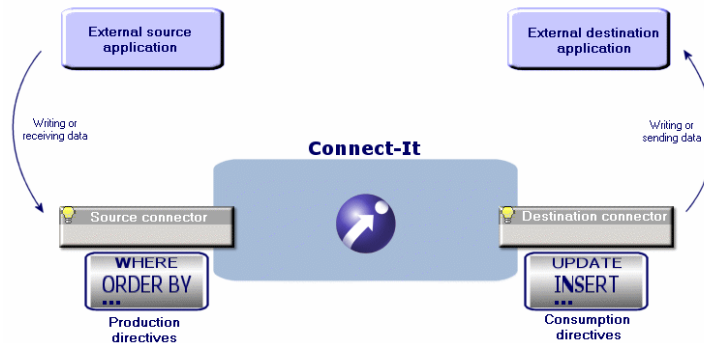



Figure 3-1: Connector directives

This chapter presents the production and consumption directives that are common to several connectors.

Defining Production Directives

The production directives concern how data that is read or received by a source connector is processed. Example: The directives for the Websphere MQ connector consist of a WHERE clause and some options concerning how to recover messages.

To define a connector's production directives:

1. Load or create a scenario that uses this connector.
2. Verify that the connector is correctly configured.
3. Open the connector (**Tools > Open** or F4 on the keyboard).
4. Select the **Document types** tab.
5. Click  or edit a document type produced in the **Produced document types** pane. A window appears in which you can create or edit a produced document type.
6. Enter the production directives in the window.

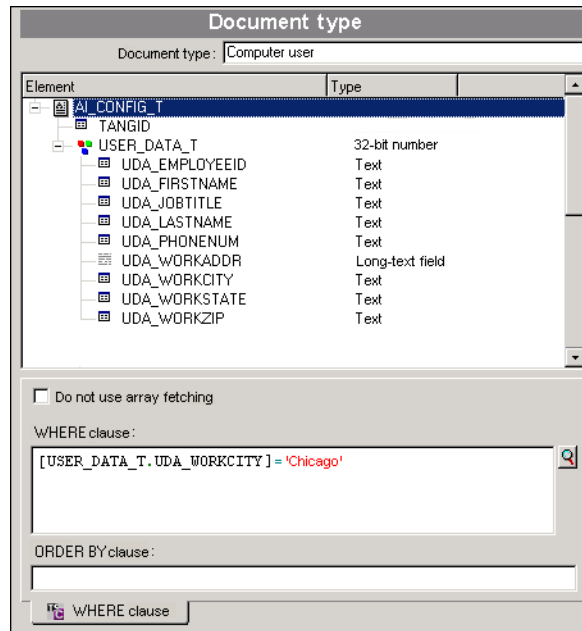


Figure 3-2: Document type window

WHERE and ORDERBY Clauses

Use the WHERE clause to filter the records in a source database. For example, you can filter records extracted from the database created on or after January 1, 2002. Simply use the following clause:

```
[field containing the creation date for a record] >= '1/01/2002'
```

For example, to filter a link, use the following syntax:

```
[OperatingSystem.OperatingSystem_ID] = 1000001107
```

Conventions used for the connectors supporting AQL

For the connectors using AQL, Connect-It imposes that you respect the following conventions:

- Character strings must be surrounded by single quotes. For example:
`'computer'`
- Numbers must be stated without quotes.
- Dates must be entered using the format defined in the regional settings of your computer and be stated in between single quotes.

Connectors supporting AQL

The following connectors support AQL:

- HP Asset Manager connector
- Database connector
- LANDesk for Inventory Connectors
- Tivoli Inventory connector (version 4.0)
- Microsoft System Center Configuration Manager (SCCM) connectors
- TS.Census 3 connector

- CA Unicenter AMO 3 connector
- Tivoli CM for Inventory 4.2 connector

The Sysdate field

In the WHERE clauses written in AQL, the use of the **sysdate** field must be replaced with the function **getdate()**.

For example, the SQL query

```
SELECT * from AmAsset where dinstall>sysdate
```

must be replaced with the AQL query

```
SELECT * from AmAsset where dinstall>getdate().
```

Do not use array fetching

This option must be selected, for example, when recovering Blob-type fields or **memo**-type fields when the table does not have a main field. If array fetching is used, Blob-type fields may not be processed correctly (they are clipped). This option may reduce performance in particular.

Using PifConstant in WHERE Clause

You can define constants for your mapping scripts. These constants are declared and saved in a file that can be accessed via the **Scenario > Script constants** menu. This file has the extension `.scp`. For more information about using constants, see the *HP Connect-It User Guide*.

PifConstant is the only function that can be used in WHERE Clause. This function enables you to identify a constant and then replace the input parameter's value with the constant. Note that PifConstant cannot be used as a nested function.

When you apply PifConstant in WHERE Clause, remember that the declaration of the constant depends on its type (integer, string, etc.).

For example:

- Use single quotes (") for a string or character (including a string or character that contains single quotes)
login = 'Maggie Smith'
NAME_CONST='Anne' 's'
- Declare a numeric constant as it is
idmax = 5
- Declare a date constant of international standard with single quotes
Instal.date = '2000/01/01'

The rules of using constants for the ServiceCenter and Service Manager connector is an exception. For more details, see [Configuring Advanced Mode Settings for the HP ServiceCenter and Service Manager Connector](#).


Defining Consumption Directives

The consumption directives concern how data that is written or sent by a connector to an external destination application is processed. For example, the directives for a database connector enable you to specify how the connector writes or updates records in the destination database.


You can define the consumption directives:

- In the Consumed document-types edit window
- In the Mappings edit window

To define a connector's consumption directives in the consumed document types edit window:

1. Open or create a scenario that uses this connector.
2. Verify that the connector is correctly configured.
3. Open the connector (**Tools > Open** or **F4** on the keyboard).
4. Select the **Document types** tab.
5. Click  or edit a consumed document type in the **Consumed document type** pane. You can create or edit a consumed document type in the Document type edit window that appears.
6. Enter the consumption directives in the window.

To define a connector's consumption directives in the Mappings edit window:

1. Open or create a scenario that uses this connector. In the scenario, the mappings need to have already been created between this connector and another connector.
2. Open your scenario's connectors.
3. Select the **Mappings** tab.
4. Click  or edit an existing mapping The Mapping edit window appears.
5. Enter the production directives in the Mappings window.

Reconciliation

For database type connectors, you specify consumption directives by indicating reconciliation parameters. These parameters indicate:

- Which fields are used to reconcile data consumed by a connector and the records in the destination database.
- Which operation the connector performs on the records in the destination database: insert, update or delete.

Consumption directives must be entered in the **Reconciliation** and **Advanced reconciliation** tabs.

Configuring Basic Reconciliation Settings

On the **Reconciliation** tab, indicate a reconciliation type and select the reconciliation keys.

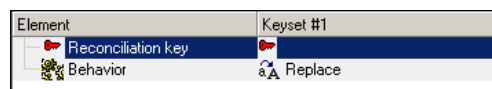


Figure 3-3: Reconciliation key

Selecting a reconciliation type

The reconciliation types enable you to select the action to be performed:

- Update or insert
- Update
- Insert only
- Delete

To indicate a reconciliation type:

1. Select a root node, a structure, or a collection in the Mappings window.
2. Select the option of your choice in the **Reconciliation type** drop-down list.

Selecting reconciliation keys

After having selected a reconciliation type for a complex element, you must select the fields enabling the reconciliation of source and destination data. These fields are identified by a reconciliation key. The fields selected as reconciliation keys must respond to the following criteria:

- **Restriction of doubles**
For example, an **ID** field in a database table does not allow two records to share the same ID. So, the **First name** field in the Employees table is a bad choice for a reconciliation key since several employees can share the same first name.
- **Restriction of null values**
For example, a **serial number** field in the Computers table of a computer hardware supplier's database. Under no circumstances can a computer not have a serial number.
- **Indexing**
The reconciliation process goes faster if an indexed field is used as the reconciliation key.

To select a reconciliation key:

1. Select this element in the work area.
2. Then, select the **Reconciliation key** option or click directly on the transparent key in the pane showing the consumed-document type.

For example, in the **amAsset** document type of the `pdi\pdi8ac44\pdiaac.scn` scenario, the **AssetTag** field and the **Brand** and **Model** fields of the Product structure are chosen as reconciliation keys.

Using a Structure as a Reconciliation Key

The first level of a document type published by the Asset Manager connector represents a table in Asset Manager. The tables linked to this table by a 1-1 or 1-N type link are represented by structures. You can identify a record with the fields of a structure by choosing them as reconciliation keys. In order for these fields to be taken into account when transferring data, you must also select the structure containing these fields.

In fact, each structure contains an **Identifier** field (not shown on-screen) that allows it to be linked to another table. Placing a reconciliation key on the structure comes down to indicating that the **Identifier** field is also used to identify the record.

In the following example, the document type representing an asset creates a record using the:

- **Name** field of the **Assets** root node.
- **Identifier** field of the **Product** structure.

- **Model** field of the **Product** structure.

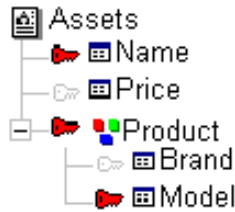


Figure 3-4: Structure as a reconciliation key

Using a Parent ID as the Reconciliation Key

1. Select a collection.
2. Select the **Use the parent Id as the reconciliation key** option.

For example, in the database of an Asset Manager application, the child tables are linked to the parent tables by a foreign key, which is the ID of the parent. In Asset Manager, the Portfolio items table (**amPortfolio**) is linked to the Sub-assets table (the table of the items of the portfolio) by a **lparentId** field.

When you select the **Use the parent Id as the reconciliation key** option, any creation of a record linked to a parent record is rejected if the relationship is different from that in the database. This is only valid if the link between two tables is a 1-1 or 1-N type link: A record in the child table can only be linked to a single record in the parent table.

There is a parent-child relationship between a computer A and a printer X. A document consumed by the Asset Manager connector has a new parent-child relationship that links a computer B to the same printer X. The names and asset tags of the asset and sub-asset are chosen as the reconciliation keys.

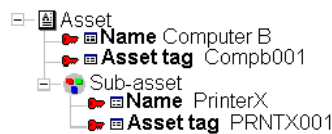


Figure 3-5: Parent ID as reconciliation key

To use the parent ID as the reconciliation key

The following example illustrates how Connect-It behaves when the **Use the parent Id as the reconciliation key** option is selected and then how it behaves when this option is not selected.

- **Using the parent Id as the reconciliation key option selected**
The parent-child relationship is not updated because the **Use the parent Id as the reconciliation key** check box indicates that the Printer X sub-asset is already linked to an asset in the Assets table.

Parent Id as the reconciliation key option selected

Assets table		Sub-assets table				
Name	Asset tag	Identifier	Parent Id	Name	Asset tag	
Computer A	COMPA001	CPAAA1	-∅-	CPAAA1	Printer X	PRNTX001
Computer B	COMPB001	CPBBB1				

- **Using the parent Id as the reconciliation key** option cleared
The parent-child relationship is updated. The **Parent Id** field is updated and links the sub-asset Printer X to the asset Computer B.

Parent Id as the reconciliation key option not selected

Assets table		Sub-assets table				
Name	Asset tag	Identifier	Parent Id	Name	Asset tag	
Computer A	COMPB001	CPBBB1	-∅-	∅ CPBBB1	Printer X	PRNTX001
Computer B	COMPA001	CPAAA1	-			

To enable a parent change

When the **Use the parent ID as the reconciliation key** option is used, the existence of the parent is checked before a record is inserted or updated. If the parent element has changed, the query to this record fails and an error is saved in the Connect-It log.

In order to update records following changes between a parent record and its dependencies (such as a monitor linked to a computer) the **Enable parent change** option is used.

This option sends a query to the Asset Manager database and uses reconciliation keys defined by the user instead of the parent ID. If a record is found, the reconciliation engine adds the parent ID to the child record in order to create a dependency.

To couple the reject of a parent node to its child node

In certain cases, you can indicate that the rejection of a parent node automatically triggers the rejection of its parent node.

1. Select a document-type element in the workspace.
2. Select the **Reject the parent node if rejection** option.

For example, in an Asset Manager application, you do not want a record to be created in the Portfolio items table (**amPortfolio**) if the record that is linked to it in the Employees table (**amEmpIDept**) - the user of this asset - is rejected. This avoids having assets without users in the Asset Manager database.

Using Alternate Reconciliation Key Sets

Connect-It enables you to use several reconciliation key sets. Each key set is assigned a given weight. This is useful when the reconciliation keys do not let you reconcile your records.

For example, in the source document, a field selected as the reconciliation key doesn't have a value even though the destination application requires that it be populated. By indicating that a

reconciliation key belongs to a second set Connect-It uses this key if the reconciliation performed using the first key set fails.

Case-Sensitive Reconciliation

For this option, the behavior of Connect-It is as follows:

- **Case-sensitive reconciliation** option selected
The reconciliation key values are case-sensitive. Example: If a field containing the e-mail address of your employees is used as a reconciliation key, the values 'jmartin@company.com', 'JMARTIN@company.com' and 'jMartin@company.com' will correspond to three different records in the destination application.
- **Case-sensitive reconciliation** option cleared
The reconciliation key values are no longer case-sensitive. Example: If a field containing the e-mail address of your employees is used as a reconciliation key, the values 'jmartin@company.com', 'JMARTIN@company.com' and 'jMartin@company.com' will correspond to the same record in the destination application.

Note: *This option only appears if the database engine is case sensitive. For connectors using an ODBC connection, this option is selected by default.*

Reconciliation on Fixed Length Fields

Enable this option to define a fixed length for a field (for example, for the field that has the reconciliation key). If this option:

- Is not enabled, and the value for the destination field does not equal the value returned by the source connector, then a new record is created.
- Is enabled, and the values for the source data and the data to be inserted are not the same, then the value of the field is normalized depending on the value present in the destination database, and the related data is updated.

For example, for a field whose length is set to 10, the string that is inserted in the database will be truncated to 10 characters:

- The string "test" (string with 4 characters) is inserted, and "test_____" (string with 10 characters) is saved in the target database.
Because of the difference in string lengths, reconciliation cannot be performed because "test" is compared with "test_____". Each time the scenario executes, a new record is created.
- Enable this option to normalize the field and not insert a new record. The "test" string takes the value of the destination field and becomes "test_____" and is compared with "test_____".
Since the values are the same, the reconciliation can be performed.

Selecting Reconciliation Behavior

For each destination field, you can select one of the following options:

- **Replace**
- **Add**

By default, the **Replace** option is selected. The **Append** option is useful for **memo**-type fields. For example, you can append comments to comments already in a database table.

Managing Replication Conflicts

If a replication conflict is encountered, you have the choice of three options:

- Generate a reject.
- Log warning if new entry or element type modified
- Overwrite

Configuring Advanced Reconciliation Settings

The advanced reconciliation tab enables you to:

- Set the reconciliation parameters relating to the collections of a consumed document.
- Minimize the number of requests when reconciling collections
- Indicate the action performed by the connector in case of a replication conflict.

Activating the Reconciliation of Collections

The **Activate collection reconciliation** option enables you to indicate certain parameters. These parameters let you control updating or inserting destination collection members according to the source collection members.

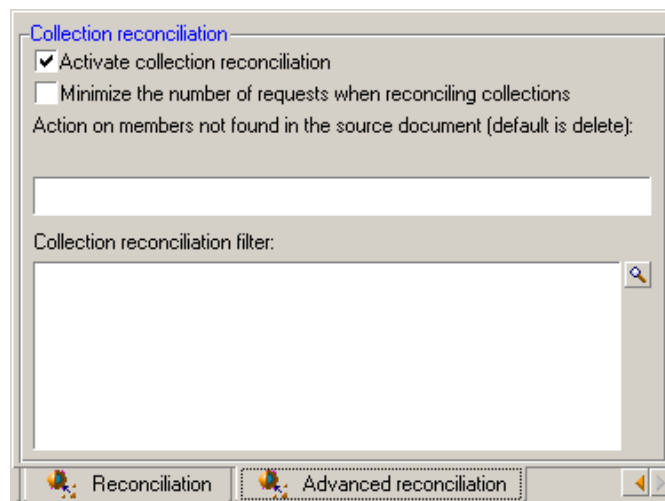


Figure 3-6: Collection reconciliation

Managing the absent members of the source collection

The **Action on members not found in the source document** field enables you to indicate that:

- The members absent from the source collection are not deleted from the destination application.
- A field of records corresponding to these absent members have a significant value. For example, if you assign to the collection's Status field the value **To delete**. You now need to find all the records that have **To delete** as the value of their Status field in the destination-application's administration. Use the following syntax to indicate this field:

```
[field name] = "[value of your choice]"
```

For example:

```
Status = "To be deleted"
```

Note: If collection reconciliations have empty data for the source database, records in the target database are not erased.

Filtering certain collection members

When updating a collection, the Reconciliation filter field for the collection allows you to:

- Delete members of the collection that are not found in the source document from the destination application.
- Limit the scope of this deletion by filtering certain collection members in the database of the destination application.

For example, in an **Asset information** mapping of a scenario between the Desktop Discovery connector to Asset Manager connector, a **Software** source collection is mapped to the **SoftInstall** destination collection. These two collections represent the software installed on an asset (a computer in this case).

Sometimes certain software applications might be uninstalled. By selecting the option **Action on members not found in the source document**, all the software that is not in the **SoftInstall** collection is deleted from the database of the Asset Manager application.

However, to avoid deleting certain software (not identified by a Desktop Discovery scan, for example), you can write a clause in the **Collection reconciliation filter** field.

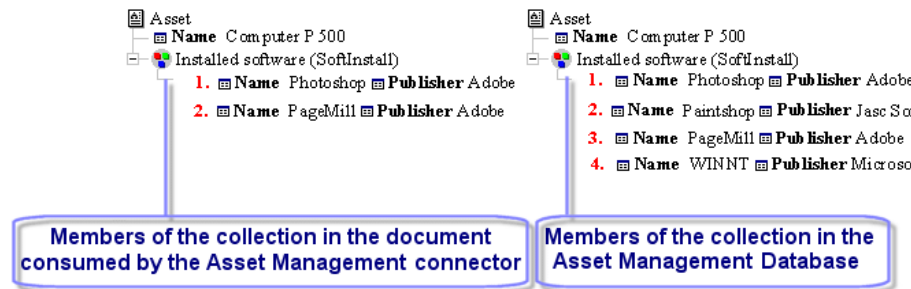


Figure 3-7: Filtering Collections

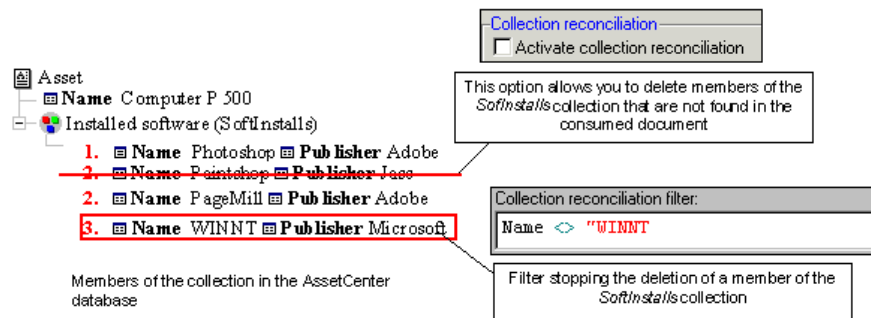


Figure 3-8: Deleting members of the destination collections

Using Reconciliation Scripts (for the Asset Manager connector only)

The **Reconciliation scripts** tab is available when the Asset Manager connector is in consumption mode.

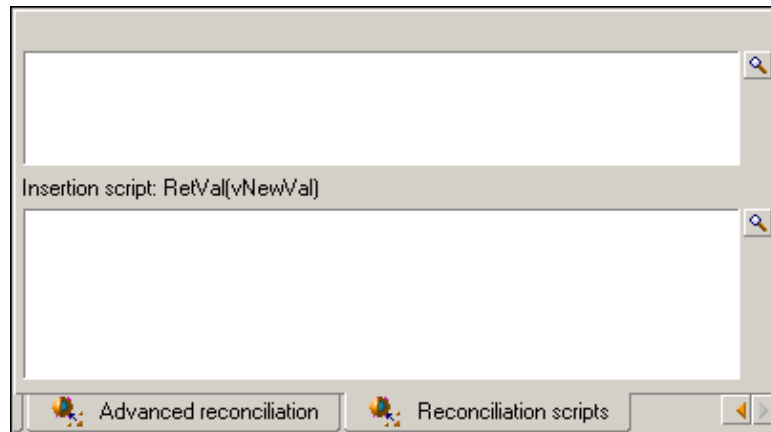


Figure 3-9: Reconciliation scripts tab

This tab enables you to enter the scripts required to reconcile two document types, a produced one and a consumed one.

These scripts enable you to perform the following actions on the target connector's document type:

- Update if the value exists in the target database
- Insert if the value is missing from the target database

Syntax

The syntax for the update and insert scripts is the following:

- vNewVal: Value of the source document
- vOldVal: Value contained in the target database
- vOldId: Current value of the ID of the record that will be modified

You can use square brackets [] and Basic functions in your scripts.

For example:

```
// Keeps the old value
RetVal = vOldVal
If vNewVal >= vOldVal and [dtLastModif] > '1980-11-12 10:11:12' Then
  pifSetPendingDocument('Document not finished')
  DbExecAql('INSERT INTO .....')
// Gets the new value
RetVal = vNewVal
End If
```

How reconciliation scripts work

The reconciliation scripts are an extra step in refining how your mappings are managed. These scripts concern data contained in the document types produced by a connector and consumed by another. Depending on whether the fields in the target database contain data or not, scripts enable you to modify the value of a mapped field (update) or create it (insert).

For example, the source connector produces a document type A containing the following fields:

- Name
- Id
- Comment

The ID field is the reconciliation key.

The Asset Manager destination connector consumes the document type A, and is connected to a destination database containing the same fields. The values of the document are as follows:

- Name = Test
- Id = 1
- Comment = Transfer

The values of the destination database are as follows:

- Name = Currency
- Id = 1
- Comment = (empty)

The update script for the Name field is the following

```
RetVal=vOldVal
```

The insert script for the Comment field is the following

```
RetVal=vNewVal
```

After executing the reconciliation scripts, the document consumed by the Asset Manager connector will contain the following information:

- Name = Currency
- Id= 1
- Comment= Transfer

Two-pass Reconciliation

This option is used to reconcile a document type in two passes to prevent the Asset Manager connector from receiving errors due to deadlock situations which may occur when a table is accessed multiple times in a mapping. Deadlock situations often occur when parallel processing mode is used (option available in the connector's configuration wizard).

The **Two-pass reconciliation** option is available when the document root is selected, and is visible on the **Reconciliation** tab. Its operation is linked to the **Execute during the first pass** option, available for complex elements whose type is structure or collection.

Caution: When the **Execute during the first pass** option is set for an element, none of its sub-elements can have this option.

In order to optimize processing, we recommend using this option with the cache enabled (option available in the connector's configuration wizard). When a table is used frequently (for example, the models table) using the cache and two-pass processing reduces the number of queries sent by Connect-It.

Processing order when using two-pass reconciliation

When the **Two-pass reconciliation** option is enabled and deadlocks occur, nodes with the **Execute during the first pass** option are processed first and the document is consumed by the connector.

Note: *Enabling this option disables parallel processing when deadlocks occur.*

The document is processed a second time and parallel mode is reactivated, if this option was enabled.

Chapter 4

RESTful Exposure

RESTful Exposure exposes data managed by a connector directly as XML resources through HTTP protocol using a RESTful design.

The aim of this feature is to address the following issues that might occur with the XML Listener connector:

- Inability to perform synchronous request/reply
- Depends on an external web component to be installed
- Rely on a proprietary protocol

RESTful Exposure is available for connectors:

- HP Asset Manager
- HP Discovery and Dependency Mapping Inventory (DDMI)
- HP Configuration Management: Inventory Manager, Usage Manager and Service Events
- HP Service Desk (Outbound)
- Database
- Altiris
- LanDesk
- SCCM
- Tivoli CM for Software Distribution
- Tivoli Inventory

You can configure JVM options for a connector if you select the **Restful expose** check box in the configuration wizard. See [Configuring the JVM](#) for more details.

Resources are published as a web feed described by the Atom Syndication Format (see RFC 4287 <http://tools.ietf.org/html/rfc4287>) using the Atom Publishing Protocol (see RFC 5023 <http://bitworking.org/projects/atom/rfc5023.html>). The Atom Publishing Protocol is an HTTP-based approach for creating and editing Web resources. It is designed on the idea of using the basic operations provided by the HTTP protocol (such as GET, PUT, and DELETE) to pass around instances of Atom 1.0 Feed and Entry documents that represent resources.

Thus all operations are performed using simple HTTP requests following a RESTful style:

- Issuing an HTTP GET request to an URI returns an Atom Feed Document which is a collection of entries.
- To create new entries in that feed, clients send HTTP POST requests to the collection's URI.
- To modify an entry, the client send HTTP PUT request to the entry's URI with its modifications.

- To remove an entry from the feed, the client send an HTTP DELETE request to the appropriate entry URI.

What Connect-It does is simply to expose the document types you've created: produced document type based on a resource will correspond to HTTP GET operations while consumed document type for same resource will correspond to either POST, PUT or DELETE. The resource (data) corresponds to the Atom entry's content, that is XML-based content.

Using a browser

The RESTful exposure comes with a default HTML representation to allow an easy access to the resources from an HTTP browser. Thus any read operation (HTTP GET) can be achieved within the browser.

Configuring Restful Exposure

In order to respond to HTTP operations while configured to expose its data, a connector will start an embedded web server. To configure your connector in this mode, just go to the wizard's page to enable this behavior, and provide the following information:

- **Port:** port of the embedded HTTP server that will run
- **User/Password:** provide Basic Authentication support to restrict the usage to allowed clients.



Figure 4-1: Configuring RESTful WebServices exposure

Produced document types design

A produced document type is for retrieving resources but not modifying them. Thus it will interact with HTTP GET requests. Because each of the resources is represented as an atom entry of an **Atom** feed, a mapping will be done to specify the fields that are part of the atom entry attributes using the **RESTful exposure** tab.

Mapping Document type attributes to Atom entry attributes

Atom attribute	Description
atom:entry	Document type attribute
atom:id	Mandatory. A field to represent a unique ID to identify the given resource.
atom:title	Optional. A field that is to describe what is the resource.
atom:published	Optional. A date field to represents when that resource has been created.
atom:author	Optional. A string field to represent the owner of this resource.
atom:summary	Optional. A description field which provide detailed info about that resource.
atom:updated	Optional. A date field to represent when the resource has been modified. When document type is using a field as the pointer for scheduling, then it must be that field.

Only one produced document type per resource can exist. If you don't want those resources to be accessed in read mode, then you can disable the created document type using the corresponding check-box. Prior to defining any consumed type document type you must specify how the resource will be exposed in read-mode using a produced document type, just because it has to reply to a write operation.

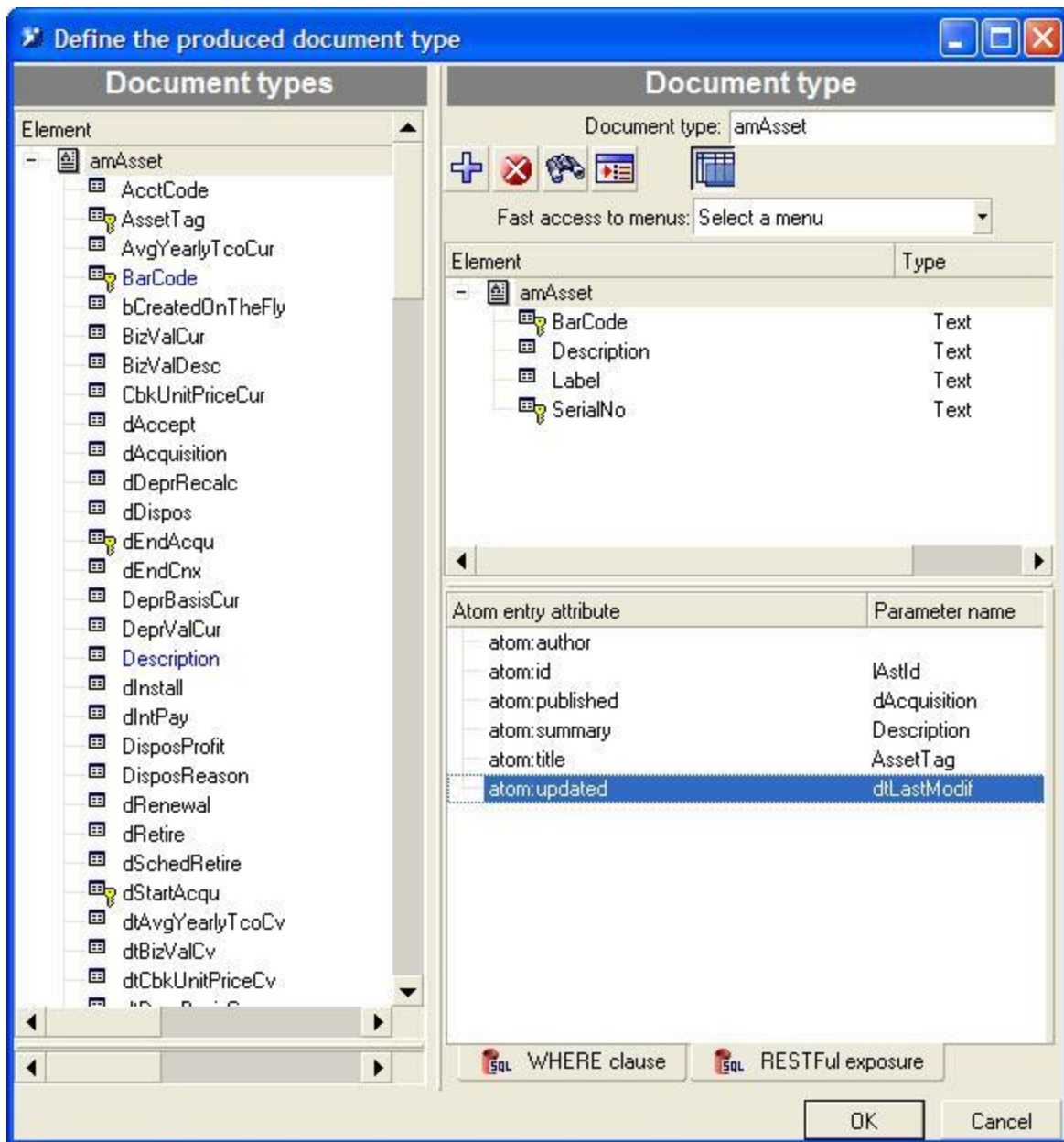


Figure 4-2: Produced document types design

Configuring Two-Way SSL

1. For connectors that use RESTful Web Services exposure, add the following to the **JVM options** field on the **Configure the JVM** wizard screen:
-Dconfig.xml=<JETTY_XML_PATH>/<JETTY_CONFIG_XML_FILE>
For example: -Dconfig.xml=C:/SSL/jetty.xml
2. Run the connector that use RESTful Web Services exposure in scheduler mode.
Note that the default port used in the connector wizards is **8080**. You can use another port number in the Jetty configuration file to override this setting (the example below uses port 8443).

The following is an example:

```
<Configure class="org.mortbay.jetty.Server">
  <Call name="addConnector">
    <Arg>
      <New class="org.mortbay.jetty.security.SslSocketConnector">
        <Set name="Port">8443</Set>
        <Set name="needClientAuth">>true</Set>
        <Set name="maxIdleTime">30000</Set>
        <Set name="keystore">file:///C:/SSL/server/keystore</Set>
        <Set name="password">password</Set>
        <Set name="keyPassword">password</Set>
        <Set name="truststore">file:///C:/SSL/server/truststore</Set>
        <Set name="trustPassword">password</Set>
      </New>
    </Arg>
  </Call>
</Configure>
```

Consumed Document Types Design

Depending whether the given connector supports the reconciliation mechanism or not, two cases are to be considered:

- **No reconciliation**

If reconciliation is not supported, then the only HTTP operation that is available is a POST, allowing to 'create' resources. Only defining the document type is required. In that case there is no way to deactivate the HTTP operation from the web other than deleting the document type.

- **Reconciliation supported**

If reconciliation is supported then you may allow to create, update and delete resources. When designing the document type, on the 'reconciliation' tab you must in that case choose the 'User' reconciliation type. On the document type, the reconciliation keys are used to respond to an HTTP POST request to create (or update) a resource. PUT and DELETE operations will not use the defined keys but instead the 'atom:id' defined in the corresponding produced document type to uniquely identify the resource which the operation should apply to.

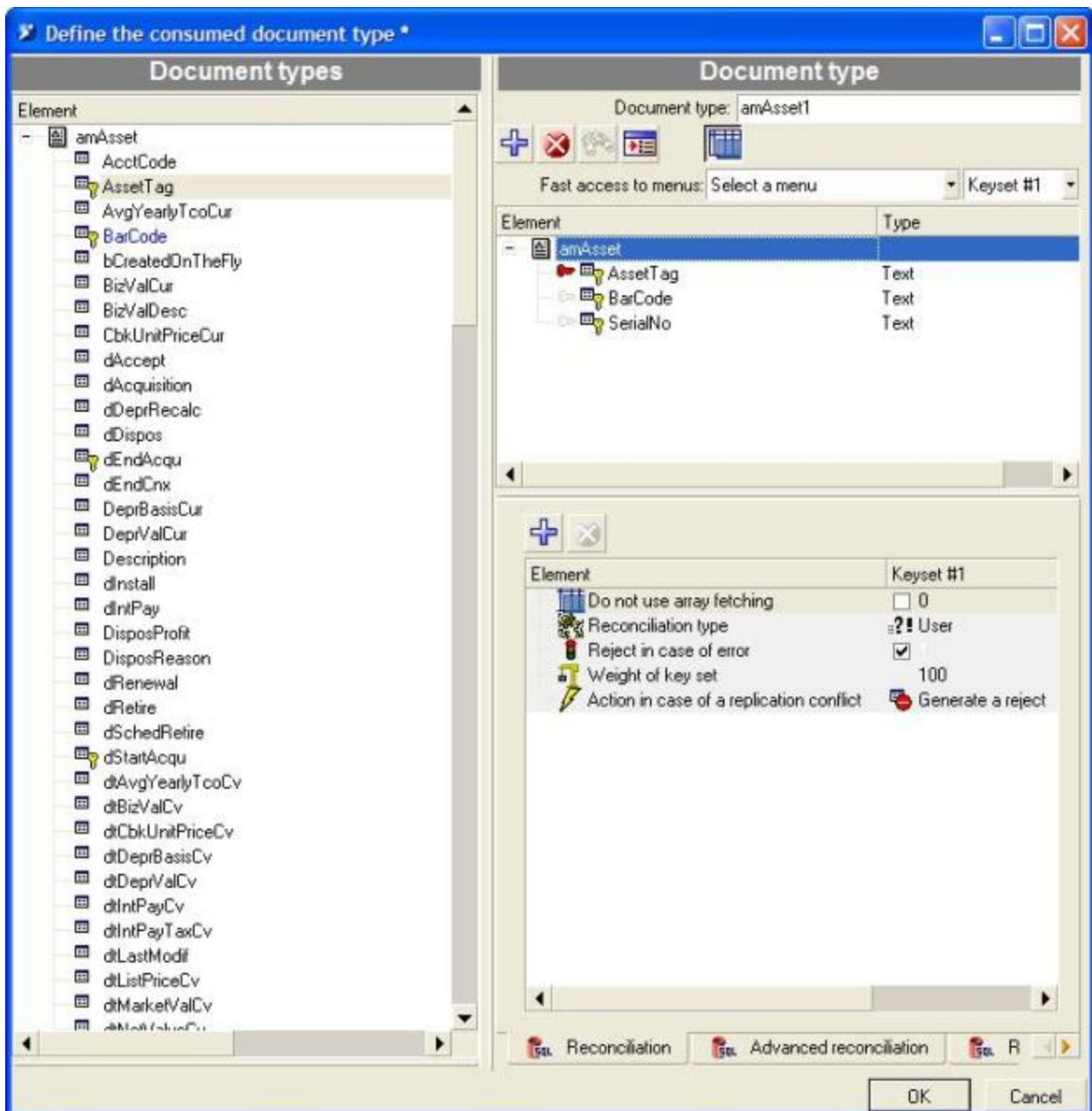


Figure 4-3: Consumed document types design: Reconciliation

On the 'RESTful exposure' tab, you have to define what are the operations a user can perform on a resource through HTTP requests by enabling the available reconciliation types such as:

- **Delete:**
A user can perform DELETE operation with XML content. It will then use the resource's id as the reconciliation criteria to delete the corresponding resource.
- **Insert only:**
A user can perform POST operation with XML content. It will then use the defined reconciliation to determine if a resource has to be created.

- **Update only:**

A user can perform PUT operation with XML content. It will then use the resource's id as the reconciliation criteria to update the corresponding resource.

- **Update or insert:**

A user can perform POST operation with XML content. It will then use the defined reconciliation to determine if a resource has to be either created or updated.

You can define multiple consumed document types for the same resource as soon as their HTTP available operations are not in conflict.

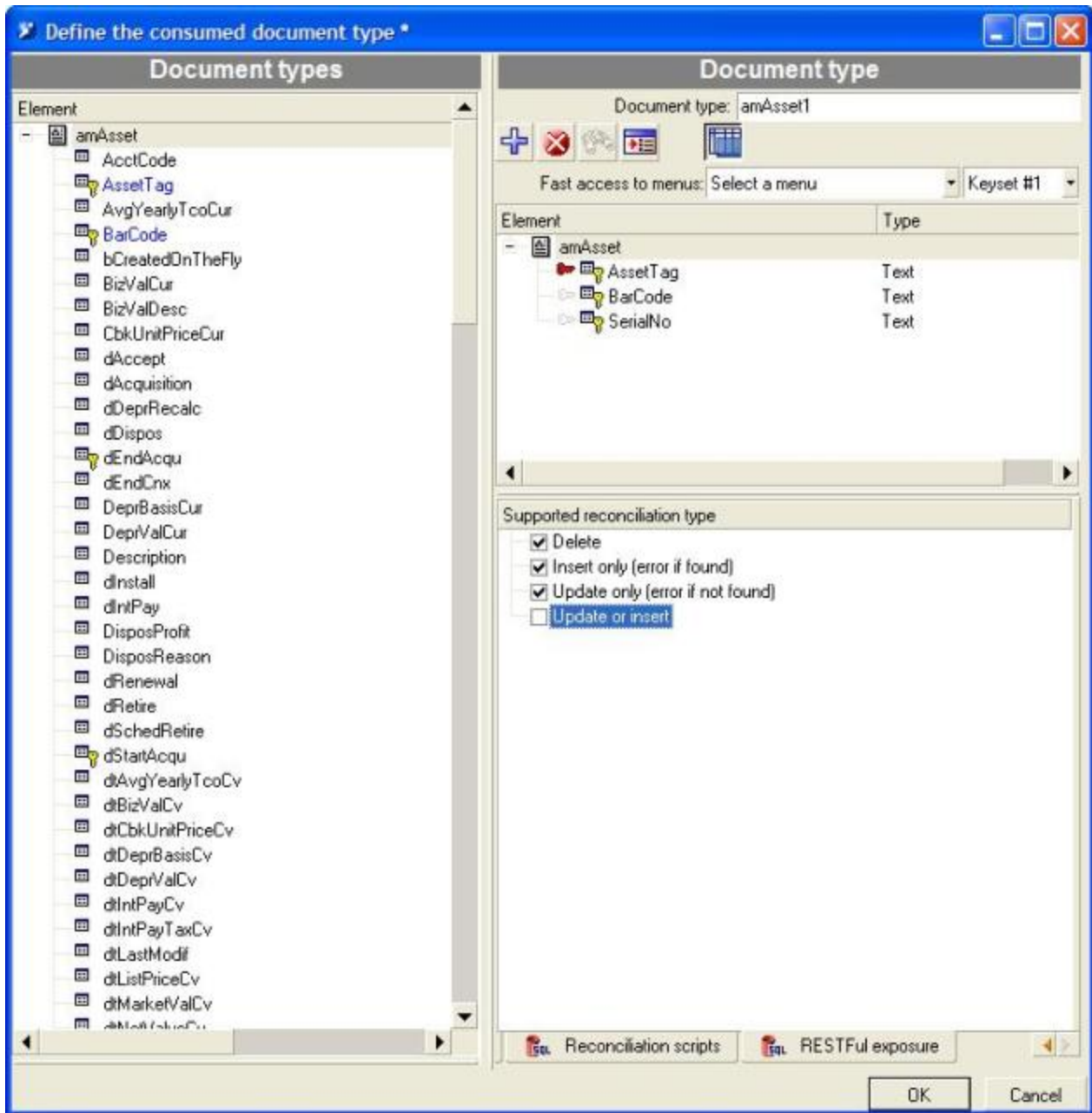


Figure 4-4: Consumed document types design: RESTful exposure

Runtime

Once you're done with designing produced and consumed document types, there is no other step to perform. In particular, when configured to expose RESTful resources, a connector's document types are not to be attached with a given scheduler.

A scenario involving such connector is started just like a regular scenario. Doing so it will start an embedded HTTP server, automatically deploy the REST web service to access data and wait for external client to connect, listening HTTP requests.

Discovering What Collections are Available

The first step to use the deployed service is to determine what collections are available and what types of resources those collections can contain. The Atom protocol specification defines an XML format known as a service document that a client can use to introspect an endpoint. To retrieve the service document, send a HTTP GET request to the service document URI of the form:

```
http://{host}:{port}/cit-rest/service
```

Acceptable response MIME types are:

- application/atomsvc+xml
- text/html
- Eg: Retrieving a service document from a server

```
GET /cit-rest/service HTTP/1.1  
Host: www.hp.com  
Accept: application/atomsvc+xml
```

The server should respond with a service document that enumerates the collections (feeds that correspond to the document types) available to the client as illustrated below:

```
| HTTP/1.1 200 OK  
| Content-Type: application/xml; charset=utf-8  
| Content-Length: nnn  
| <service xml:base="http://www.hp.com/cit-rest/service/"  
|   xmlns:atom="http://www.w3.org/2005/Atom"  
|   xmlns="http://www.w3.org/2007/app">  
|   <workspace>  
|     <atom:title>CIT Workspace</atom:title>  
|     <collection href="incidents">  
|       <atom:title>incidents</atom:title>  
|       <accept/>  
|     </collection>  
|     <collection href="assets">  
|       <atom:title>asset</atom:title>  
|       <accept/>  
|     </collection>  
|   </workspace>  
| </service>
```

Each collection of the service can be further accessed using its URI (relative or absolute) given by the 'href' attribute:

```
<collection href="assets">
```

Retrieving the Entries in a Collection

Clients can retrieve all the entries of a collection (feed) by issuing a GET request to the collection's URI of the form:

```
http://{host}:{port}/cit-rest/service/{feed}
```

Acceptable response MIME types are:

- application/atom+xml
- application/json
- application/xml
- text/xml
- text/html

For example, listing the all the assets from their corresponding feed:

```
GET /cit-rest/service/assets HTTP/1.1
```

```
Host: www.hp.com
```

```
Accept: application/atom+xml
```

The server should reply as illustrated below:

```
| HTTP/1.1 200 OK
| Content-Type: application/xml; charset=utf-8 | Content-Length: nnn
|
| <feed xml:base="http://www.hp.com/cit-rest/service/assets"
|   xmlns="http://www.w3.org/2005/Atom">
|   <id>assets</id>
|   <updated>2009-02-24T15:42:34.518+01:00</updated>
|   <title type="text" xml:lang="en">assets</title>
|   <link href="assets?start-index=0" rel="first"/>
|   <link href="assets?start-index=3" rel="next"/>
|   <link href="assets" rel="self"/>
|   <author>
|     <name>HP Connect-It</name>
|   </author>
|   <entry>
|     <id>66863</id>
|     <updated>2008-10-19T15:30:00.000+02:00</updated>
|     <title type="text" xml:lang="en">DSK001256</title>
|     <summary type="text" xml:lang="en"> 128 MB
DIMM, SDRAM (DSK001256) </summary>
|     <published>2008-10-19T00:00:00.000+02:00</published>
|     <link href="assets/66863" rel="self"/>
|     <author>
|       <name>HP Connect-It</name>
|     </author>
|     <content type="application/xml">
|       <assets>
|         <BarCode/>
```

```
|           <Label>DSK001256</Label>
|           <SerialNo/>
|           <Description> 128 MB DIMM,
SDRAM(DSK001256)</Description>
|         </assets>
|       </content>
|     </entry>
|   <entry>
|     <id>66858</id>
|     <updated>2008-10-19T15:29:57.000+02:00</updated>
|     <title type="text" xml:lang="en">DSK001255</title>
|     <summary type="text" xml:lang="en"> 32768 MB FLASH(DSK001255)
|     </summary>
|     <published>2008-10-19T00:00:00.000+02:00</published>
|     <link href="assets/66858" rel="self"/>
|     <author>
|       <name>HP Connect-It</name>
|     </author>
|     <content type="application/xml">
|       <assets>
|         <BarCode/>
|         <Label>DSK001255</Label>
|         <SerialNo/>
|         <Description> 32768 MB FLASH(DSK001255)</Description>
|       </assets>
|     </content>
|   </entry>
| </feed>
```

Each entry of the feed can be further accessed using its URI (relative or absolute) given by the link identified by the type of relation 'self' as follow:

```
<link href="assets/66863" rel="self"/>
```

So that to retrieve the entry using the above link, clients would have to issue the following request:
GET /cit-rest/service/assets/68863" HTTP/1.1

Because the response can contain a huge amount of entries, the result will be using paging mechanism if necessary. Thus response might contain additional links of relation type, either 'first', 'next' or 'previous' to navigate within the feed:

```
<link href="assets?start-index=0" rel="first"/>
<link href="assets?start-index=3" rel="next"/>
```

So that to retrieve the next entries from the above feed, clients would have to issue the following request:

```
GET /cit-rest/service/assets?start-index=3" HTTP/1.1
Host: www.hp.com
Accept: application/atom+xml
```

To reduce or increase the payload of the response, the default page size can be overridden using the 'page-size' query parameter on the request's URI such as:

```
GET /cit-rest/service/assets?page-size=10 HTTP/1.1
Host: www.hp.com
Accept: application/atom+xml
```

Managing Entries

Retrieving an entry

To retrieve an entry, clients need to issue a GET request to its given URI, for example:

```
http://{host}:{port}/cit-rest/service/{feed}/{entry}
```

• Acceptable response MIME types are:

- application/atom+xml
- application/json
- application/xml
- text/xml
- text/html

For example, retrieving the ATOM representation of an asset from its corresponding URI:

```
GET /cit-rest/service/assets/66863 HTTP/1.1
```

```
Host: www.hp.com
```

```
Accept: application/atom+xml
```

The server response will be similar to the following:

```
| HTTP/1.1 200 OK
| Content-Type: application/xml; charset=utf-8 | Content-Length: nnn
|
| <entry xml:base="http://www.hp.com/cit-rest/service/assets/66863:
|   xmlns="http://www.w3.org/2005/Atom">
|   <id>66863</id>
|   <updated>2008-10-19T15:30:00.000+02:00</updated>
|   <title type="text" xml:lang="en">DSK001256</title> | <summary
type="text"
|   xml:lang="en"> 128 MB DIMM, SDRAM(DSK001256)
| </summary>
| <published>2008-10-19T00:00:00.000+02:00</published>
| <link href="66863" rel="self"/>
| <author>
|   <name>HP Connect-It</name>
| </author>
| <content type="application/xml">
|   <assets>
|     <BarCode/>
|     <Label>DSK001256</Label>
|     <SerialNo/>
|     <Description> 128 MB DIMM, SDRAM(DSK001256)</Description>
|   </assets>
| </content>
| </entry>
```

For example, retrieving the XML representation of an asset from its corresponding URI:

```
GET /cit-rest/service/assets/66863 HTTP/1.1
```

```
Host: www.hp.com
```

```
Accept: application/xml
```

The server response will be similar to the following:

```
| HTTP/1.1 200 OK
| Content-Type: application/xml; charset=utf-8
| Content-Length: nnn
|
| <?xml version="1.0" encoding="UTF-8"?>
| <assets>
|   <BarCode/>
|   <Label>DSK001256</Label>
|   <SerialNo/>
|   <Description> 128 MB DIMM, SDRAM(DSK001256)</Description>
| </assets>
```

Adding an Entry

To add an entry, clients need to issue a POST request to the URI of the feed. For example:

```
http://{host}:{port}/cit-rest/service/{feed}
```

Acceptable response MIME types are:

- application/atom+xml
- application/json
- application/xml
- text/xml

For example, adding an asset to its feed :

```
| POST /cit-rest/service/assets HTTP/1.1
| Host: www.hp.com
| Accept: application/atom+xml
| Content-Type: application/xml; charset=utf-8
| Content-Length: nnn
|
| <?xml version="1.0" ?>
| <assets>
|   <BarCode/>
|   <Label>SRV001156</Label>
|   <SerialNo/>
|   <Description>Hewlett Packard ProLiant BL680c G5 (SRV001156)
|   </Description>
| </assets>
```

The server should in return along with an HTTP code 201, give a 'Location' header as a link to further access it:

```
| HTTP/1.1 201 Created
| Content-Type: application/xml; charset=utf-8
| Location: http://www.hp.com/cit-rest/service/assets/7890
| Content-Length: nnn
|
| <entry xml:base="http://www.hp.com/cit-rest/service/assets/7890"
|   xmlns="http://www.w3.org/2005/Atom">
|   <id>7890</id>
```

```
| <updated>2008-10-19T15:30:00.000+02:00</updated>
| <title type="text" xml:lang="en">DSK001256</title>
| <summary type="text" xml:lang="en"> 128 MB DIMM, SDRAM(DSK001256)
| </summary>
| <published>2008-10-19T00:00:00.000+02:00</published>
| <link href="7890" rel="self"/>
| <author>
|   <name>HP Connect-It</name>
| </author>
| <content type="application/xml">
|   <assets>
|     <BarCode/>
|     <Label>SRV001156</Label>
|     <SerialNo/>
|     <Description>Hewlett Packard ProLiant BL680c G5 (SRV001156)
|     </Description>
|   </assets>
| </content>
| </entry>
```

Updating an entry

To update an entry, clients need to issue a PUT request to the URI of the corresponding entry. For example:

```
http://{host}:{port}/cit-rest/service/{feed}/{entry}
```

Valid content types are:

- application/xml
- text/xml

Acceptable response MIME types are:

- application/atom+xml
- application/json
- application/xml
- text/xml

For example, updating an asset:

```
| PUT /cit-rest/service/assets/7890 HTTP/1.1
| Host: www.hp.com
| Accept: application/atom+xml
| Content-Type: application/xml; charset=utf-8
| Content-Length: nnn
|
| <?xml version="1.0" encoding="UTF-8"?>
| <assets>
|   <BarCode>123456789</BarCode>
| </assets>
```

Server's response:

```
| HTTP/1.1 200 OK
| Content-Type: application/xml; charset=utf-8
| Content-Length: nnn
|
| <entry xml:base="http://www.hp.com/cit-rest/service/assets/7890"
|   xmlns="http://www.w3.org/2005/Atom">
|   <id>7890</id>
|   <updated>2008-10-19T15:38:00.000+02:00</updated>
|   <title type="text" xml:lang="en">DSK001256</title>
|   <summary type="text" xml:lang="en"> 128 MB DIMM, SDRAM(DSK001256)
|   </summary>
|   <published>2008-10-19T00:00:00.000+02:00</published>
|   <link href="7890" rel="self"/>
|   <author>
|     <name>HP Connect-It</name>
|   </author>
|   <content type="application/xml">
|     <assets>
|       <BarCode>123456789</BarCode>
|       <Label>SRV001156</Label>
|       <SerialNo/>
|       <Description>Hewlett Packard ProLiant BL680c G5(SRV001156)
|       </Description>
|     </assets>
|   </content>
| </entry>
```

Deleting an entry

To delete an entry, clients need to issue a DELETE request to the URI of the corresponding entry.

For example:

```
http://{host}:{port}/cit-rest/service/{feed}/{entry}
```

Example code:

```
DELETE /cit-rest/service/assets/7890 HTTP/1.1
Host: www.hp.com
Accept: application/atom+xml
```

As the response, server will return the corresponding deleted entry.

```
| HTTP/1.1 200 OK
| Content-Type: application/xml; charset=utf-8
| Content-Length: nnn
|
| <entry xml:base="http://www.hp.com/cit-rest/service/assets/7890"
|   xmlns="http://www.w3.org/2005/Atom">
|   <id>7890</id>
|   <updated>2008-10-19T15:38:00.000+02:00</updated>
|   <title type="text" xml:lang="en">DSK001256</title>
|   <summary type="text" xml:lang="en"> 128 MB DIMM, SDRAM(DSK001256)
|   </summary>
|   <published>2008-10-19T00:00:00.000+02:00</published>
```



```
| <link href="7890" rel="self"/>
| <author>
|   <name>HP Connect-It</name>
| </author>
| <content/>
| </entry>
```

OpenSearch Descriptor

Issuing a HTTP GET request to the link of the form below returns an OpenSearch document (see <http://www.opensearch.org/Home>):

```
http://{host}:{port}/cit-
rest/service/{feed}?alt=application%2Fopensearchd escription%2Bxml
```

This document informs interested requesters how they may search the feed to obtain a subset of the entire feed containing just interesting results.

For example, retrieving the openSearch descriptor of the 'assets' feed will return the document as illustrated below:

```
| <OpenSearchDescription>
|   <ShortName>Search assets</ShortName>
|   <Description>HP Connect-It 'assets' feed search</Description>
|   <Url type="text/html" template="http://www.hp.com/
|     cit-rest/service/asset?query={cit:whereClause?}"/>
|   <Url type="text/html" template="http://www.hp.com/
|     cit-rest/service/asset?order-by={cit:orderByClause?}"/>
|   <OutputEncoding>UTF-8</OutputEncoding>
|   <InputEncoding>UTF-8</InputEncoding>
| </OpenSearchDescription>
```

The Documents above depict what are the OpenSearch parameters a client may use to refine the 'assets' collection using either :

```
| http://www.hp.com/cit-rest/service/asset?query={queryValue}
| or
| http://www.hp.com/cit-rest/service/asset?order-by={orderByValue}
| or
| http://www.hp.com/cit-rest/service/asset?query=
|   {queryValue}&order-by={orderByValue}
```

The following table shows the OpenSearch parameters that may be available for any feed:

OpenSearch parameters

OpenSearch parameter	Description
query	Corresponds to the 'WHERE' clause directive of the produced document type. If directive already provides a value, then it will be overridden by the OpenSearch parameter value. It follows same syntax as the directive. Example: query=AssetTag LIKE 'UTL%' Available only if related directive is supported.
order-by	Corresponds to the 'ORDER-BY' clause directive of the produced document

OpenSearch parameter	Description
	<p>type. If directive already provides a value, then it will be overridden by the OpenSearch parameter value. It follows same syntax as the directive. Example: <code>order-by=Name DESC</code> Available only if related directive is supported.</p>
watermark	<p>Used to limit the response for entries to only those having a value for 'atom:updated' more recent(after) than the specified date/time. The lexical value follows the XMLSchema datetime datatype specifications. Example: <code>watermark=2007-11-15T21:19:46Z</code> Available only if scheduled pointer is supported on the corresponding produced document type.</p>

Resources Metadata

To determine what is the form of the XML data to be exchanged, clients can retrieve the XMLSchema description of a resource by issuing a GET request to it's schema URI of the form:

```
http://{host}:{port}/cit-rest/schema?feed={feed}&method={method}
```

Where:

- {feed} is the name of the collection
- {method} is the considered HTTP method (either POST,PUT, GET or DELETE).

For instance to retrieve the description of an entry of the feed 'computer' while updating it, clients would issue an HTTP GET request such as:

```
GET /cit-rest/schema?feed=computer&method=PUT HTTP/1.1
```

```
Host: www.hp.com
```

```
Accept: application/xml
```

The server would return the corresponding XMLSchema of the feed for that method:

```
| HTTP/1.1 200 OK
| Content-Type: application/xml; charset=utf-8
| Content-Length: nnn
|
| <?xml version="1.0" encoding="UTF-8"?>
| <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
|   attributeFormDefault="unqualified" elementFormDefault="qualified">
|   <xsd:element name="computer">
|     <xsd:complexType>
|       <xsd:sequence>
|         <xsd:element maxOccurs="1" minOccurs="0" name="AssetTag"
|           type="xsd:string"/>
|         <xsd:element maxOccurs="1" minOccurs="0"
name="ComputerType"
|           type="xsd:string"/>
|         <xsd:element maxOccurs="1" minOccurs="0"
name="ComputerDesc"
|           type="xsd:string"/>
|         <xsd:element maxOccurs="1" minOccurs="0" name="Name"
```

```
|         type="xsd:string"/>
|         <xsd:element maxOccurs="1" minOccurs="0" name="FullName"
|         type="xsd:string"/>
|         <xsd:element maxOccurs="1" minOccurs="0" name="LastUpdated"
|         type="xsd:dateTime"/>
|     </xsd:sequence>
| </xsd:complexType>
| </xsd:element>
| </xsd:schema>
```

Asset Manager Exposure

For the Asset Manager connector, while designing the REST exposure, the required 'atom:id' value is automatically populated depending on the considered produced document type that is to be created. This value should not be changed. In consumption, as a limitation, the collections of a document type are not to be exposed.

Chapter 5

Hewlett-Packard Connectors

There are several connectors associated with Hewlett-Packard products.

HP Asset Manager Connector

Connector type: **CONSUMPTION and PRODUCTION**

The Asset Manager connector is dedicated to the following external applications: Asset Manager and InfraCenter for Workgroups. Connect-It uses the DLLs and APIs provided with these applications. The Asset Manager application client must be installed on the computer on which Connect-It is installed.

Note: For example, the different tables in an Asset Manager database correspond to a single document type published by the Asset Manager connector in the Scenario Builder.

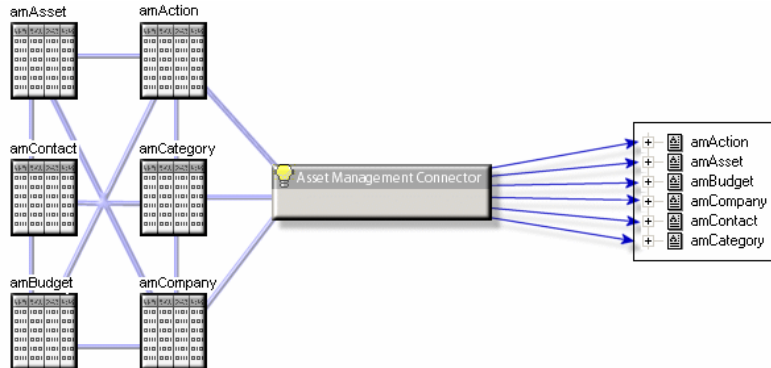


Figure 5-1: Asset Manager connector - Published document types

Remarks concerning the configuration of the Asset Manager applications

amdb.ini file (Asset Manager version 4.4)

If you use Asset Manager version 4.4, you must edit the `amdb.ini` file which describes the connections and add the following parameter: **UseSql92Join=1**.

Amdb.ini file (versions before Asset Manager 4.2.1)

This file contains the list of database connections created on a computer on which the Asset Manager application is installed. This file is created automatically when the Asset Manager application is installed. It is located in the Windows system folder.

In order to access an Asset Manager database from Connect-It, you need to activate the API package during the installation of your Asset Manager application (please read the Asset Manager documentation for more information).

For example:

```
C:\WINNT\amdb.ini
```

When you are configuring your connector, the Connect-It Asset Manager connector reads this file and proposes the list of available connections. If your computer does not have access to Asset Manager, you cannot use the Asset Manager connector.

amdb.ini files (from version 4.2.1 of Asset Manager)

From the version 4.2.1 of Asset Manager, the available connections on a computer are declared in several `amdb.ini` files. These files are located in the following locations:

- Windows system folder if it is a system connection.
- One of the sub-folders of `C:\Documents and Settings` with the name of the Windows users that created the connection, if it is a user connection.

For example:

```
C:\Documents and Settings\Doe\amdb.ini
```

Features tables

Features allow you to extend the Asset Manager data-model. In Connect-It, features appear as fields or collections in the document types published by the Asset Manager connector.

For example, in the **amAsset** document type, the version of DOS used by an asset (a computer, in this case) appears in the **fv_DOSVersion** field.

To fully integrate data from Desktop Discovery and Network Discovery, you must make sure that the features involved in the data transfers have been created correctly in your Asset Manager database. Otherwise, these values given in the documents cannot be imported. The Connect-It datakit includes a text file (`stdfeat.txt`) that you must import into your Asset Manager database. Importing this file automatically creates records in the Features table.

Mandatory fields in an Asset Manager database

In an Asset Manager application, a given field or link may be mandatory by default or have been customized this way by the administrator of the Asset Manager application. In the case of reconciliation, each structure published by the Asset Manager connector corresponds to a record. If an element in this structure is a mandatory field and is not populated, the structure is rejected.

For example, in the Assets table, the **AssetTag** field is mandatory. If the element that represents it in a document type corresponding to the Assets table is not populated, the entire structure is rejected.

API: DLL

To be able to use the Asset Manager connector correctly, you must perform a full or customized installation of Asset Manager, Asset Manager Cable and Circuit, or InfraCenter in which the "API" package is selected.

Managing batches

A special mapping is used for elements managed in batches. The mapping of monitors is performed in the **AddOn** collection and its nature is **MON**. To ensure that identical monitors attached to a given computer are not merged into a single record, this nature is managed as a batch of monitors (the value of the management constraint is 'Asset tag'). In mappings of out-of-the-box scenarios for the Asset Manager connector, the Code element of the **AddOn** collection is mapped with the monitor's SerialNumber field so that the merge mechanism does not apply.

UNIX installation

If you wish to use an Asset Manager database in UNIX, you must follow the installation procedures given in the Asset Manager Installation guide.

Note: Make sure you fully respect the procedures concerning the declaration of environment variables and the creation of the connections file (`amdb.ini`).

Once your Asset Manager database is correctly installed, do the following:

1. Develop your scenario in Windows then adapt the connection settings of your Asset Manager connector to your UNIX configuration.
2. Once your connections are declared and your scenario is created, save it in the `ConnectIt/scenario` folder of the Windows folder.
3. Copy the scenario saved in your `ConnectIt/scenario` in UNIX.
4. You can then create a service (demon) and put your scenario into production.

Remarks on Asset Manager migration

Connect-It is used in conjunction with specialized Asset Manager migration tools that not only transfer data but modify the database structure as well. You must not use Connect-It by itself to convert the old format Asset Manager production database.

Multi-tenancy in Asset Manager

Asset Manager version 9.30 introduces a new architecture optimized for simpler implementation and better support of multi-tenancy functionality. The Asset Manager multi-tenancy architecture supports multiple client organizations with minimum hardware, software, installation and maintenance requirements and costs.

With the Asset Manager multi-tenancy architecture, Managed Services Providers (MSPs) are able to host, maintain and manage assets of multiple customers using a single instance of Asset Manager and the same set of tables hosted on a single copy of a physical database.

The Asset Manager connector in this version of Connect-It supports the multi-tenancy in Asset Manager with TenentID values.

Example using multi-tenancy in Asset Manager

The following graphic illustrates the specification of two multi-tenant IDs for a computer inserted into the Asset Manager database.

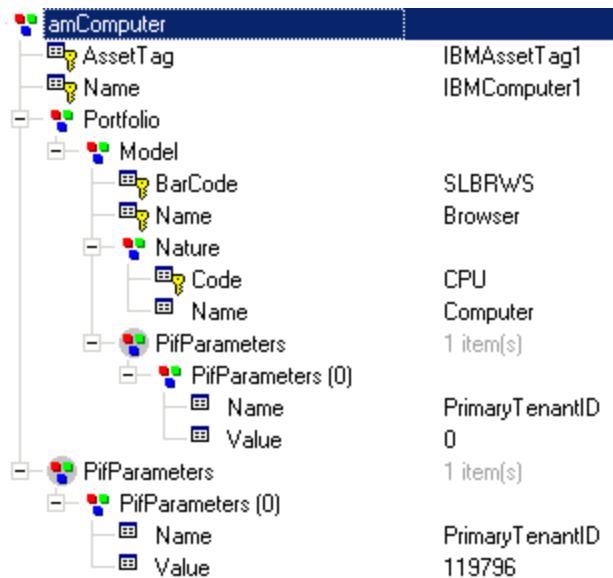


Figure 5-2: Multi-tenancy in Asset Manager

In this example, the document **amComputer** is created in order to insert a new multi-tenant computer into the Asset Manager database using Connect-It. Two multi-tenant IDs will be created, one at the document root, and one for a specific computer model with the value 119796.

- At the root structure of the document, the following are set for the **PifParameters** collection:
 - **Name:** PrimaryTenantID
 - **Value:** 119796
- At the computer mode level, the following are set for the **PifParameters** collection:
 - **Name:** PrimaryTenantID
 - **Value:** 0 (the value of '0' means 'shared')

Note: Be sure to set a value for the root tenant ID. If no value is set for the root tenant ID, the same ID from the last record will be used. If there is no tenant ID, the primary tenant ID of the user will be used.

For more information about multi-tenancy, see the Asset Manager Multi-Tenancy Guide available on the HP Support website: <http://support.openview.hp.com/selfsolve/manuals>.

Limitation of the HP Asset Manager connector

- The Asset Manager applications do not process dates less than (before) 1970 or greater than (after) 2038.
If a scenario between an Asset Manager connector and another connector uses a date that does not match these criteria (such as a date used in a mapping script), an error message will appear during the scenario's execution.

Using Older Versions of Asset Center

If you are using Asset Center 5.0x (and previous versions) with Connect-It 3.9x (and later versions), note the items below.

If Asset Center connections do not appear in Connect-It, try the following:

1. Close Connect-It completely.
2. Create the following folder:
`%APPDATA%\HP\AssetManager\conf.`
3. Move the existing `amdb.ini` file from the Windows folder (for example, `C:\WINNT\amdb.ini`) into the newly created folder.
4. Start Connect-It.

The connections you created in Asset Center should appear. You can also configure the connections in Asset Center (if not configured yet), and then configure the same connections in Connect-It. The same connections should appear in both Asset Center and Connect-It.

Out-of-Box Scenarios

See [HP Asset Manager Connector Scenarios](#).

Configuring the HP Asset Manager Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Caution: *This page is only displayed if the Display advanced configuration wizards option is selected.*

Dynamic library

This option enables you to specify the full path of the DLL containing the APIs associated with your Asset Manager application.

As an example, for Asset Manager 4.4:

```
C:\Program Files\HP\AssetCenter 4.40 en\bin\amapi44.dll
```

By default, the connector automatically locates the DLL to use.

Note: *If using Asset Manager 9.30 on UNIX, you must provide the following library:*

```
~/AssetManager/bin/libcrypto.so
```

Expose the primary key of the tables

If you select this option, the document types published by the Asset Manager connector contain fields that correspond to the primary keys of the database tables. Example: If you select this option, the **IAstId** fields appears in the **amAsset** document type published by the connector. The identifier of each record in the Assets table is the primary key of this table.

Expose the foreign keys of the tables

If you select this option, the document types published by the Asset Manager connector contain fields that correspond to the foreign keys of the database tables.

Executing reconciliation scripts

See [Defining Consumption Directives](#).

Executing an initial import

This option lets you avoid having to use SELECT queries in reconciliation scripts for initial imports (for example, when the database is empty or the items to be inserted do not yet exist). This shortens processing time. No SELECT will be performed on:

- A root element
- A 1-1 link if the **Follow the link** option is enabled and the parent document does not exist
- A 1-N link if the parent element was just inserted

Advanced Options

This page enables you to specify details for certain elements of consumed document types. An advanced option enables you to define full name structures (**FullName**) for each document type, only if the full names used are not the standard names. For information, a **FullName** field is a unique field generated by the application.

For example, for the Employees table (**amEmplDept**), the option is populated as follows:

```
amEmplDept.FullNameDef  
lParentId,Name,bDepartment,[Name],[First-Name],[IdNo]
```

The structure defined in the value field of the option is the following:

- **lParentId**: Hierarchic field
- **Name**: Hierarchic field used to create the hierarchy
- **bDepartment**: Test field enabling you to know whether the record is last record linked to the full name
- **[Name]**, **[FirstName]**, **[IdNo]**: Structure of the last record of the full name. The fields must be in square brackets [...].

Production Directives

For information about how to enter a connector's production directives, refer to the chapter [Connector Directives](#).

Sysdate field

In the WHERE clauses written in AQL, the use of the **sysdate** field must be replaced with the function **getdate()**. For example, the SQL query

```
SELECT * from AmAsset where dinstall>sysdate
```

must be replaced with the AQL query

```
SELECT * from AmAsset where dinstall>getdate()
```

For example, if you wish to recover all record modified since less that a day:

```
DaysDiff(GetDate(), dtLastModif) <= 1
```

Outer joins

This option only functions with Asset Manager 4.3 and higher. If the produced document type contains a non-exclusive link (corresponding to the **LEFT OUTER JOIN** SQL function) then the **Make an outer join** is available. This option enables you to recover all the records from a table for which the link is non-exclusive.

For example, for a mapping between the Computers table and the Employees table recovering all the computers assigned to an employee, the behavior is the following:

- Option not selected: Only the computers assigned to an employee are recovered
- Option selected: All the computers are recovered including those that are not assigned

Follow link

This option can be enabled in the **Reconciliation** tab for any selected structure. This option is useful in two cases:

- For overflow tables
The specificity of these tables is that they are linked to the reference tables by a 1-1 link. For this reason, using a reconciliation key is optional. Following the link of a item in a reference table enables you to retrieve information from this linked table.
- When you cannot define a reliable reconciliation key (for example, if you cannot perform a reconciliation using the Employee ID but only with their Name and First Name).

The **Follow link** option is particularly useful for the Portfolio Items table (**amPortfolio**) and enables you to follow the link between this table and the Assets table (**amAsset**) without having to define a reconciliation key for a given element.

Note: We recommend using this option in Update or insert mode.

Collection-to-collection mapping

- When mapping N-N links (collection-to-collection mapping), the **Use the parent ID as a reconciliation key** option is selected by default. Unselect this option in order to avoid errors with this type of mapping.
- During advanced reconciliation for a collection-to-collection mapping the **Minimize the number of requests when reconciling collections** option activates the following behavior:
 - A SELECT type query is performed for the set of elements that makes up the collection.

If this option is not activated, a query is performed for each of the elements that makes up the collection.

Consumption directives of the Asset Manager connector

For information about how to enter a connector's consumption directives, refer to [Defining Consumption Directives](#).

To set a connector's consumption directives, you must enter reconciliation to parameters in the **Reconciliation** and **Advanced reconciliation** tabs.

Document type produced by the Asset Manager connector

The document type produced by the Asset Manager connector exposes Asset Manager tables. In particular, the produced document type accounts for the normalization functionality developed for Asset Manager version 4.4. This functionality can be seen in the scenario provided for the HP Desktop Inventory 8 connector.

Since Asset Manager version 4.4, it is possible to associate software installation models and their names with the results of an inventory from an external application; For example, this enables you to associate the names of software models (from an inventory tool) with their internal names in Asset Manager (from a catalog, for example). This is defined in the **amInventModel** table and is based on a reconciliation key created from the name of the external application and a unique

identifier of the software (PDI|1234, for example). This key also includes the name of the software and its version.

The scenario `pdi/pdi8ac44/pdiac.scn` uses this new normalization functionality. The software model is mapped to the **InventModel** link in the **AddOn** collection. Each unknown inventory model is recorded in the Software Installations table (`amSoftInstall`) and linked to a temporary model `sysUNKNOWN_SOFT` thus creating a normalization proposal; The normalization proposal is processed using a wizard. In this way, you associate a temporary model with an existing definitive model in Asset Manager. Once the normalization process is completed, the update is propagated to all inventoried applications linked to the computer.

Note: *You must create a reconciliation script including the **PiflignoreSubDocumentReconc** function when performing updates so that the definitive model does not get reset to **sysUNKNOWN_SOFT**.*

The reconciliation cache

The reconciliation cache reduces the number of database queries. The cache is stored in memory. The reconciliation cache is useful when:

- constants are used in a mapping
- the mapping concerns relatively small tables such as the Models (`amModel`) or Natures (`amNature`) tables
- the mapping concerns tables that will not be updated

When a record is found in the reconciliation cache, it is assumed that the contents of the document has not changed; As a consequence, this record is not updated.

The cache is purged when the maximum number of documents is reached. This number is defined in the connector options (**Edit > Options > Connector > Maximum number of documents in the reconciliation cache** menu).

In the **inventorySrc-amComputerDst** mapping of the scenario `pdiac.scn`, the reconciliation cache is used for the **Model** structure. We assume that the Asset Manager models themselves remain unchanged but the related information changes (in this case, the asset).

HP Business Service Management Dashboard / HP Universal CMDB Connectors

Connector type: **CONSUMPTION** and **PRODUCTION**

These connectors enable data related to MAM or BAC. These connectors are described in the ServiceCenter-MAM/BAC Integration Solution guide which is supplied with Connect-It.

Out-of-Box Scenarios

See [HP Business Service Management Dashboard Connector Scenarios](#).

HP Client Automation Inventory Manager Connector

Connector type: **PRODUCTION**

The HP Client Automation Inventory Manager connector enables you to process data from an HP Client Automation Inventory Manager database.

Out-of-Box Scenarios

See [Configuration Management 5.1 / Client Automation 7.x Scenarios](#).

Configuring the HP Client Automation Inventory Manager Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

This connector supports OVCM 5.1 and HPCA 7.5.

Note: See the [Configuration Management 5.1 / Client Automation 7.x Scenarios](#) topic for the Client Automation scenarios.

HP Client Automation Management Portal Connector

Connector type: **CONSUMPTION and PRODUCTION**

The HP Client Automation Management Portal connector enables you to process data from an HP Client Management Portal database.

Out-of-Box Scenarios

See [Configuration Management 5.1 / Client Automation 7.x Scenarios](#).

Configuring the HP Client Automation Management Portal Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

This connector supports OVCM 5.1 and HPCA 7.5 (selected by default). Schema files are located in the Connect-It installation folder:

- HPCA 7.5: config\hpovcm\xsd\CA75_MP_Web_Services.xsd

Configuring the connection to HP Client Automation Management Portal

Populate the following fields:

- **URL:** Address to the server's Web services. For example:
`http://RadiaServer:3466/proc/Radia`
- **Login:** User name encrypted and encoded using the `NVDKIT.exe` tool.
Note: The NVDKIT.exe tool can be found in the HP Client Automation installation folder.
- **Password:** Password encrypted using `NVDKIT.exe` following the same procedure as for the user name.

There are three options depending on the HP Client Automation configuration. See the examples below with admin as the user name and secret as the password:

Option	Command line
Encrypt with AES and encode	<pre>\$nvdkit.exe % Base64_Encode [password encrypt admin aes] % Base64_Encode [password encrypt secret aes]</pre>
Encrypt with DES and encode	<pre>\$nvdkit.exe % Base64_Encode [password encrypt admin des] % Base64_Encode [password encrypt secret des] Or \$nvdkit.exe % Base64_Encode [password encrypt admin] % Base64_Encode [password encrypt secret]</pre>
Encode only	<pre>\$nvdkit.exe % Base64_Encode admin % Base64_Encode secret</pre>

Specifying the configuration files (advanced mode)

Populate the following fields:

- **XML schema:** Path to the HP Client Automation Management Portal Web services XML schema.
- **Properties file:** Path to the properties file.

Note: See the [Configuration Management 5.1 / Client Automation 7.x Scenarios](#) topic for the Client Automation scenarios.

HP Client Automation Service Events Connector

Connector type: **PRODUCTION**

The HP Client Automation Service Events connector enables you to process data from a HP Client Automation Service Events database.

Out-of-Box Scenarios

See [Configuration Management 5.1 / Client Automation 7.x Scenarios](#).

Configuring the HP Client Automation Service Events Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

This connector supports OVCM 5.1 and HPCA 7.5.

Prerequisites

Before configuring your connector, make sure a valid connection to your ODBC database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and password.

Note: See the [Configuration Management 5.1 / Client Automation 7.x Scenarios](#) topic for the Client Automation scenarios.

HP Client Automation Usage Manager Connector

Connector type: **PRODUCTION**

The HP Client Automation Usage Manager connector lets you process data from a Usage Manager database.

Out-of-Box Scenarios

See [Configuration Management 5.1 / Client Automation 7.x Scenarios](#).

Configuring the HP Client Automation Usage Manager Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

This connector supports OVCM 5.1 and HPCA 7.5.

Prerequisites

Before configuring your connector, make sure a valid connection to your ODBC database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and password.

Note: See the [Configuration Management 5.1 / Client Automation 7.x Scenarios](#) topic for the Client Automation scenarios.

HP Cloud Service Automation Connector

Connector type: **CONSUMPTION and PRODUCTION**

The HP Cloud Service Automation (CSA) Connector provides a set of document types of CSA catalog modules. With this connector, you can retrieve from CSA the information of organization, person, catalog, subscription, service instance, service offering, service requests, etc. The retrieved catalogs and subscriptions can be information of certain user in an organization, or the information of one or multiple organizations.

The out-of-box document types in the HP Cloud Service Automation connector:

- RetrieveApprovalProcess
- RetrieveCatalog
- RetrieveOrganization
- RetrieveServiceInstance
- RetrieveServiceOffering
- RetrieveServiceRequest
- RetrieveServiceSubscription

Out-of-Box Scenarios

See [HP Cloud Service Automation Connector Scenarios](#)

Configuring the HP Cloud Service Automation Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Specify the server information

- **Server** Enter the CSA server name in `<server address>:<port>/csa/rest` format.
- **Login** Enter the username of the CSA administrator role.
- **Password** Password for this login.
- **Organization name** The name of the organization from which you want to retrieve data.
- **User name** The user name belongs to certain organization. Note that a user name of the administrator role can retrieve all the catalogs and subscriptions of the organization; a non-administrator role can only retrieve data the user allowed to view.
- **Test** Click the button to test the connection to the CSA REST API and verify the logon information.
- **Compression support for HTTP request and response**
 - **Compress requests** If the option is selected, the connector sends the compressed HTTP request that contains a HTTP header `Content-Encoding:gzip`.
 - **Accept compressed responses** If the option is selected, the connector sends the HTTP request that contains a HTTP header `Accept-Encoding:gzip`, which tells the server that the connector allows a compressed response.

Note: You must specify a consumed document type in the **Produced document types** edit window.

HP Discovery and Dependency Mapping Inventory Connector

Connector type: **PRODUCTION**

For additional information on customizing and using the Discovery and Dependency Mapping Inventory tool, refer to the separate documentation. The Discovery and Dependency Mapping Inventory connector is used to process an HP Discovery and Dependency Mapping Inventory database. A Discovery and Dependency Mapping Inventory database contains information about the enterprise's entire IT portfolio. For each computer stored in the database, the out-of-the-box scenarios transfer the data concerning those records to an Asset Manager or ServiceCenter database.

Out-of-Box Scenarios

See [HP Discovery and Dependency Mapping Inventory Connector Scenarios](#).

Configuring the HP Discovery and Dependency Mapping Inventory Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that

section, they are available below.

To use a MySQL native connection, you must copy the MySQL dynamic library (DLL) client to the bin folder of Connect-It's installation folder (or to the operating system's system32 folder).

Configuring the connection

- **Server:** specify the port used to connect to the server. If Connect-It was installed on a workstation other than the Discovery and Dependency Mapping Inventory server, populate this field with the DNS or the IP address of the Discovery and Dependency Mapping Inventory server; the DNS name or the IP address comes before the colon. For example:
`server.company.com:8108`
`127.0.0.1:8108`
- **User:** Specify the login. You are advised to use the admin login, which has all authorizations. The login used must have the required read and write authorizations; the login profile is defined in the Discovery and Dependency Mapping Inventory Web user interface.
- **Password:** specify the password for this login.
- **Database:** specify the name of the database used. The default name is Aggregate.
- **Detect the version:** Tests the HP Discovery and Dependency Mapping Inventory version and automatically configures the connector.

SSL configuration (advanced mode)

If you select **MySQL (native)** for the **Connection type**, you can enable SSL for secure communication:

- **Use an SSL connection:** select this option to use SSL, and choose the following files:
 - **Client key file:** the client key file to be used
 - **Certificate file:** the certificate file to be used
 - **CA certificate file:** the CA certificate file to be used

Define document types (advanced mode)

Fill in the extension file according to your version of Discovery and Dependency Mapping Inventory.

Published document-type

The Discovery and Dependency Mapping Inventory connector publishes:

- A **Devices** document type:
This document type publishes the inventory information contained in an Discovery and Dependency Mapping Inventory database.
- A **DeviceStructure** document type:
This document type publishes inventory information involving sub-components of a network element that require individual tracking such as a chassis or a module. These sub-components must have a serial number and be managed using a **Unique asset tag** in Asset Manager.

OpenView Configuration Management Connectors

- **OpenView Configuration Management 5.1 Inventory Manager connectors**
These connectors enable you to process data from an OpenView CMI Inventory Manager

database.

Connector type: **PRODUCTION**

- **OpenView Configuration Management 5.1 Service Events connectors**

These connectors enable you to process data from a OpenView CM Service Events database.

Connector type: **PRODUCTION**

- **OpenView Configuration Management 5.1 Usage Manager connectors**

These connectors enable you to process data from a Usage Manager database.

Connector type: **PRODUCTION**

- **OpenView Configuration Management 5.1 Management Portal connectors**

Connector type: **CONSUMPTION and PRODUCTION**

Out-of-Box Scenarios

See [Configuration Management 5.1 / Client Automation 7.x Scenarios](#).

Configuring OpenView Configuration Management Connectors

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Prerequisites

For the Service Events and Usage Manager connectors, note the following:

Before configuring your connector, make sure a valid connection to your ODBC database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and password.

Configuring the Management Portal connector

Configure the connection to OpenView CM Management Portal.

Populate the following fields:

- **URL:** Address to the server's Web services. For example:
`http://RadiaServer:3466/proc/Radia`
- **Login:** User name encrypted using the Radia tool. For example:

```
$ nvdkit.exe  
% password encrypt login  
{DES}XXXXXXXXX:X  
% Base64_Encode
```
- **Password:** Password encrypted using the Radia tool following the same procedure as for the user name.

Specify the configuration files (advanced mode) Populate the following fields:

- **XML schema:** Path to the OpenView CM Management Portal Web services XML schema.
- **Properties file:** Path to the properties file.

HP DecisionCenter Connector

Connector type: **CONSUMPTION and PRODUCTION**

The DecisionCenter connector is for use with DecisionCenter: a business intelligence application that enhances your ability to make decisions about your IT infrastructure.

Out-of-Box Scenarios

See [HP DecisionCenter Connector Scenarios](#).

Configuring the HP DecisionCenter Connector

Note: We highly recommend using the connector with an ODBC driver and ODBC Administrator using identical version levels of their DLL files.

Production directives of the connector

For information about how to enter a connector's production directives, see [Defining Production Directives](#).

Sysdate field

In the WHERE clauses written in AQL, the use of the **sysdate** field must be replaced with the function **getdate()**. Example: The SQL query

SELECT * from AmAsset where dinstall>sysdate

must be replaced with the AQL query

SELECT * from AmAsset where dinstall>getdate()

NULL

When a "Numeric" type field is not populated (its value is NULL), Connect-It sets its value to "0". Similarly, the absence of a link is comes out as "Link = 0" or "foreign key = 0". Example: "Location=0" or "ILocald=0".

Consumption directives of the HP DecisionCenter connector

To set a connector's consumption directives, you must enter reconciliation parameters in the **Reconciliation** and **Advanced reconciliation** tabs.

Published document-type

The document types published by the DecisionCenter connector correspond to tables in the DecisionCenter database.

HP Network Node Manager i-series Web Service Connector

Connector type: **CONSUMPTION**

The Network Node Manager i-series (NNMi) Web Service connector lets you interact with a Network Node Manager i-series Web service. In an integration scenario, the NNMi Web Service connector consumes a document and sends it in the form of a query to a Web service. Then it receives a reply which it automatically transforms into a produced document.

Note: This connector only supports the NNMi WS-I APIs. It does not support NNMi WS-Management APIs.

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Specifying the WSDL

This page enables you to enter the connector's connection parameters to a Web service.

- **WSDL address** field: The WSDL URL address enables you to retrieve for each Web service:
 - The communication protocols used
 - The supported operations.
 - The format of these operations
 - The data consumed and the data produced.
- **User Name** field: User identifier
- **Password** field: User defined

For example: `http://localhost/NodeBeanService/NodeBean?wsdl`

Filtering the Connector

This section provides an example for filtering the NNMi Web Service connector using the following logic: **conditionA && (conditionB || conditionC)**.

The following is an example procedure using document type **getNode** in `http://localhost/NodeBeanService/NodeBean?wsdl`:

1. Edit the node **arg0** under **getNode** and chose a node.
Note: You can only edit one filter type: condition, constraint, or expression.
2. Use the **AND** operator and note the following:
 - There will be two **conditionsubFilters(0)** and **subFilters(1)**.
 - Add values to the condition for **subFilters(0)** only.
3. In **subFilters(1)**, use the **OR** operator and note the following:
 - Under **subFilters(1)**, there will also be two additional **conditionsubFilters(0)** and **subFilters(1)**.
 - You can add values to the conditions for both **subFilters(0)** and **subFilters(1)**.

In the following figure, the logic is as follows: **DeviceModel EQ hplatinum AND (longName LIKE %fc.hp.com OR longName LIKE %boi.hp.com)**.

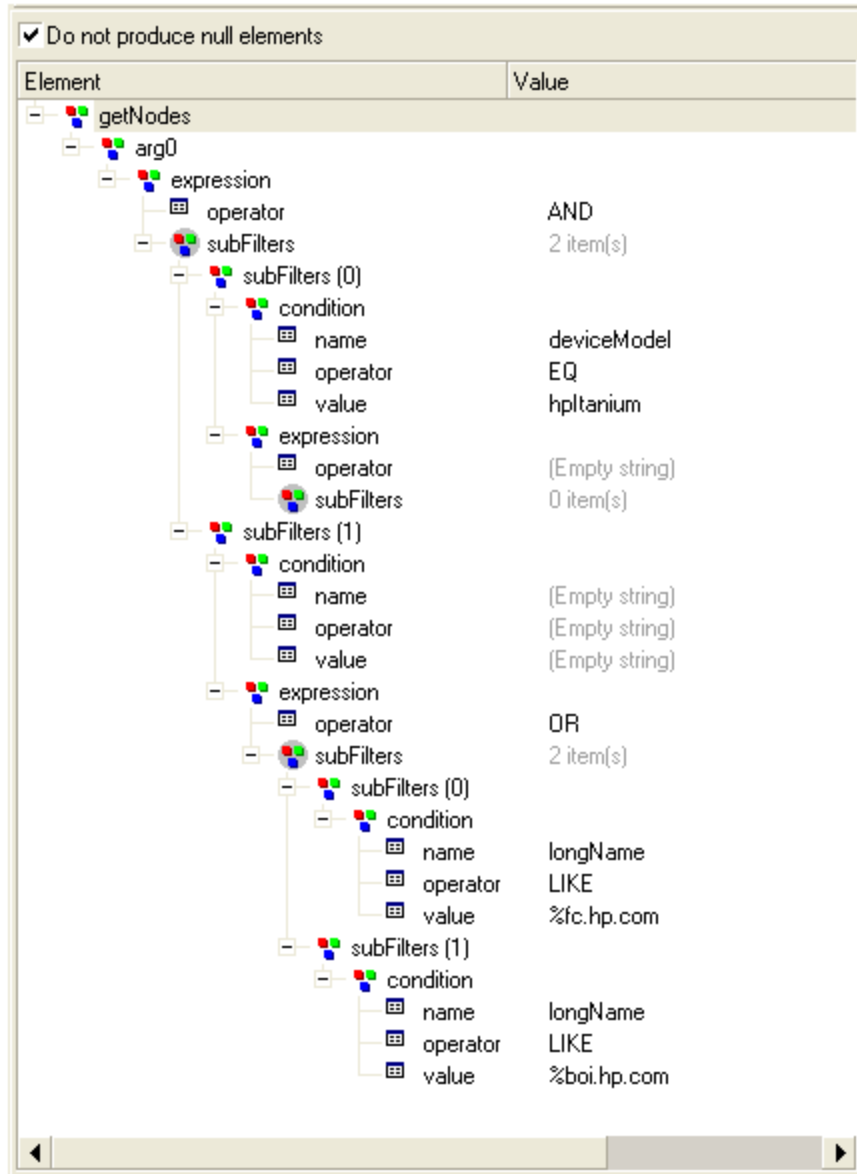


Figure 5-3: Filtering the NNMi connector

Out-of-Box Scenarios

See [HP Network Node Manager i-series Web Service Scenarios](#).

HP Operations Manager for Unix Connector

Connector type: **PRODUCTION**

Only the HP Operations Unix connector is supplied with Connect-It. However, a scenario for HP Operation Windows that works with the delimited text connector is available. This connector exposes the following main tables of a HP Unix application:

- OPC_NODES
- OPC_ACT_MESSAGES
- OPC_HIST_MESSAGES

Out-of-box Scenarios

See [HP Operations Manager Scenarios](#).

Configuring the HP Operations Manager for Unix Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Published document types

The published documents are the following:

- **Node:** Interpretation of the OPC_NODES table
- **ActiveMessage:** Interpretation of the OPC_ACT_MESSAGES table
- **MessageHistory:** Interpretation of the OPC_HIST_MESSAGES table

HP Project and Portfolio Management Center Connector

The HP Project and Portfolio Management Center 7.5 (PPM) connector enables a connection to a PPM web service. The PPM connector consumes a document (from the mapping box) and sends a request to the web service. The response (from the web service) is transformed into a produced document and sent to the mapping box.

Out-of-Box Scenarios

See [HP Project and Portfolio Management Center Scenarios](#).

Configuring HP Project the Project and Portfolio Management Center Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Define the connection parameters

Specify the parameters for the connection to the PPM server.

- **Server:** Name or IP address of the PPM server
- **Port:** Port of the PPM server
- **Context path:** Context path of the PPM web application
Default context path is: itg/ppmservices

- **Service name:** Name of the PPM service.
Available web services are: DemandService, FinanceService, ProjectService, ResourceService and TimeService.

Define the security parameters

Specify the security parameters (server security configuration) for the connection to the PPM server. Select Use HTTP Basic Authentication if transport-level security is enabled on the server side.

- **Login:** HTTP Basic Authentication PPM username
- **Password:** HTTP Basic Authentication PPM password
The Login and Password you enter must match the settings found in the server configuration file `<PPM_HOME>/server/<PPM_Server_Name>/deploy/itg.war/WEB-INF/conf/axis2.xml` in the Rampart section:

```
<parameter name="userName" locked="false">Login</parameter>
<parameter name="password" locked="false">Password</parameter>
```


Select Use WS-Security if message-level security is enabled on the server side.
- **Configuration path:** path to `client.jks`, `client.properties` and `client-config.wsdd` files.

A configuration is provided in `<CIT_HOME>\datakit\ppm\config` containing default `client.jks`, `client.properties` and `client-config.wsdd` files. Refer to the WS-Security client configuration on your PPM server. Can be found under `<PPM_HOME>/server/<PPM_Server_Name>/deploy/itg.war/WEB-INF/conf` folder.

- **User:** WS-Security username
- **Password:** WS-Security password

HP Service Desk (Inbound) Connector

Connector type: **CONSUMPTION**

The HP Service Desk connector (inbound) is used to process .xml files for an HP Service Desk application.

Configuring the HP Service Desk (Inbound) Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Select files or folders

This screen enables you to select the XML file or files that your connector must write. Before defining the connector's behavior, specify the folder for writing files (**Folder name** field). The path for this folder is used in a mapping in combination with the **UrlFileInfo.Path** element. You can choose between the following behaviors:

- Write documents to one single file
Enter the path of a folder on your computer or your network (**Folder name** field). The document will be overwritten or left alone depending on the parameters defined in the next page of the

wizard.

- Write to a different file for each document
Specify the folder in which the files will be written (**Folder name** field). The name of these files will correspond to the generic name that you have entered in the consumption directives of the XML connector. Refer to [Consumption Directive of the XML Connector \(write\) - FTP mode](#). You can also define the names of these files by mapping the element **UriFileInfo.Path**. For example, create a file with the name of each employee in Asset Manager. In this case, the path defined in the wizard page is concatenated with the path defined in the mapping.

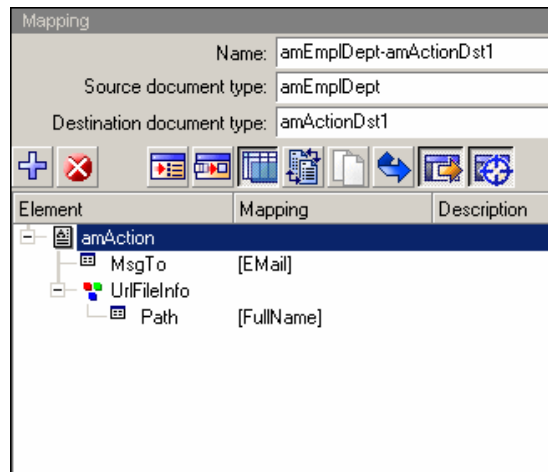


Figure 5-4: Defining a path in the mapping

In order for the path defined in the mapping to be used as the default path, and as a result to avoid using the path defined in the wizard, use the '/' character in the mapping.

Note: This behavior does not prevent you from specifying a valid path in the connector configuration wizard.

If the path specified for the **UriFileInfo.Path** element does not exist, it is created automatically provided that:

- the path uses part of the path specified in the configuration wizard.
- the path does not add more than one level to the specified tree. For example, for a path *a/b/c* specified in the configuration wizard, only a path *a/b/c/d/filename*, specified in the mapping, is possible.

Behavior between two sessions

This page enables you to select how documents are processed between two data-writing sessions.

Append to the same file

The connector starts writing data to the open file again from where the previous session was interrupted.

Overwrite the last file

The connector deletes the file to which it wrote data during the previous session.

Number the different files

The connector creates a file whose name is the same as the previous file, but incremented by 1 unit. Example: `file.xml`, `file1.xml`, `file2.xml`, etc.

HP Service Desk (Outbound) Connector

Connector type: **PRODUCTION**

The HP Service Desk connector (Outbound) is used to process data from an HP Service Desk database.

Out-of-Box Scenarios

See [HP Service Desk \(Outbound\) Connector Scenarios](#).

Configuring the HP Service Desk (Outbound) Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Advanced configuration (advanced mode)

- **SQL92 supported:** if you use the Asset Manager Database ODBC driver, disable this option.
- **Execute an initial import:** This option lets you avoid having to use SELECT queries in reconciliation scripts for initial imports (for example, when the database is empty or the items to be inserted do not yet exist). This shortens processing time.

HP ServiceCenter and Service Manager Connector

Connector type: **CONSUMPTION and PRODUCTION**

The ServiceCenter connector is used to retrieve data from or insert data into a ServiceCenter database. Within a given scenario, you can use multiple ServiceCenter connectors connected to the same server or to different versions of the server.

Creating input events

The ServiceCenter connector cannot directly consume documents because ServiceCenter does not authorize the direct updating of records for data integrity reasons. Instead, a corresponding input event must be created in ServiceCenter. When the documents are consumed, this event is placed in an event queue. Next, a mapping in ServiceCenter maps the elements of this event to fields in ServiceCenter.

For example, by default, there is no input event corresponding to the description of software installed on a computer. In order for information concerning software to be taken into account by the ServiceCenter connector, you need to do the following:

1. Create a synchronous input event for these software items (called **pcsoftware** in the supplied scenario).
2. Map the fields of the input event to those in the **pcsoftware** table.

The Connect-It datakit includes a file (`pcsoftware.unl`) that allows you to perform these two actions automatically in ServiceCenter. A UNL file is available for each out-of-the-box scenario

involving ServiceCenter. Please contact **Hewlett-Packard Development Company, L.P.** technical support. For details on how to import these files, refer to the ServiceCenter documentation.

Input event synchronization

Input events in ServiceCenter are processed either synchronously or asynchronously. In asynchronous mode, ServiceCenter defines how the event queue is managed. In synchronous mode, the events are processed as soon as they are inserted in the ServiceCenter event queue. When developing your scenarios, use the synchronous mode if you want to verify immediately that your scenario is working. Selecting the asynchronous processing mode prevents you from immediately verifying in ServiceCenter that the data has been processed correctly: A warning message is issued to indicate that the document is pending processing. To verify the errors, verify the input queues in ServiceCenter. In synchronous mode, it indicates whether a document has been processed successfully or rejected.

Note: *Performance is improved when working in asynchronous mode (improvements in excess of 50%). It is recommended using synchronous mode if performance levels are deemed suitable.*

Processing an synchronous mode will produce warnings in the document log.

Join files

In ServiceCenter, join files let you display fields from multiple tables in one single screen. For example, the **Device** and **Computer** tables contain fields exposed in the **joincomputer** join file. These join files are declared in a configuration wizard and are visible in the produced document types.

Using the ServiceCenter connector in Unix

To use the ServiceCenter connector in Unix:

1. Create the following text files:
 - `sc.ini`
 - `sc.log`
2. Write **log=sc.log** in the `sc.ini` file.
3. Place these two files in the `[Connect-It installation folder]/bin` folder.

Using the ServiceCenter connector on Linux

When Connect-It runs on Linux or Solaris, the path to `bin/sc6` located in the Connect-It installation folder must be added to the **\$LD_LIBRARY_PATH** variable before connecting to a ServiceCenter 6 database. Please note that several C++ and gcc system libraries are present in this directory. Changing the **\$LD_LIBRARY_PATH** variable may cause other applications to execute improperly. We recommend that you limit the change to the **\$LD_LIBRARY_PATH** variable to only include the shell used to execute Connect-It or to execute Connect-It via a script that sets the **\$LD_LIBRARY_PATH** variable.

sccl32.dll

When updating Connect-It, delete the `sccl32.dll` file located in the `c:\windows` or `c:\windows\system32` folder.

Compatibility with Service Manager 9.2

An option is available in the connector's configuration wizard to ensure the compatibility of the connector with Service Manager databases.

Note: For this version, Connect-It is unable to determine which version of Service Manager you are connecting to. If you are using Service Manager version 9.2 or later, you must select a check box on the connector configuration screen. If you are using a customized extension file, you will also need to modify it.

Out-of-Box Scenarios

See [HP ServiceCenter and Service Manager Connector Scenarios](#).

Configuring the HP ServiceCenter and Service Manager Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Certain options concerning this connector can be enabled using the **Edit > Options > Connector** menu. For more information, refer to the User's Guide.

Defining the connection parameters

- **Server name:**

Enter the name of the server in one of the following ways:

- [ServiceCenter Server Name].[Port number of the ServiceCenter client]. For example:
SC.hp.com.12670.
- [Service Manager Server Name].[Port number of the Service Manager client]. For example:
SM.hp.com.13080 or SM.hp.com.12690

Note: Check the Service Manager port number. The default port numbers are as follows:

- Version 9.20: 13080.

This is the httpPort value, which is set in the following file: {Service Manager installation folder}\Server\RUN\sm.ini.

- Versions prior to 9.20: 12690.

This is the -listener value, which is set in the following file: {Service Manager installation folder}\Server\legacyintegration\RUN\sc.cfg.

Note: the configuration file might be located in a different path for the version of Service Manager that you are using. Contact your Service Manager administrator for the correct path.

If you are unable to write to the Service Manager database or have other problems with the Service Manager connector, you might have the incorrect port number.

- **Login**

Enter the login required to connect to the ServiceCenter server. The profile of this login must allow you to execute the actions performed by your scenario (reading data or sending input events).

- **Password**

Enter the password associated with the login.

- **Service Manager Version**

If using Service Manager 9.2 or later, select Service Manager 9.20 and later versions. Also select whether or not to write to the Service Manager database.

- **Test**

This button enables you to test your connection. To test your connection, enter your connection parameters and click Test. The Test the connection window appears and tells you whether the connection succeeded or failed. If it failed, there will be messages explaining the cause.

Writing to a SM7 database

This option is used to change the behavior of the connector in order to write to a Service Manager 7 database. In order to operate correctly, this option requires:

- That a ServiceCenter 6.2 server and a Service Manager 7 server access the same Service Manager database.
- That the ServiceCenter 6.2 server be set to read-only for this database.

When this option is enabled:

- All of the connector's write operations go through the Service Manager 7 API. Read operations go through the ServiceCenter client API.
- The server port must be specified (Port field of the SM7 server).

This option was developed to ensure the compatibility of existing integration scenarios. We recommend that you NOT use this option to develop new integration scenarios for a Service Manager database.

Caution: *As the ServiceCenter dynamic library is approaching the end of its life cycle, we recommend using the new Service Manager web service API.*

Configuring Advanced Mode Settings for the HP ServiceCenter and Service Manager Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

This advanced mode settings enables you to set the following parameters:

- List of virtual tables: This field lets you select the linked tables, which are defined in the joindefs table, that will be displayed in read-only mode. By default, all linked tables starting with join are displayed. Enter a list of values separated by semi-colons (;). Wildcard characters (*, \$, ?) are authorized.

Note: *If an event references a linked table in a definition of a structure collection, then the table is automatically loaded (and exposed to be read), even if it has not been specified in the connector configuration wizard.*

- Format for date and time type fields of input events. Select this option if the format of input events is different from that of the server.
- Format for date and time type fields of output events. Select this option if the format of output events is different from that of the server.

Production directives of the ServiceCenter connector

These directives use WHERE and ORDERBY clauses, which allow you to filter the produced documents. The following table lists the available operators and operands for these two clauses.

Operators:

- =
- ~=
- >
- <
- AND
- OR
- NOT
- #

Operand type	Examples
Number	1 (without quotes)
Character	"a" (double quotes)
Date	'2002-01-10' (international standard)
String	"string" (double quotes)

For example: The **devicepc** document type represents the equivalent table in ServiceCenter in order for the connector to only produce documents representing Compaq computers:

- In use since January 1, 2001
- Still on the network monitored by Network Discovery

Use the following WHERE clause:

(as an example) `vendor = "Compaq" and Instal.date > '2000/01/01' and Ind.removed = 0`

For the **ORDERBY** clause, you must indicate which fields are to be used to perform an ascending sort (alphanumeric) of the documents. Separate these fields with commas. For example: For a produced e-mail document-type, you can sort these documents by recipient, then author, with the following **ORDERBY** clause: `user.to, user.from`

Directives for consumed documents

You can define reconciliation keys for document types consumed by the connector. If no reconciliation key is defined, the connector's default behavior is used (processing by event).

Note: *Reconciliation keys can only be defined for first-level elements. Lower level elements included in structures or collections may not have a reconciliation key.*

An event is generated from the document type that holds the reconciliation key. The ServiceCenter application ensures that integrity rules are followed for changes made by the user. Generally, the

field that holds the reconciliation key is used for the reconciliation procedures. In most cases, a field used as a reference key for events exists. This field is defined in ServiceCenter and is mandatory. If no value is provided for this field, a random value based on the concatenation of fields that make up the reconciliation key set is created.

Note: *The logical.name field is generally required to handle events.*

Behavior of the reconciliation keys

Values for event fields

A value for required event fields is provided	A value for required event fields is not provided	A value is provided for required event fields but it does not match the value in the database
The record does not exist: The event (in Insert/Update or Update only mode) is fired.	The record does not exist: The event (in Insert/Update or Update only mode) is fired.	The record does not exist: The event (in Insert/Update or Update only mode) is fired. Records with the keys are not updated.
The record exists: The event (in Insert/Update or Update only mode) is fired.	The record exists: The event (in Insert/Update or Update only mode) is fired.	The record exists: The event (in Insert/Update or Update only mode) is fired. Records with the keys are not updated.
The record exists and its value has not changed: No event is fired.	The record exists and its value has not changed: No event is fired.	The record exists and its value has not changed: No event is fired. Records with the keys are not updated.

Behavior of the reconciliation scripts

The behavior of events is defined in ServiceCenter. Generally, an event can update and insert records but cannot delete records. A specific event exists to delete records. The manner in which reconciliation is carried out depends on the table that is selected. If the reconciliation entails record deletion, the corresponding event defined in ServiceCenter will be used.

Additional Information for the ServiceCenter and Service Manager Connector

This section presents additional information about the connector.

The scdb.cfg file

The ServiceCenter database descriptions are contained in an `scdb.cfg` file. In this section, when you are asked to modify this file, you must modify the file corresponding to the ServiceCenter database used:

- `scdb6.cfg`: For a ServiceCenter 6 database
- `scdb7.cfg`: For a Service Manager 7 database
- `scdb9.cfg`: For a Service Manager 9 database

Deleting events processed successfully by the ServiceCenter connector

Deleting events processed successfully by the ServiceCenter connector enables you to purge the ServiceCenter queues. Example: the ServiceCenter connector processes output events transmitted to the E-mail connector (sending). To delete these events, you must use the processing reports produced by each connector. For more information about processing reports, refer to User's guide, chapter Processing reports.

Inserting attachments in the ServiceCenter database

For a document consumed by the ServiceCenter connector to be able to insert attachments in a target database, you must modify the document type on which it is based (document type corresponding to an input event). To modify this document type, you must edit the `scdb.cfg` file located in the following sub-folder: [Connect-It installation folder]/config/sc.

For a document type to support the processing of attachments, you must:

- Add it to a collection relating to the attachments.
- Initialize the parameters linking these attachments stored in the SYSBLOB file to the target record.

Two cases arise:

- The attachment is associated with an event. In this case, the ServiceCenter event manager associates the attachment to the target record.
- The attachment is inserted directly in the SYSBLOB file.

Modifying the `scdb.cfg` file

In the following sections, the extract of code shows you how the `pmo` document type (help ticket creation) is modified so that it supports attachment processing. Example of an attachment: a screen capture attached to a helpdesk ticket.

1. Step 1: Declare the document type (structure) corresponding to the event.

The following code corresponds to the declaration of the `pmo` document type:

```
{ STRUCT pmo
NODETYPE = EVENT
[...]
}
```

To indicate that the document type does not behave like the other document types, you must declare it as an exception in the `AllTables` structure. Like this:

```
{ STRUCT AllTables
Exception = $(LINK_TABLES), pmo
{ ATTRIBUTE AllFields
}
}
```

Another solution is to create a variable containing the list of events for which the attachments are processed. Then, you have to reference this variable in the declaration of exceptions:

```
#define EVENT_ATTACHMENT pmo
{ STRUCT AllTables
Exception = $(LINK_TABLES), $(EVENT_ATTACHMENT)
{ ATTRIBUTE AllFields
}
```

```
}

```

2. Step 2: Add the corresponding collection to the attachments:

Adding an **attachments** collection in the document-type declaration enables it to process the attachments. This collection must have the following elements:

- A **name** field corresponding to the name of the attachment. (Example: sc.ini). This mandatory field must not contain the file path.
- A **blob** field corresponding to the binary contents of the attachment.

```
{ STRUCT pmo
  NODETYPE = EVENT
  { ARRAY attachments
    CIRCULAR = ATTACHMENT_TEMPLATE_EVENT
    [...]
  }
}
```

The **CIRCULAR** property enables you to change a template that will add the name and blob fields to the attachments collection.

```
{ STRUCT ATTACHMENT_TEMPLATE_EVENT
  MODEOUT = 0
  MODEIN = 0
  { STRING name MANDATORY = 1
  }
  { BLOB attachment MANDATORY = 1
  }
}
```

3. Step 3: Configure the attachments:

```
{ STRUCT pmo
  { ARRAY attachments
  [...]
  // Not displayed for the output event pmo
  MODEOUT = 0
  NODETYPE = BLOB
  BLOBTYPE = 5
  BLOBFORMAT = SC
  APP = problem
  INSERTBLOB = 1
  [...]
}
```

MODEOUT

The value 0 enables you to indicate that the attachments can only be processed in consumption mode the value 1 enables you to indicate that the attachments can also be processed in production mode. This is notably the case for document types available in both production and consumption.

NODETYPE

Enables you to indicate that the **Attachments** collection manages binary-type fields (BLOB-type fields for Connect-It). This element is mandatory and must have **BLOB** for its value.

BLOBTYPE

This parameter enables ServiceCenter to classify these BLOB-type fields. The value 5 corresponds to an attachment, the only BLOB-type field supported by ServiceCenter in insertion.

BLOBFORMAT

This parameter enables ServiceCenter to specify a BLOB-type file storage format. For the attachments, this parameter must be **SC** for its value.

INSERTBLOB

This parameter indicates whether there is an insertion or an insertion/replacement of attachments in the target record. The value by default, 1, corresponds to the insertion only.

APP

Indicates the ServiceCenter file in which the attachment is inserted. The name of the file is contained in the **application** field of the **SYSBLOB** file. If your version of ServiceCenter enables you to associate attachments with an event, the event manager automatically makes the association between the SYSBLOB and the target record. To find out if ServiceCenter supports this, an attachments tab appears when you consult the event queue (eventqueue) in the manager. In this case, the reconciliation between the target record and the SYSBLOB file is based on the number field of the target record.

The following extract of code shows the modification of the pmo document type so that it supports the processing of attachments.

```
{ STRUCT pmo
NODETYPE = EVENT
{ ARRAY attachments
[...]
// Insert the attachments into the event
BLOBRECONCTYPE = EVENT
APP = eventin }
{ ATTRIBUTE AllFields
}
}
```

BLOBRECONCTYPE

The **EVENT** value indicates that the attachment is associated with the event.

APP

The **eventin** value indicates in the SYSBLOB file the table in which the attachment is inserted. The event manager then replaces the eventin name by the name of the table of the record associated with the event. Example: the problem table during the creation of a **pmo** event.

***Note:** As soon as the association between attachments and event is possible, you must use this type of processing. The configuration of pmo pmu, pmc and smin events in the scdb.cfg file enables you to use this type of processing.*

When ServiceCenter cannot associate attachments to an event, you must specify how the value of the **topic** field is obtained. This SYSBLOB file field defines the join with the target record. The following contexts are possible:

- a. The value of the **topic** field is contained in the document consumed by the ServiceCenter connector. For example, in the consumed document-type corresponding to an **ICMpc** event (event concerning the computers), the logical.name field enables you to create the link between the **SYSBLOB** field and the target record of the **devicepc** in the ServiceCenter database. The value of this field is directly extracted from the fields mapped in the event, using the following syntax:

```
{ STRUCTURE ICMpc
[...]
}
```



```

{ STRING logical.name MANDATORY = 1
}
{ ARRAY attachments
[...]
BLOBRECONCTYPE = INTERNAL
PIFLINK = @{'..'logical.name'}
[...]
}
{ ATTRIBUTE AllFields
EXCEPTION = 'logical.name'
}
}

```

BLOBRECONCTYPE

The **INTERNAL** value of this parameter indicates that the value of the **topic** field is contained in the fields of the document consumed by the connector.

PIFLINK

The `@{..field_name}` value enables you to recover the value of a field in the document consumed by the connector. The path of the field is relative to the **attachments** collection in the example. The ellipsis (..) enable you to indicate that you must go up a level to find the `logical.name` field. Like in the file path, each ellipsis (..) corresponds to a level above the current one. Thus, `@{...field_name}` enables you to go up two levels.

Note: *If a field contains one point (.), its name must be surrounded by single quotation marks (.'). Example: PIFLINK = @{'..'logical.name'}*

In the example, the **logical.name** is made mandatory because its value is necessary to insert the attachment.

- b. The value of the **topic** field is obtained using a request sent by the ServiceCenter database. In this case, a field different from the **topic** field is used to perform the reconciliation of records. For example, the **contact.name** field is necessary to populate the **topic** field, but only the **first.name** and **last.name** fields are present in the consumed document.

```

{ STRUCTURE eventcontacts
[...]
{ STRING last.name
MANDATORY = 1
}
{ ARRAY attachments
[...]
BLOBRECONCTYPE = QUERY
TOPICFIELD = contact.name
PIFLINK = last.name = @{'..'last.name'}
[...]
}
{ ATTRIBUTE AllFields
EXCEPTION = last.name
}
}

```

BLOBRECONCTYPE

The **QUERY** value indicates that the value of the **topic** field is obtained using a query performed on the ServiceCenter database

TOPICFIELD

In the example, the value **contact.name** indicates that this field is used to populate the value of the **topic** field.

PIFLINK

In this example, the value **@{..'last.name'}** indicates the value used in the WHERE clause, sent to the ServiceCenter database.

Note: *This processing mode requires the event be treated in a synchronous way.*

- c. The value of the **topic** field can be obtained after the document is processed by the ServiceCenter event manager.

Important:

This processing mode does not apply when your version of ServiceCenter does not support the association between event and attachments.

This value is obtained when the event's **evid** field is read. The title of this field is **Problem ID** or **Incident ID**, depending on the version of ServiceCenter you are using. This processing mode corresponds to the creation of a work order (**event pmo**) or a helpdesk call (**event smin**).

```
{ STRUCT pmo
[... ]
{ ARRAY attachments
[... ]
BLOBRECONCTYPE = EVID
[... ]
}
[... ]
}
```

BLOBRECONCTYPE

The value EVID indicates the value of the **topic** field is obtained using the **evid** field of the event. The **PIFLINK** parameter is not necessary. **Note:** This processing mode requires the event be treated in a synchronous way.

Using Service Manager 9.2

If you are using Service Manager 9.2 with a customized extension file, you must add `SM_FIELD_PATH` entries in tables **cm3r** and **cm3t**. See the default extension file (`{install directory}\config\scdb9.cfg`) for field path information.

Warning for events in asynchronous mode

When events configured in asynchronous mode are used to insert data in ServiceCenter, Connect-It cannot verify if these events were correctly processed. In this case, a warning message appears:

- In the Connect-It log when the session is opened.
- In the Document log for each document processed.

In order that these messages no longer appear:

1. Select the **Edit > Options** menu.
2. Unfold the **ServiceCenter** node in the Connector section.
3. Assign the value **No** to the **Display a warning for events used in asynchronous mode** option.
4. Click **OK**.

Synchronous processing vs. asynchronous processing - ServiceCenter connector

Each document consumed by a ServiceCenter connector corresponds to a query sent to a destination application. To improve the time it takes to consume documents, you can choose from:

- **Synchronous data processing**

Each consumed document is sent once the previous document has been processed by the destination application.

- **Asynchronous data processing**

Each consumed document is sent even if the previous document has not been consumed by the destination application.

Interpretation of text fields

In a ServiceCenter database, long text fields are held in the form of an array. Connect-It interprets these arrays and brings together the lines of text in a single long text field. If you wish to interpret these arrays as collections, for example a list of values from the local.software computers table, you must modify the `sc.cfg` file located in the `Connect-It\config\sc` application folder.

1. Edit the `sc.cfg` file.
2. Search for the **NotMemo** section.
The first line of each section contains the name of the table for which the processing exception is to be applied. This table name is prefixed with the asterisk character (*) in order to take into account other tables containing the same field. For example:

```
{ NotMemo
{ *deviceworkstation
'boot.files'
}
{ *pc*
'controlling.software'
'local.software'
'remote.software'
}
}
```

Here, the **PC** table contains three fields that will be processed as collections.

3. Add the sections and fields to not be interpreted and then save.

Note: *Syntax used:*

Field and table names are case sensitive. If the field or the table name contains a period, the whole name must be included in single quotes '(...)'.

Interpretation of date and time fields

In a ServiceCenter database, date and time fields are held in the form of an array. Connect-It interprets these arrays and presents the date and time fields in the form of a collection. Date and

time fields are problematic as Connect-It cannot differentiate between fields containing a date and time fields containing a duration.

To interpret a date and time field as a duration, you must modify the scdbN.cfg file (where N is the version number of ServiceCenter) as shown in the following example:

1. Make sure that the table or event is correctly declared.

```
// List of tables/events having a processing to add a link or
process blob or have duration field
#define LINK_TABLES ocmo, eventregister, contacts, problem,
enclapplication, application, probsummary, incidents, ocml, rmlin,
clocks
#define LINK_DEVICES device, deviceparent, computer, joincomputer,
displaydevice, joindisplaydevice, furnishings, joinfurnishings,
handhelds, joinhandhelds, mainframe, joinmainframe,
networkcomponents, joinnetworkcomponents, officeelectronics,
joinofficeelectronics, softwarelicense, joinsoftwarelicense,
storage, joinstorage, telecom, jointelecom
#define EVENT_ATTACHMENT ICMcomputer, ICMapplication,
ICMdisplaydevice , ICMfurnishings, ICMhandhelds, ICMmainframe,
ICMnetworkcomponents, ICMofficeelectronics, ICMsoftwarelicense,
ICMstorage, ICMtelecom, pmo, pmu, pmc, smin
```

2. If the table or the event is not declared as an exception, add the following script:

```
{ STRUCT incidents
MODEIN = 0 // Do not display the table in consumption. Put this
line only for a table
MODEOUT = 0 // Do not display an event in production. Put this line
only for an event and if this event is not available in production.

{ ATTRIBUTE AllFields
}
}
```

3. Declare the duration field as a long integer with the DURATION property:

```
{ STRUCT incidents
MODEIN = 0 // Do not display the table in consumption. Put this
line only for a table
MODEOUT = 0 // Do not display an event in production. Put this line
only for an event and if this event is not available in production.

{ LONG handle.time
DURATION = 1 }
{ ATTRIBUTE AllFields
EXCEPTION = handle.time
}
}
```

4. If the ServiceCenter connector is open, close it and then open it again. The **handle.time** field now appears as a 32-bit long integer and no longer as a date and time field.

5. If the fields has already been mapped, you must:
 - In production mode: Edit the document type, delete the field before adding it again.
 - In consumption mode: Edit the mapping, delete the field from the mapping before adding it again.

HP ServiceCenter and Service Manager Web Service Connector

Connector type: **CONSUMPTION and PRODUCTION**

The ServiceCenter and Service Manager Web Service (SC/SM) connector lets you interact with a SC/SM Web service. In an integration scenario, the SC/SM Web Service connector consumes a document and sends it in the form of a query to a Web service. Then it receives a reply which it automatically transforms into a produced document.

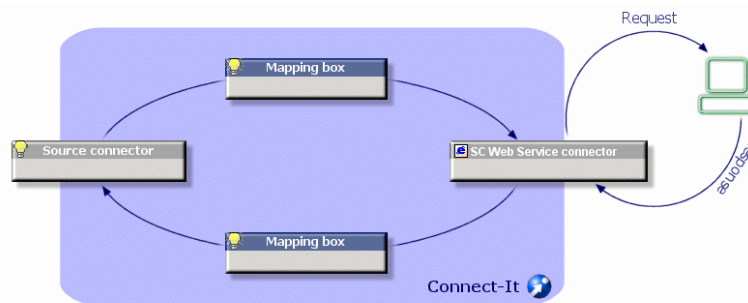


Figure 5-5: Web service connector request and response

Out-of-Box Scenarios

See [HP ServiceCenter and Service Manager Web Service Scenarios](#).

If you have any questions about HP Service Manager when using the out-of-box scenarios, see the Service Manager Event Services Guide and Web Services Guide.

To download the documents:

1. Log on to the support website <http://support.openview.hp.com/selfsolve/manuals>
2. Select **Service Manager** in the Product list.
3. Select the version number in the Product version list, for example, **9.30**.
4. Select the operating system where the Service Manager product runs.
5. Provide key words, for example, **Services Guide**, and then click **Search**. The search results display at the bottom of the page.
6. Select your target document and download it.

Note: The search results may differ depending on your search criteria.

Configuring the HP ServiceCenter and Service Manager Web Service Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Define the connection parameters

Use this page to indicate the connector's connection parameters to the ServiceCenter and Service Manager (SC/SM) Web Service server.

- **Server name:** Name of the SC/SM server in **computer:port** format.
- **Context path:** For example: `sc62server/PWS`.
- **Service name:** Name of the SC/SM ITIL service, incident management, configuration management, problem management, change management, service desk and service level management.
- **Test:** This button enables you to test your connection. To test your connection, enter your connection parameters and click Test. The Test the connection window appears and tells you whether the connection succeeded or failed. If it failed, there will be messages explaining the cause.
- **Login:** SC/SM login.
- **Password:** The password for this login.

Options for Service Manager

Use this page to configure some HTTP options. Note that the following options are for Service Manager only.

- Choose the protocol you want to use: You can choose SOAP 1.1 or SOAP 1.2.
- HTTP options for performance tuning: Select the following options as needed.
 - Compress the SOAP requests (gzip) - If the option is selected, the connector sends the compressed HTTP request that contains an HTTP header `Content-Encoding:gzip`.
 - Accept compressed SOAP responses - If the option is selected, the connector sends the HTTP request that contains a HTTP header `Accept-Encoding:gzip`, which tells the server that the connector allows a compressed response.
- Close the HTTP connection after each SOAP request - Select the option if you want to disable the HTTP persistent connection and open a new connection after a SOAP request/response pair.
- Socket Timeout (in seconds): Specify the socket timeout for HTTP requests in seconds. The default value is **60**. The value of **0** means to turn off timeout functionality (never timeout).

Configure the JVM

The HP ServiceCenter and Service Manager Web Service Connector provides a JVM option **cit.smws.version** that applies to Connect-It 9.50 only. Set the value to V1 to convert back to the old implementation Version 9.40 when you encounter any regression error in Version 9.50.

Note about consumption directives

For each document that is consumed by the connector, an option is available when the document root is selected: **Ignore response failure status**. The ServiceCenter Web service operates in the following manner: Any query sent to the ServiceCenter Web service produces a response. This response contains the status, which indicates either success or failure.

- When the Ignore response failure status option is enabled, the response is always produced.
- When this option is not enabled, the response is not produced if an error is returned by the Web service, and an error is saved for the consumed document type.

When documents are produced, you can choose to produce elements with null values via the **Produce null elements** command.

Note: *Connect-It specified JVM options are applicable to this connector. See the [Configuring the JVM](#) topic for more information.*

HP Service Anywhere Connector

Connector type: **CONSUMPTION and PRODUCTION**

The Service Anywhere connector lets you interact with a Service Anywhere web service. In an integration scenario, the Service Anywhere connector consumes a document and sends it in the form of a query to a web service. Then it receives a reply which it automatically transforms into a produced document.

Out-of-Box Scenarios

See [HP Service Anywhere Connector Scenarios](#).

Configuring the HP Service Anywhere Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Define the connection parameters

Use this page to indicate the connector's connection parameters to the Service Anywhere server.

- **Server name:** Name of the Service Anywhere server in **computer:port** format.
- **Context path:** `ServiceAnywhere/ws`.
- **Service name:** Name of the ITIL service, incident management, problem management, change management, service desk, and so on.
- **Test:** This button enables you to test your connection. To test your connection, enter your connection parameters and click Test. The Test the connection window appears and tells you whether the connection succeeded or failed. If it failed, there will be messages explaining the cause.
- **Login:** The Service Anywhere login.
- **Password:** The password for this login.

Options for Service Anywhere

Use this page to configure some HTTP options.

- Choose the protocol you want to use: You can choose SOAP 1.1 or SOAP 1.2.
- HTTP options for performance tuning: Select the following options as needed.
 - Compress the SOAP requests (gzip) - If the option is selected, the connector sends the compressed HTTP request that contains an HTTP header `Content-Encoding:gzip`.
 - Accept compressed SOAP responses - If the option is selected, the connector sends the HTTP request that contains a HTTP header `Accept-Encoding:gzip`, which tells the server that the connector allows a compressed response.
- Close the HTTP connection after each SOAP request - Select the option if you want to disable the HTTP persistent connection and open a new connection after a SOAP request/response pair.
- Socket Timeout (in seconds): Specify the socket timeout for HTTP requests in seconds. The default value is **60**. The value of **0** means to turn off timeout functionality (never timeout).

Note about consumption directives

For each document that is consumed by the connector, an option is available when the document root is selected: **Ignore response failure status**. The Service Anywhere operates in the following manner: Any query sent to the Service Anywhere produces a response. This response contains the status, which indicates either success or failure.

- When the Ignore response failure status option is enabled, the response is always produced.
- When this option is not enabled, the response is not produced if an error is returned by the Web service, and an error is saved for the consumed document type.

When documents are produced, you can choose to produce elements with null values via the **Produce null elements** command.

Note: *Connect-It specified JVM options are applicable to this connector. See the [Configuring the JVM](#) topic for more information.*

HP SCAuto Listening Connector

Connector type: **CONSUMPTION and PRODUCTION**

The SCAuto listening connector emulates a ServiceCenter SCAutomate server (SCAuto). It can receive events and process client requests.

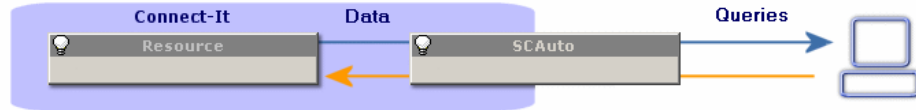
SCAuto listening connector in event-receiving mode

The SCAuto listening connector in event-receiving mode receives events that enable it to produce documents each time that it is started in manual or scheduled mode. This connector uses communication by socket.



Figure 5-6: Event-receiving mode**SCAuto listening connector in client-request-processing mode**

Using the connector in client-request-processing mode depends on the use of a resource connector that processes requests addressed to your Connect-It server by clients.

**Figure 5-7: Client-request-processing mode****Limitations of the SCAuto listening connector**

This connector does not support the following adapters:

- The Japanese version of the SCAuto HP Network Node Manager adapter version 3.x.
- SCAuto Email adapter
- All the adapters using the following commands:
 - QUERY2
 - INSERTBLOB
 - CREATEBLOB
 - DESCRIBE OBJECT
 - SELECT OBJECTS
 - END SELECT OBJECTS
 - GETNEXT OBJECT
 - STORE OBJECT

If the SCAuto listening connector supports the DELETE command of the SCAuto SDK, its use does not allow the deletion of events. To delete events, you must use processing reports.

Out-of-Box Scenarios

See [HP SCAuto Listening Connector Scenarios](#).

Configuring the SCAuto Listening Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Configuring SCAuto server emulation

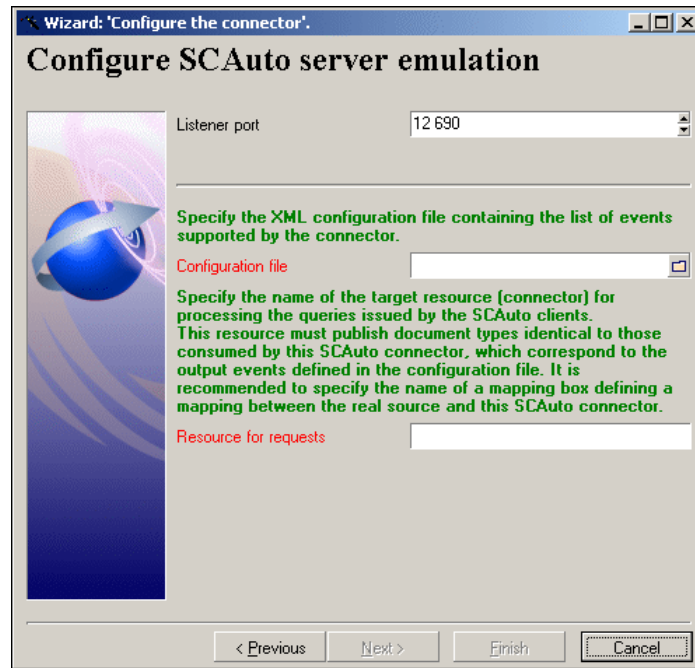


Figure 5-8: SCAuto server emulation configuration

This page enables you to enter the connection parameters to your SCAuto server.


Listening port

Specify the listening port that enables the connector to receive client requests. This field is mandatory.

Configuration file

Indicate the path of the XML configuration file containing the descriptions of the document types that the SCAuto listening connector publishes. This field is mandatory. These document types correspond to the SCAuto event types. An Connect-It scenario enables you to obtain this configuration file from ServiceCenter event types.

To obtain the configuration file:

1. Launch the Scenario Builder.
2. Open the scacfg.scn scenario. This file is located in the following file: [Connect-It installation folder]\scenario\scauto\scacfg.
3. Configure the scenario's ServiceCenter connector.
4. Open the scenario connectors by clicking .
5. Start all the schedulers (Ctrl + F5).
The scenario creates the XML file in the [Connect-It installation folder]\scenario\scauto\scacfg file.

Specify the name of the tool that processes the requests sent to your Connect-It server.

Additional information

Modifying the SCAuto listening-connector configuration file

Modifying the SCAuto listening-connector configuration file enables you to support more (or less) event types. You can also decide to process character-string collections as Memo-type fields. The configuration file contains two sections:

- **InputEventTypes**

Events consumed by the SCAuto listening connector. The consumed events correspond to those requested by the SCAuto adapters.

- **OutputEventTypes**

Events produced by the SCAuto listening connector. The produced events correspond to those produced by the SCAuto adapters.

An event type is represented as a structure having the same name as the event type. This structure contains sub-nodes representing the fields of that event. Four types of simple fields are supported:

- String (Character)
- Byte (Logical)
- Long (Number)
- TimeStamp (Date)

The types in parentheses are the names of the equivalent types in ServiceCenter. There are several ways in which you can represent array-type fields: as a collection of simple fields (which is the default behavior of the scacfg.scn scenario); as a simple Memo-type field whose value is a paragraph (inside which each array element value will be represented by a line).

Caution: *Only the character string arrays should be represented in this way. If this is not the case, the array elements will still be considered as character strings.*

A simple field is characterized by an index, a name and a type. For example:

```
<ATTRIBUTE index="11" name="orig.operator" type="String"/>
```

An array field is characterized by an index, a name, a separator, an element name and an element type. The name of the element is arbitrary and can be different from that of the field.

For example:

```
< COLLECTION index="2" name="comments.2" separator="|">  
< ATTRIBUTE name="comments" type="String"/>  
< /COLLECTION>
```

In this example, the array field can also be represented by the following line:

```
< ATTRIBUTE index="2" name="comments.2" type="Memo" separator="|" />
```

In this last representation, a three-element array ("abc", "def" and "ghi") will give the attribute value:

```
"abc  
def  
ghi"
```

If the **separator** attribute is not there, or if it is empty, the default separator '|' will be used. There are two particular separators: **lf** and **cr**. They are interpreted as end of line and carriage return, respectively. With the exception of these two particular separators, a separator must be composed of one single character other than ^.

HP Universal CMDB Connector 8

This connector is used to exchange data with HP UCMDB 8.x via the Java API of UCMDB. For version 9.x and later, see [HP Universal CMDB Connector](#).

Out-of-Box Scenarios

See [HP Business Service Management Dashboard / HP Universal CMDB connector scenarios](#).

Configuring the HP Universal CMDB Connector 8

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Universal CMDB business model definition

Specify the parameters that define the business model.

- **Configuration file:** The Topology Query Language (TQL) file which is exported from Query Manager of UCMDB. Represents a user-defined query to retrieve specific data from UCMDB.
- **Only extract data model from the specified TQL configuration file:** Select to not execute a server-side query.

Consumption Directives of the UCMDB Connector

For each document that is consumed by the connector, two options are available when the document root is selected:

- Generate response
- Insert mode

If **Generate response** is enabled, the response is produced when consuming data into uCMDB. Default is not enabled. The **Insert mode** option determines the behavior of UCMDB with respect to CIs and relations that already exist. A CI exists if the values of key attributes are the same as those of an existing CI. A relation exists if there is a relation of the same type between the same CIs. There are three different insert modes in UCMDB:

- **FAIL_IF_EXIST:** Fail the creation of the whole batch if any of the supplied CIs or relations already exist.
- **IGNORE_EXISTING:** Create non-existing CIs and relations and ignore existing ones.
- **UPDATE_EXISTING:** Create non-existing CIs and relations and update the non-key properties of existing ones.

The default option is **Update existing**.

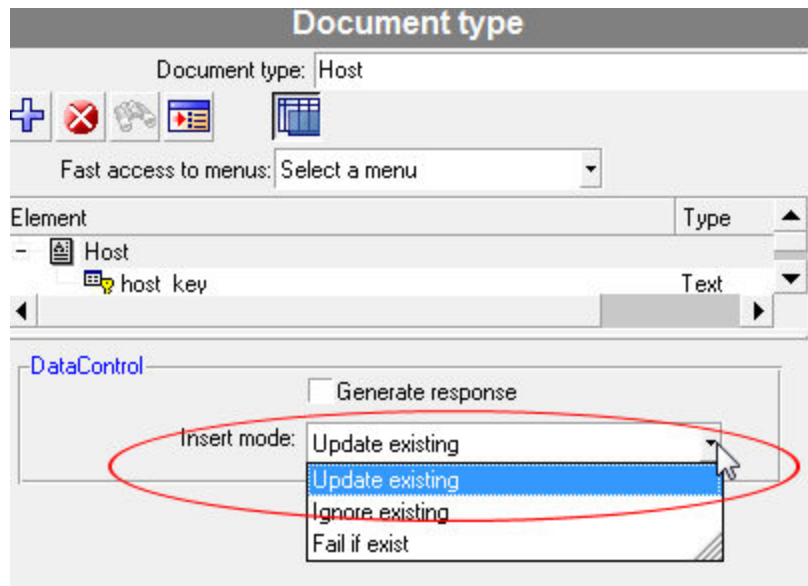


Figure 5-9: Insert mode: Update existing

Server-side or client-side query

By default, the UCMDB connector queries data from the server side, which means that data is retrieved by executing a query defined at the client side over the result of another query stored at the server. The query stored at the server (the parent query) is identified by the TQL file specified when configuring the connector. The server-side query is more efficient because it does not look for objects in the entire UCMDB, but is restricted to the topology returned by the parent query. However, if the query related to the specified TQL file is removed or renamed at the uCMDB server, then the parent query no longer exists and the connector query will fail. If the content of the query changed at the server, the connector query will keep the changes.

Data can be queried from the client side (data is retrieved by executing a query defined at the client side). For such a query, there is no relationship between the client-side query and the query stored at the server. Therefore, even if the query at the uCMDB server is renamed or removed or the contents are changed, the connector query will not be affected. Server-side or client-side queries can be selected when configuring the connector in the following dialog:

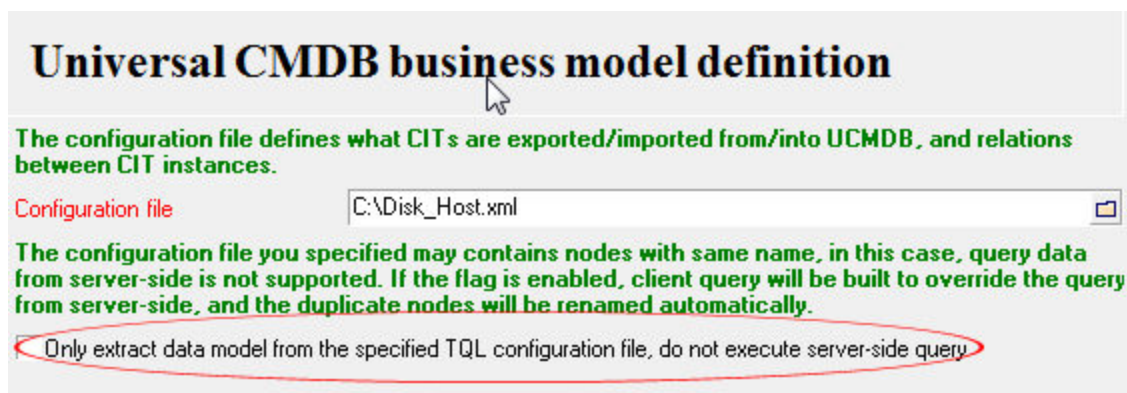


Figure 5-10: UCMDB business model definition

Name conflict in configuration file

A name conflict occurs when the TQL file contains nodes with duplicate names. A query from the server-side is not supported when a name conflict occurs because of the uCMDB Java API restriction. The workaround for this issue is to use the query from the client side. The duplicate node will be renamed automatically when parsing the TQL file.

Bulk consumption

This is a page of the "Configure the connector" wizard. You can choose to activate bulk operation when using the HP Universal CMDB connector to import data to UCMDB. Each document consists of one or more CIs or CI relationships. By default documents are consumed one by one, and it may take a long time for reconciliation particularly when a large amount of data is to be consumed with high network latency. To improve performance, you can select the "Commit after each bulk" option, and then specify the number of CIs and relations in the bulk. In this way, the Universal CMDB connector can consume volumes of data in batch. Note that the maximum number of CIs and relations in a bulk is 20,000. See the wizard page as below:

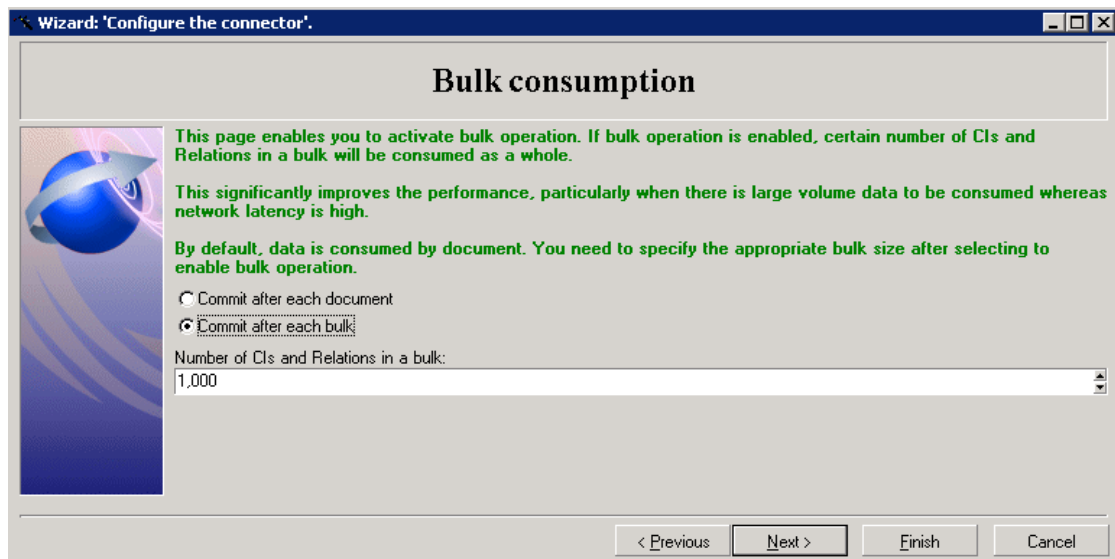


Figure 5-11: Bulk consumption page

HP Universal CMDB Connector

Connector type: **CONSUMPTION and PRODUCTION**

This connector is used to exchange data with HP UCMDB 9.x and 10.0x by using the Java API of UCMDB. Both data push and data population are supported and depend on the integration query defined in the UCMDB server.

Out-of-Box Scenarios

See [HP Business Service Management Dashboard / HP Universal CMDB connector scenarios](#).

Configuring the HP Universal CMDB Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that

section, they are available below.

Define the connection parameters

Specify the parameters for the connection to the UCMDB server.

- **Server:** Name or IP address of the UCMDB server
- **Port:** Port of the UCMDB server
- **User:** User of the UCMDB server
- **Password:** Password of the UCMDB server
- **Customer name:** Name of the customer
- **Query name:** Specify the name of the query you defined in Modeling Studio on the Universal C MDB server. The name is case-sensitive.

Consumption directives

For each document that is consumed by the connector, the following options are available to control data consumption:

- **Generate response**
If Generate response is enabled, the response is produced when consuming data into UCMDB. This is not enabled by default.
- **Operation type**
There are three types of operation support:
 - **Insert**
 - **Update**
 - **Delete**

***Note:** The Update and Delete operations require the **Global Id** attribute for each CI and relation.*

- **Execution mode**
The Execution mode determines the behavior of UCMDB with respect to CIs and relations that already exist. A CI exists if the values of key attributes are the same as those of an existing CI. A relation exists if there is a relation of the same type between the same CIs. Two options are available:
 - **Optimistic**
For the **Insert** operation: creates non-existing CIs and relations, and then updates the non-key properties of existing ones.
For the **Update** operation: updates already existing CIs and relations and ignore the others.
For the **Delete** operation: deletes already existing CIs and relations and ignore the others.
 - **Strict**
For the **Insert** operation: cancels the creation of the whole batch if any of the supplied CIs or relations already exist.
For the **Update** operation: cancels the updating of the whole batch if any of the supplied CIs or relations do not exist.
For the **Delete** operation: cancels the deleting of the whole batch if any of the supplied CIs or relations do not exist.

Bulk Consumption

This is a page of the "Configure the connector" wizard. You can choose to activate bulk operation when using the HP Universal CMDB connector to import data to UCMDB. Each document consists of one or more CIs or CI relationships. By default documents are consumed one by one, and it may take a long time for reconciliation particularly when a large amount of data is to be consumed with high network latency. To improve performance, you can select the "Commit after each bulk" option, and then specify the number of CIs and relations in the bulk. In this way, the Universal CMDB connector can consume volumes of data in batch. Note that the maximum number of CIs and relations in a bulk is 20,000. See the wizard page as below:

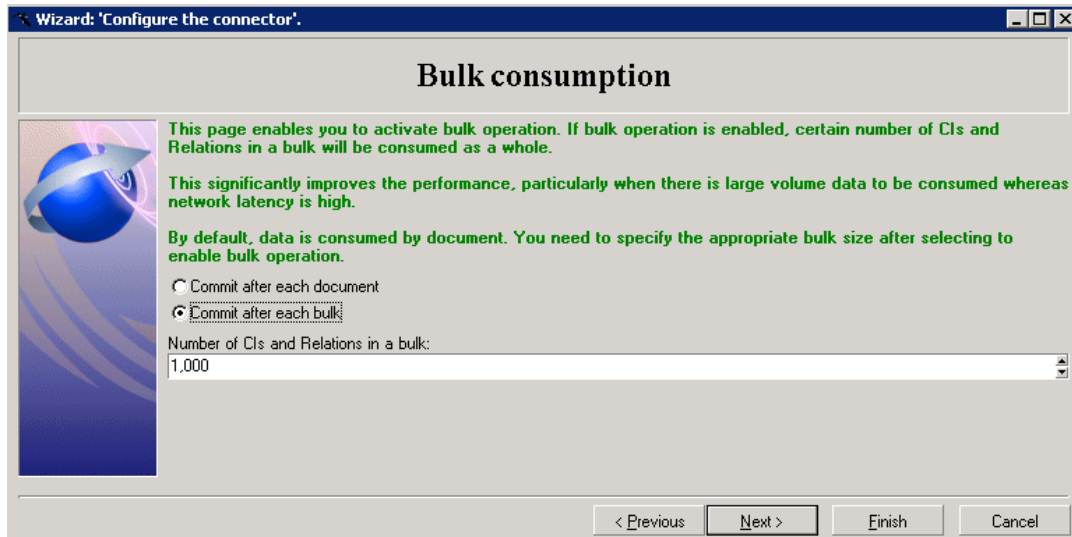


Figure 5-12: Bulk consumption page

Data query performance tuning

- **Chunk size**

In UCMDB, when the number of TQL query results exceeds a threshold, the results will be returned in batches (in the form of certain chunks). The threshold value here is the chunk size.

The chunk size impacts the overall performance of data produce from UCMDB. In general, the memory consumption decreases with a smaller chunk size, but the number of queries will increase and the overall query speed will be reduced. Meanwhile, the number of queries decreases and the overall speed increases with a bigger chunk size, but the memory consumption will increase and may cause the out-of-memory issue.

The default chunk size for the UCMDB connector is 50,000, which can balance the memory usage and query performance. In most cases, you can use this default value. But depends on your actual requirement, you can change it by using the JVM option as format – `Dcit.ucmdb.chunksize=N`. N is usually between 20,000 and 100,000. The UCMDB server configuration will be used when N is set to -1. For UCMDB server settings about the chunk size, refer to either the TQL Result Utils Chunk Max Result Size option in the UCMDB UI or the JMX `tql.resultutils.chunk.maxresultsize` setting.

Note: The timeout setting for chunking data can be configured in `tql.resultutils.chunk.keepingperiod.seconds` via the UCMDB JMX console, and the default value is 5 minutes. If any of the chunks is not requested for 5 minutes, the whole data of query result will be cleared on the UCMDB server and you will get a UCMDB error message

such as “*ErrorCode [200] The TQL query is missing*”. In such cases, you have to either decrease the chunk size or increase the timeout setting on the UCMDDB server.

- **Query based on document or server**

A TQL query is created based on the document type so that the query result will exactly match with the document type. This is more like a sub-query. When the TQL on the UCMDDB server is complex with many query nodes but only part of the nodes are defined in the document type, the amount of returned data will be significantly reduced and the query performance is improved.

You can disable this feature by using the `-Dcit.ucmdb.subquery=0` JVM option. When this feature is disabled, all query results on the UCMDDB server will be returned despite of the defined document type. Some results are irrelevant if the corresponding nodes are not defined in the document type. Although these irrelevant data will be ignored automatically, they still impact the system performance.

Note: *The sub-query feature can accelerate the performance by getting a smaller volume of items from UCMDDB. But If you are NOT using UCMDDB version 10.01, version 9.05 with CUP 7, or later versions/patches, and experiencing some data loss issues, you need to disable this feature.*

HP Universal CMDB Connector (XML)

Connector type: **CONSUMPTION and PRODUCTION**

The Universal CMDB XML connector enables the:

- Processing of XML files located on the computer or network on which Connect-It is installed
- Processing of XML files located on FTP or Web sites.

Note: *The XML files processed can be compressed (gzip).*

Known limitations of the uCMDB (XML) connector

In the document types processed by the XML connector, the following field types are not supported:

- **any**
- **PCDATA** (blob)
- **CDATA**
- **Namespaces**, except qualified names when XSD is selected

UNIX environment

Under UNIX, the DTD specified in the configuration wizard of the uCMDB (XML) connector must reference a local file, even if it is possible to specify the file on an FTP or HTTP server.

FTP and HTTP protocols

For the connector to function correctly using the HTTP and FTP protocols, the dlls `wininet.dll` and `shlwapi.dll` must be installed on the Connect-It client machine.

Configuring the HP Universal CMDB Connector (XML)- Read

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Path of a file or folder

Often when configuring a connector, you must indicate the path of a file or folder in the configuration wizard. If this file or folder is in a folder associated with a network drive on your computer, you must never indicate the letter of this drive; only indicate the folder associated with this letter.

Using a network drive in a path will prevent the scenario from working properly when it is associated with a service. For more information, refer to the User's guide, chapter Document type mappings, section Edit the mappings, sub-section Path of a document's elements.

Choosing a processing mode

This page enables you to indicate if the connector is used in read or write mode.

Selecting a connection protocol

The three available options are:

- HTTP Web site
- FTP server
- Local or network file(s)

Connecting to an HTTP Website

If you have chosen to read the XML files on a Web site, you must indicate the HTTP connection parameters:

- **Address**
Enter an address of the type: `[protocol]://[address]:[port]/[path]`. The `[address]` part is sometimes the only mandatory part. The usual port number for an HTTP server is **80**.
- **Login**
Enter the login that enables you to access the chosen site.
- **Password**
Enter the password corresponding to your login.

Secured connection (HTTPS protocol)

This option enables you to indicate if you want to connect to your site via a secured connection (HTTPS).

Important: *If the `[protocol]` part of the Web site's address does not correspond to the HTTPS protocol, the selection of the Secured connection (HTTPS protocol) option forces the use of an HTTPS protocol. The usual port number for an HTTPS server is **443**.*

Client certificate

This field enables you to select an HTTPS certificate from among those present on your computer. Important: If you change or delete a client certificate specified in this field after the connector has already read or written to these documents, you must close and reopen the Scenario Builder in order for this modification to be accounted for.

Manage the list of client certificates

The list of client certificates proposed by the configuration wizard corresponds to the list of certificates in your **Root console > Certificates - current user > Personal > Certificates** folder of the Microsoft Management application.

To add certificates to this folder using Windows XP:

1. Select **Run** from the Windows Start menu.
2. Enter **MMX** in the **Open** field.
3. Select **File > Add/Remove Snap-in**.
4. Click **Add** in the window that is displayed.
5. Select **Certificates** in the window that is displayed.
6. Click **Add**.
7. Select My user account in the dialog box that is displayed.
8. Click **Finish**.
9. Click **OK**.
10. Add or delete the files in the **Certificates - Current user > Personal > Certificates** folder.

Do not verify the server identity

Use this option to support a proxy server in a secure connection. If a proxy server is installed, the address specified in the server certificate does not match the address specified for the proxy server. Select this option to disable the automatic verification of the server identity.

Connect via a proxy

This option is available for HTTP and FTP connections. Use this option to access a proxy server for a given connection. When you select this option, enter the name (or IP address) of the proxy server and, if necessary, select the Use authentication option.

Note: *Cascading proxies are supported if the same authentication is used for all proxies.*

Do not use a proxy server

For HTTP type connections, you can specify conditions when the proxy server should be bypassed (**Do not use a proxy server for** field). You must enter an HTTP type address in this field depending on the connection that is used. Names must be separated by a space.

FTP Protocol

Note: *Connection in passive FTP mode is supported.*

If you have chosen to read the XML files on an FTP site, you must:

1. Select the FTP connection parameters
2. Select an action to perform after the files have been processed.

FTP server connection

On this page, you must populate three fields, which enable you to connect to your FTP server:

- **Server:** your FTP server.
- **Login:** the login that enables you to access the chosen site.
- **Port:** indicate the port to use to access the FTP server.
- **Password:** the password for the login.
- Select the option **Connect in passive mode** if your connection is blocked by a firewall and you wish to open an extra port.
- **Connect via a proxy**
This option is available for HTTP and FTP connections. Use this option to access a proxy server for a given connection. When you select this option, enter the name (or IP address) of the proxy server and, if necessary, select the **Use authentication** option. You can specify the address of the proxy server (**Server address** field) and the list of exclusions (List field).
Note: Cascading proxies are supported if the same authentication is used for all proxies.
- **Use authentication:** If this option is enabled, enter the user name and password.
- **Do not use a proxy server:** For HTTP and FTP type connections, you can specify conditions when the proxy server should be bypassed (**List** field). You must enter an HTTP or FTP type address in this field depending on the connection that is used. Names must be separated by a space.

You can test the validity of your connection using the **Test** button.

Defining Post-Processing Actions

After a file is read by the uCMDB (XML) connector, Connect-It gives you three options:

- Leave it in the folder
- Delete it from the folder
- Move it to another folder
If you choose this last option, you must indicate the path of the folder to which you want to move the processed file.

You must specify one of these options in the case of failure, and one in the case of success, in processing the XML files by your connector.

Note: In scheduled mode, when you select the "On successful processing of a file: Leave in the folder" option, all the files are processed, with no regards to their modification date (there is not schedule pointer).

In order to apply a post-processing action to the documents consumed by the other connectors and mapping boxes of your scenario, you must use the processing reports that each one produces. For more information about processing reports, refer to the User's Guide.

Local or Network Files

If you choose to read local or network files, you must:

1. Choose a file or folder.
2. Select an action to perform after the files have been processed.

Define post-processing actions

After a file is read by the XML connector, Connect-It gives you three options:

- Leave it in the folder
- Delete it from the folder
- Move it to the folder
If you choose this last option, you must indicate the path of the folder to which you want to move the processed file.

You must specify one of these options in the case of failure, and one in the case of success, in processing the XML files by your connector.

Using post-processing actions

To use post-processing actions, you must create a mapping between the **URLFileInfo.Path** structure of the document type produced by the source connector and the **URLFileInfo.Path** element of the **SuccessReport** document type.

Options Associated with the uCMDB (XML) Connector

You can access the options of the XML connector using the menu **Edit > Options > Connector > Delimited text and XML**. The following options are associated with the uCMDB (XML) connector in FTP protocol:

- Display the URL being processed in the Connect-It log
- Copy locally the files to read from FTP server. Select this option to make a local copy of the files on an FTP server and to read the data from the local file.

This option must be selected if the network configuration does not enable a connection to be maintained on the FTP server long enough for a file to be processed.

Configuring the HP Universal CMDB Connector (XML)- Write

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

You can also see sections under [Configuring the HP Universal CMDB Connector \(XML\)- Read](#) for additional instructions.

Select a processing mode

This page enables you to indicate if the connector is used in read or write mode.

Define the connection parameters of the HTTP server

On this page, you must populate three fields enabling you to connect to the HTTP server on which you want to write files.

- **Address**

This field is mandatory. Enter an address of the type:

[protocol]://[address]:[port]/[path].

- **Login**

Enter the login that enables you to access the chosen site.

- **Password**

Enter the password corresponding to your login.

Secured connection (HTTPS protocol)

This option enables you to indicate if you want to connect to your site via a secured connection (HTTPS).

Write command

There are two write commands used to write to an HTTP server. Select the command used by your HTTP server:

- **POST** sends data to the program located at the indicated address. It is different from the PUT method in that the sent data must be processed. Example: sending data from a CGI form.
- **PUT** sends data so that it can be saved to the indicated address. Example: updating pages of a Web site. Select the command used by your HTTP server.

Local network files

If you have chosen to write local or network files, you must select a file or folder.

Select files or folders

This page enables you to select the XML file or files that your connector must write. Before defining the connector's behavior, specify the folder for writing files (**Folder name** field). The path for this folder is used in a mapping in combination with the `UrlFileInfo.Path` element.

You can choose between the following behaviors:

- Write documents to one single file
Enter the path of a folder on your computer or your network (**Folder name** field). The document will be overwritten or left alone depending on the parameters defined in the next page of the wizard.
- Write to a different file for each document
Specify the folder in which the files will be written (**Folder name** field). The name of these files will correspond to the generic name that you have entered in the consumption directives of the XML connector. You can also define the names of these files by mapping the element **UrlFileInfo.Path**. For example, to create a file with the name of each employee in Asset Manager. In this case, the path defined in the wizard page is concatenated with the path defined in the mapping.

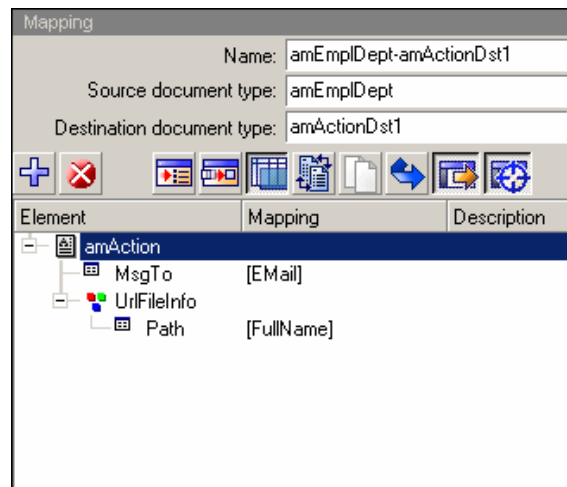


Figure 5-13: Default path in the mapping

In order for the path defined in the mapping to be used as the default path, and as a result to avoid using the path defined in the wizard, use the '/' character in the mapping.

Note: *this behavior does not prevent you from specifying a valid path in the connector configuration wizard.*

If the path specified for the `UrlFileInfo.Path` element does not exist, it is created automatically provided that:

- the path uses part of the path specified in the configuration wizard.
- the path does not add more than one level to the specified tree.
For example, for a path `a/b/c` specified in the configuration wizard, only a path `a/b/c/d/filename`, specified in the mapping, is possible.

FTP protocol

See [FTP protocol](#) for the UCMDB Read connector.

Selecting Files or Folders

Before defining a connector's behavior, specify the write files folder (**Path** field). The specified path is used in a mapping in combination with the `UrlFileInfo.Path` element. Select the option **Write the documents to a single folder** or **Write to a different file for each document** according to your needs.

Behavior Between two Sessions

This page enables you to select how documents are processed between two data-writing sessions. The behavioral options are different if you write data to one single file or if you write it to several files in a folder.

Write the documents to a single file

To write to a single file, you have the following options:

- Append to the same file
If you choose this option, the connector starts writing data to the open file again the moment the previous session is interrupted.

- Overwrite the last file
The connector deletes the file in which it wrote data during the previous session.
- Number the different files
The connector creates a file whose name is the same as the previous file, but incremented by 1 unit. Example: `file.xml`, `file1.xml`, `file2.xml`, etc.

Write to a different file for each document

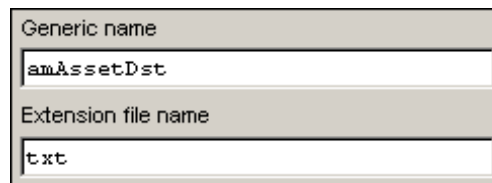
To write in several files, you have the following options:

- Delete all previous files and start numbering them from the beginning
The files in the folder are all deleted (all the files dating from all the previous session; not just from the last one). The connector starts incrementing all the files it writes.
- Continue numbering files
The connector starts writing files again and incrementing them without deleting the existing files.

Consumption Directive of the uCMDB (XML) Connector (write) - FTP Mode

The consumption directives indicate for each document type consumed by the XML connector:

- A generic name
By default, the value of this field is the name of the document type consumed by the XML connector.
- A file name extension
Enter this name without the period mark. Example: Enter `xml` instead of `.xml`; "`xml`" is the default value.



The image shows a configuration window with two input fields. The first field is labeled 'Generic name' and contains the text 'amAssetDst'. The second field is labeled 'Extension file name' and contains the text 'txt'.

Figure 5-14: Generic name and extension file name

These two fields will only be used if you have chosen the **Write to a different file for each document** option during the configuration of your XML connector. When you run your scenario, the files that are written will have a composite name, which is created from the generic name, a number corresponding to the file's creation order (`_01`, `_02`, `_03`, etc.) and from the file-name extension indicated.

For example: You enter the values `ebizz` and `xml`. The files created have the following names: `ebizz_01.xml`, `ebizz_02.xml`, `ebizz_03.xml`, etc.

Caution: *If your XML connector consumes several document types, do not use the same generic name twice: the most recently created files will overwrite the previously created files having the same generic name.*

Additional Information - uCMDB (XML) Connector

This section presents additional information about the connector.

Writing to one or more files

The uCMDB (XML) connector in write mode enables you to write data to one or more files.

Write the documents to a single file

This mode enables you to save each document consumed by the connector in one single file. You can, for example, use this mode in a single file so that a single XML file contains an inventory of assets recorded in Asset Manager. You can use the following DTD:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!ELEMENT AssetInventory (amAsset*)>
<!ELEMENT amAsset (AssetTag?)>
<!ELEMENT AssetTag (#PCDATA)>
```

In this DTD, the root element (AssetInventory) has the amAsset collection as its sub-element. This collection's members correspond to asset records in the Asset Manager database. Each asset is identified by its asset tag. In this case, the uCMDB (XML) connector writes a file. The following example shows the file's contents:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<AssetInventory>
  <amAsset>
    <AssetTag>DEMO-NTR1</AssetTag>
  </amAsset>
  <amAsset>
    <AssetTag>DEMO-OFF1</AssetTag>
  </amAsset>
  <amAsset>
    <AssetTag>DEMO-SFT2</AssetTag>
  </amAsset>
</AssetInventory>
```



Figure 5-15: Writing documents to a file: source and destination mapping

The mapping between the **amAsset** and the DTD used by the uCMDB (XML) connector presents the following aspects:

1. The asset tag field is directly mapped to the AssetTag of the DTD.
2. Only the child nodes (amAsset) of the root element (AssetInventory) are displayed in the destination document type.

With the following DTD:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!ELEMENT AssetInventory (amAsset*, amSoftware)>
<!ELEMENT amAsset (AssetTag?)>
<!ELEMENT AssetTag (#PCDATA)>
<!ELEMENT amSoftware (Version?)>
<!ELEMENT Version (#PCDATA)>
<!ATTLIST AssetInventory ID CDATA #REQUIRED>
```

The amSoftware element and the ID field will be ignored and will not be displayed in the document type consumed by the uCMDB (XML) connector. It is as though they are not declared in the DTD.

3. The **Publish a document type for each first-level collection** option must not be selected.

Write to a different file for each document

This mode enables you to save each document consumed by the uCMDB (XML) connector in different folders. In the previous example, each asset record gives rise to the creation of a file by the uCMDB (XML) connector. You can use the following DTD:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!ELEMENT amAsset (AssetTag?)>
<!ELEMENT AssetTag (#PCDATA)>
```

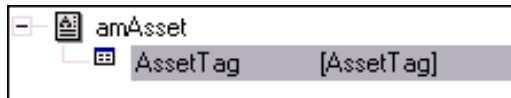


Figure 5-16: Writing documents to a file: source and destination mapping

In this case, the uCMDB (XML) connector writes several files. The following examples show their contents.

File 1:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<amAsset>
<AssetTag>DEMO-OFF1</AssetTag>
</amAsset>
```

File 2:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<amAsset>
<AssetTag>DEMO-NTR1</AssetTag>
</amAsset>
```

Mapping between source and destination

The mapping between the **amAsset** and the DTD used by the uCMDB (XML) connector presents the following aspects:

- All the elements declared in the DTD are displayed in the destination: collections, structures, fields.
- The **Publish a document type for each first-level collection** option must not be selected.

Transformation rules for end-of-line markers

Depending on whether the uCMDB (XML) connector is used in read or write mode, the rules for the end-of-line markers (CRLF or LF) differs.

Utilization in read mode

- In Windows, all end-of-line characters are transformed to CRLF (Carriage Return and Line Feed).
- In Unix, all end-of-line characters are transformed to LF (Line Feed).

This behavior can be modified using a script, as shown in the example below:

```
RetVal = Replace([Record.cc.dd.ee], Chr(13)&Chr(10),Chr(10),0)
```

Utilization in write mode

- In Windows, all end-of-line characters are transformed to CRLF (Carriage Return and Line Feed).
- In Unix, all end-of-line characters are transformed to LF (Line Feed). This behavior may not be modified using a script.

Published Document Types

The uCMDB (XML) connector publishes the **UriFileInfo** structure for all document types. This structure is mainly used for post-processing actions

The **UriFileInfo** structure contains the following fields:

- **creationdate**
This field corresponds to the creation date of the document.
- **lastmodificationdate**
This field corresponds to the last modification date of the document.
- **path**
This field corresponds to the path of the file.

Note: *If you wish to use multiple XML file types, you must deploy as many connectors as there are different files. Since each connector has a unique behavior (production or consumption), you must create as many connectors as there are behaviors.*

Chapter 6

Inventory Connectors

Inventory connectors enable you to process data from applications allowing you to inventory computer-based repositories.

Altiris Connectors

Connector type: **PRODUCTION**

This section describes the Altiris connectors. The Altiris connectors enables you to process an Altiris for Inventory solution database. An Altiris database contains data related to inventoried assets (desktop and portable computers, servers, etc.) and their dependencies (printers).

Altiris 6 Connector

Prerequisites

- In order to allow **LastUse** type data to be processed by Connect-It, a view is created in the Altiris database. This view is required by the Altiris 6 connector. The following script is used to create the view:

```
CREATE VIEW [vCIT_LastUseInformation]
AS
SELECT softinv._id, MAX(softstats.[Last Start]) AS [Last Use]
FROM vResource r INNER JOIN
Inv_AeX_AM_Monthly_Summary softstats ON r.Guid = softstats._
ResourceGuid INNER JOIN
AeXInv_AeX_SW_Audit_Software softinv ON r.ResourceId =
softinv.WrkstaId
AND softstats.[Internal Name] = softinv.InternalName
AND softstats.[Product Version] = softinv.ProductVersion AND
softstats.[File Name] = softinv.[File Name]
GROUP BY softinv._id
```

- We recommend that the following indexes be created in order to increase processing performance:

- **Inv_AeX_SW_Audit_Software** table (if the table exists):

```
CREATE UNIQUE
INDEX [idxCIT_Inv_AeX_SW_Audit_Software__id] ON [Inv_AeX_SW_Audit_
Software] ([_id])
ON [PRIMARY]
```

- **Inv_AeX_SW_Audit_Software_spt** table (if the table exists):

```
CREATE UNIQUE
INDEX [idxCIT_Inv_AeX_SW_Audit_Software_spt__id] ON [Inv_AeX_SW_
Audit_Software_spt] ([_id])
ON [PRIMARY]
```

- **ItemResource** table:

```
CREATE UNIQUE
```

```
INDEX [idxCIT_ItemResource_ResourceId] ON [ItemResource]
([ResourceI d])
ON [PRIMARY]
```

Altiris 7 Connector

The Altiris 7 connector is created to support Altiris Version 7.0 and 7.1. See the following table for the differences between the Altiris 7 connector and the Altiris 6 connector:

Difference	Altiris 7 Connector	Altiris 6 Connector
Prerequisites	Not required.	Required to create a view for LastUse type data processing.
Foreign key	ResourceGuid	WrkstaId
Root node	<ul style="list-style-type: none"> • Computer • Software • Network_Printer 	<ul style="list-style-type: none"> • Computer
Structure	<p>All the tables under the WrkstaID document type are moved to Computer.</p> <p>Note: The original data of all the modified relationships are displayed in the tables starting with INV.</p>	The WrkstaID document type contains fields and tables.
Document type	<p>The two connectors have inconsistent document types. Below are listed some examples of changed document types in the Altiris 7 connector:</p> <ul style="list-style-type: none"> • New - Chassis, ComputerSystem, DeviceDriverWindows, etc. • Obsolete - PCIBus, Videos, Modems, etc. • Restructured - OperatingSystem, AltirisSWInventory, SCSIController (SCSI), etc. 	

Out-of-Box Scenarios

See [Altiris Scenarios](#).

Configuring the Altiris Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Additional information

Unique reconciliation key

The **Guid** field of the **vComputer** view is used as the unique identifier for the mapping to an Asset

Manager table.

Schedule pointers

The **Collection Time** field in the **AltirisHWInventory** document type can be used as the schedule pointer. Only computers entered into the inventory table are taken into account. Computers that have not been entered into the inventory table must be scheduled separately.

Aperture VISTA Connectors

Aperture VISTA Web Service Connector

Aperture VISTA extends the management capabilities of AM for Data Centers. The Aperture VISTA Web Service connector is used for the data modification of Aperture VISTA. This connector is not a base connector and it is not free.

Aperture VISTA Portal connector

The Aperture VISTA Portal connector is used to read data from the Aperture VISTA Portal Database. The configuration of this connector is described in AM520-IntegrationWithDCIM-EN.pdf, which is supplied with Asset Manager 5.20.

Aperture VISTA Repository connector

The Aperture VISTA Repository connector is used to read/write data from/to Aperture VISTA Repository Database. The configuration of this connector is described in AM520-IntegrationWithDCIM-EN.pdf, which is supplied with Asset Manager 5.20.

Aperture VISTA Symbols connector

The Aperture VISTA Symbols connector is used to read data from the Aperture VISTA Symbols Database. The configuration of this connector is described in AM520-IntegrationWithDCIM-EN.pdf, which is supplied with Asset Manager 5.20.

Out-of-Box Scenarios

See [Aperture Vista Scenarios](#).

Configuring Aperture VISTA Web Service Connectors

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Define the connection parameters

Use the page to configure the connection parameters for Aperture VISTA server

- **Server:** Aperture VISTA server IP address or server name.
- **Context path:** VISTA/wsdl
- **Interface type:** The interface type exposed. Example: Form
- **Object type:** The object type that user will modify through a function. Example: EMAC_Install

- **Function:** The functions Aperture VISTA web service exposed for the object type. If you choose Form for the Interface type, the functions list can be displayed in this field. One of them that must be provided by Aperture VISTA server can be chosen. Example: getformstatus.

Define the security parameters

- **User:** The user name of the account for Aperture VISTA Web Services invoking
- **Password:** The password of the account for Aperture VISTA Web Services invoking

CA Unicenter DSM 11 Connector

Connector type: **PRODUCTION**

The CA Unicenter DSM gateway connector allows usage of databases whose data have been obtained using the inventory utility of the Unicenter DSM software suite. This inventory utility recuperates all the information for a set of computers. From this information the supplied CA Unicenter DSM to Asset Manager (amoac.scn) scenario allows you to create records in an Asset Manager asset table.

Known limitations

For proper usage of the CA Unicenter DSM connector, you are advised to use an ODBC driver and administration utility whose DLL versions are identical. For example, your CA Unicenter DSM connector will not function if you use a version 3.0 ODBC administration utility and a Microsoft Access 4.00 ODBC driver.

Out-of-Box Scenarios

See [CA Unicenter DSM Scenarios](#).

Configuring the CA Unicenter DSM 11 Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Prerequisites

Before configuring your connector, make sure a valid connection to your ODBC database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and password.

Published Document-type

The CA Unicenter DSM 11 connector publishes the following document types:

- Computer
- Domain
- Motor
- Group
- User

LANDesk for Inventory Connectors

Connector type: **PRODUCTION**

The LANDesk inventory connectors enable you to process data from databases retrieved using the LANDesk inventory tool. This inventory tool gathers extensive information about a computer population.

Known limitations

We highly recommend using the LANDesk connector with an ODBC driver and ODBC Administrator that use the same DLL versions. For example, the LANDesk connector will not work if you use ODBC Administrator version 3.0 and a Microsoft Access 4.00 ODBC driver.

Out-of-Box Scenarios

See [LANDesk Scenarios](#).

Configuring LANDesk for Inventory Connectors

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Prerequisites

Before configuring your connector, make sure a valid connection to your ODBC database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and password.

Production Directives of the LANDesk Connector

Published document-type

The LANDesk connector publishes a single document type **LD_OBJECTROOT**. This document type corresponds to the scan that LANDesk has performed on a computer.

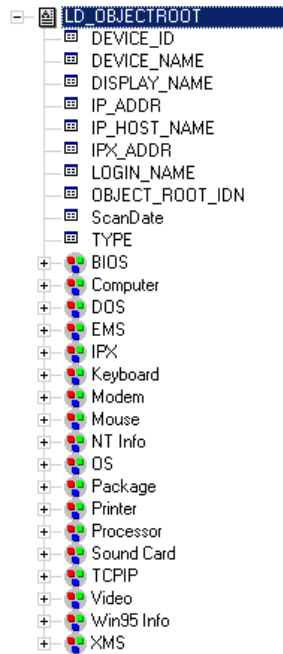


Figure 6-1: LANDesk LD_OBJECTROOT document type

LANDesk Software Distribution Connector

Connector type: **CONSUMPTION**

The LANDesk Software Distribution connector enables interaction with the LANDesk Web service. In an integration scenario, the LANDesk Software Distribution connector consumes a document and sends it as a query to a Web service, then receives a reply which it automatically transforms into a produced document.

Limitations of the LANDesk Software Distribution connector

Only the NTLM authentication protocol is used. You must use LANDesk Management Suite version 8.8 or later and the MBS SDK must be installed on the LANDesk server.

Out-of-Box Scenarios

See [LANDesk Scenarios](#).

Configuring the LANDesk Software Distribution Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Prerequisites

- A LANDesk account with administrative rights must exist. This account is populated when the connector is configured and saved in the LANDesk Software Distribution to Asset Manager scenarios. As the account that is used is a LANDesk administrator account and since the tasks modify information contained in LANDesk, you should restrict access to these tasks to persons authorized to create scheduled tasks in Asset Manager.

- LANDesk Software Distribution scenarios use inventory data that is retrieved from a network discovery scan to an Asset Manager scenario.
- Agents must be installed.

Specify the WSDL

This page enables you to enter the connector's connection parameters to a Web service.

- **WSDL address** field: The WSDL URL address enables you to retrieve for each Web service:
 - The communication protocols used
 - The supported operations.
 - The format of these operations
 - The data consumed and the data produced
- **Domain\login** field: User ID
- **Password** field

Microsoft System Center Configuration Manager (SCCM) Connectors

Connector type: **PRODUCTION**

The SMS 2003 (Microsoft Systems Management Service) and SCCM (System Center Configuration Manager 2007 and 2012) gateway connectors enable you to process data from SCCM data sources. The SCCM connectors only enable read-access to data in SCCM databases. They do not enable you to write data to an SCCM database.

Warnings relating to the gateway connectors

Before going to production phase with a scenario containing a gateway connector, we highly recommend testing the scenario on a copy of the database into which the data is to be imported (for example: a copy of your Asset Manager database).

This test phase makes it possible to:

- Validate the reconciliation keys used in the mappings. An invalid reconciliation key can bring about the creation of duplicate records in the target database. Removing these duplicate records manually is extremely difficult.
- Tailor the maptables, character string tables, and global functions. The standard maptables, string tables, and global functions provided with Connect-It or the sample scenarios may not match your needs exactly. For example: You may need to add entries in the **Brand** mactable. This table, which contains the brands of products (contained in the Genmaps.mpt file), may not contain all the brands you need.

Warnings related to the SCCM connectors

All SCCM connectors have been tested on original SCCM databases. The out-of-the-box scenarios might not properly work or might not open in the following two cases:

- The database to which your SCCM connector is connected has been customized (tables added or removed, field names modified, etc.). The document types produced by the connector may not include the elements mapped in the scenario.

- The database to which your SCCM connector is connected is not an SCCM database. The produced document-types will not include the elements mapped in the scenario.

Out-of-Box Scenarios

See [Microsoft SCCM Scenarios](#).

Configuring Microsoft System Center Configuration Manager (SCCM) Connectors

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Prerequisites

Before configuring your connector, make sure a valid connection to your SCCM database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and password.

Tally Systems TS.Census 3 Connector

Connector type: **PRODUCTION**

Note: *TS.Census version 3.0 is not supported. TS.Census 3.1 and 3.2 are still supported.*

The TS.Census gateway connectors enable you to process data obtained from Tally System TS.Census 3. This application belongs to the Tally System TS.Census 3 software suite. Tally System TS.Census 3 gathers information about an IT portfolio. Using this information, the out-of-the-box scenario lets you create records in the target database.

Known limitations of the TS.Census 3 connector

We highly recommend using an ODBC driver and ODBC Administrator using identical version levels of DLL files. For example, the TS.Census 3 connector will not work if you use ODBC Administrator version 3.0 and a Microsoft Access 4.00 ODBC driver.

Out-of-Box Scenarios

See [TS Census Scenarios](#).

Configuring the Tally Systems TS.Census 3 Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Prerequisites

Before configuring your connector, make sure a valid connection to your ODBC database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and password.

Published document-type

The **TS.Census 3** connector publishes all document types, including **NC_Workstation**, which is customized corresponding to the computer-related information gathered from a scan. For more information about document types, refer to the Tally System TS.Census documentation. All tables and fields are described in the appendices.

For more information about document types, refer to the Tally System TS.Census documentation for the correct version.

Tivoli Connectors

These connectors are for use with various types of IBM Tivoli software.

Out-of-Box Scenarios

See [Tivoli Scenarios](#).

Tivoli Inventory Connector (version 4.0)

Connector type: **PRODUCTION**

The Tivoli Inventory Management 4.0 connector enables you to process databases whose data was obtained using Tivoli Inventory 4.0. This application belongs to the Tivoli Inventory Management application suite. Tivoli Inventory gathers information from computer scans. With this information, the dedicated out-of-the-box scenario lets you create records in the destination database.

Known limitations

Proper use of the Tivoli Inventory Management 4.0 connector requires that you use an ODBC driver and an ODBC administration application whose DLL versions are identical. Example : Your connector will not work if you use ODBC Administrator version 3.0 and an Access 4.00 ODBC driver.

Configuring the Tivoli Inventory 4.0 Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Prerequisites

Before configuring your connector, make sure a valid connection to your ODBC database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and password.

Tivoli CM for Inventory 4.2 Connector

Connector type: **CONSUMPTION and PRODUCTION**

The Tivoli CM Inventory 4.2 connector enables you to process the databases whose data was obtained using the Tivoli Configuration Management version 4.2 application. This inventory function recovers complete information on an IT portfolio. From this information, the out-of-the-box scenarios migrate the data to an Asset Manager or ServiceCenter database.

Known limitations of the Tivoli CM - Inventory 4.2 connector

Proper use of the Tivoli CM - Inventory 4.2 connector requires that you use an ODBC driver and an ODBC administration application whose DLL versions are identical. For example: your Tivoli CM - Inventory 4.2 connector will not work if you use a version 3.0 of the ODBC administration software and an ODBC Microsoft Access 4.00 driver.

Configuring the Tivoli CM for Inventory 4.2 Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Prerequisites

Before configuring your connector, make sure a valid connection to your ODBC database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and password. The configuration of a connector in Connect-It is performed using a wizard.

Mandatory initialization of the status pointer

In a scenario that includes the Tivoli CM Inventory 4.2 connector, the value of the connector pointer must be '0'. By default, the Tivoli CM Inventory 4.2 connector is unable to give a value to this pointer before the first launch of a scenario in scheduled mode. Warning: Without this operation the Tivoli CM Inventory 4.2 connector cannot produce any documents.

To assign this value to the pointer:

1. Select the **Scenario > Scheduling** menu.
2. Associate the documents produced by the Tivoli CM Inventory 4.2 connector to the schedulers that you have already created.
3. For each produced document, double-click in the **Pointer status** column.
4. Enter **0** in the text box.

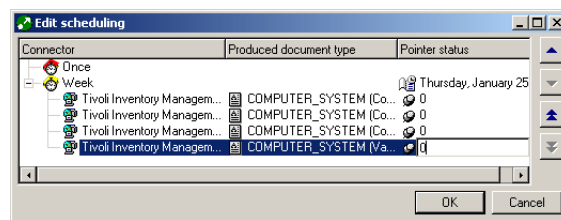


Figure 6-2: Scenario Scheduling

Published document-type

The Tivoli CM Inventory 4.2 connector publishes only one **COMPUTER_SYSTEM** document type. This document type corresponds to the scan that Tivoli Inventory launched on a given computer. The elements of this document type vary according to the configuration of your scanner generator. For example: Collections corresponding to database tables can occur or disappear from one scan to another one.

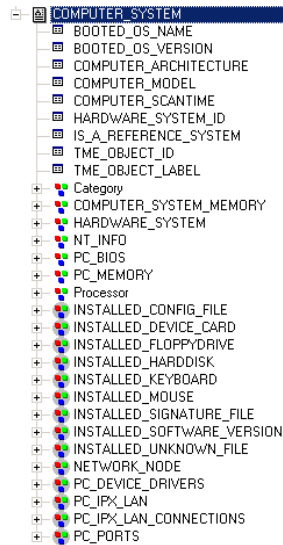


Figure 6-3: The **COMPUTER_SYSTEM** document type

Caution: The elements of this document type can vary according to the structure of your Tivoli Configuration Management 4.2 databases. In addition, the tables represented by collections in the **COMPUTER_SYSTEM** document type can be absent from the database. In this case, if the scenario mapping calls on a field from one of these absent tables, this scenario cannot run. Solution:

You must remove the implicated fields in the scenario mappings and the document types produced by the Tivoli CM Inventory 4.2 connector.

Production directives

The production directives of the Tivoli CM Inventory 4.2 connector enable it to filter and sort data from the ODBC data source. To filter and sort the data that will appear in the documents produced by the connector, you must write clauses similar to those used in an SQL query:

- The **WHERE** clause
This clause enables you to filter records in the ODBC database. You can, for example, filter records extracted from the database created on or after January 23, 2001. Simply use the following query:

```
[COMPUTER_SCANTIME] >= '2001-01-23'
```
- The **ORDER BY** clause
This clause enables you to indicate the field used to define the order in which documents are produced by the Tivoli CM Inventory 4.2 connector. You can enter a list of fields separating them with commas each time.
- **Do not use array fetching**
This option must be selected, for example, when recovering Blob-type fields or **memo**-type fields when the table does not have a main field. If array fetching is used, Blob-type fields may not be processed correctly (they are clipped). This option may reduce performance in particular.

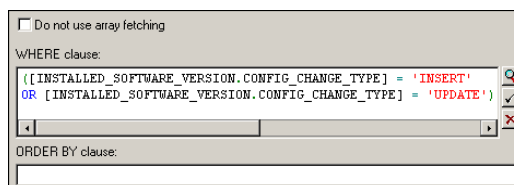


Figure 6-4: Tivoli CM Inventory 4.2 connector - production directives

Tivoli CM for Software Distribution Status 4.2 Connector

This connector enables you to read data from the **DIST_STATE** table.

Note: *Scheduling pointers are not supported for this connector. This connector is documented separately in the **TCM Integration** guide provided with Connect-It.*

Tivoli CM for Software Distribution 4.2 Connector

This connector reads Tivoli Region Manager data, in particular data from the **SD_PACKAGE** table.

Note: *This connector is documented separately in the **TCM Integration** guide provided with Connect-It.*

Chapter 7

Application Connectors

The application connectors are those which enable you to process the data from external applications or specific operating systems.

Action Request System Connector

Connector type: **CONSUMPTION and PRODUCTION**

The Action Request System connector enables you to process data from an Action Request System database.

Prerequisites: Action Request System connector: dynamic libraries

To use the Action Request System connector, you must make sure that the path of these libraries is defined in the appropriate environment variable

(%PATH% for Windows, \$LD_LIBRARY_PATH for Linux and Solaris, \$LIBPATH for AIX).

Known limitations

In read (production) and write (consumption) mode, the Action Request System connector processes the following Action Request System-type data fields:

- **Text**
- **Diary**

In read mode (production), this field enables you to recover all the history available for this field. In write mode (consumption), this field enables you to insert a string containing the name of the user who writes the data and the time and date it is written.
- **Integer**
- **Real number**
- **Radio button**

To use this field, you must indicate the number of the element that you want to initialize. You cannot directly indicate its value.
- **Drop-down list**
- **Date and time**
- **Attachment**

The connector publishes a structure for each attachment. This structure presents the following three fields:

 - **FileName**

Contains the full path of the attachment.
 - **AttachMemoVal**

In the case of an attachment file, this field contains the contents of the file.
 - **AttachBlobVal**

This file always contains the binary contents of the attachment.

In write mode (consumption), you must map the full path of the attachment to the **ARS attachment FieldName** field. If you want to create a copy of the memo or Blob file in a shared folder, you must map the path of the shared folder to the **ARS attachment FieldName** field. Then you need to map the contents of the file to the **AttachMemoVal** or **AttachBlobVal** field.

Note: Since the version 5.0 of the Action Request System, the memo and blob files are stored directly in the database. You no longer need to have a copy of these files in a shared folder.

In read mode (production), you must perform the following mappings:

- **AttachMemoVal** field to a memo-type field
 - **AttachBlobVal** field to a Blob-type field
 - **FieldName** field to a Blob-type field
- In the mapping edit window, you must make sure the **Load Blob from file** option is selected.

Unsupported field types

- Data types containing BLOB fields (**AR_DATA_TYPE_BYTES**)
- Attachments (**AR_DATA_TYPE_ATTACH**)
- The **STAT_HISTORY**, **VALUE_SET**, **LOCAL_VARIABLE**, and **QUERY** fields

Unix

DNS addresses are not supported on UNIX. You need to specify the IP address of the server.

Names of the fields in the document types published by the Action Request System connector

- If you connect to an ARS version 5 server, the attachments are directly saved on the server.
- In the document types published by the Action Request System connector, the "#" character is replaced by the "_" character.
For example, the **Field#1** field is displayed as **Field_1** in the published document type. In both cases, the queries using either **champ#1** or **champ_1** will produce the same result. In effect, Action Request System uses the displayed name or the internal Ids indifferently.
- In the WHERE or ORDERBY clauses:
 - To use the display name of a field, use the following syntax:
 `DisplayName`
 - To specify the system name of a field, use the following syntax:
 `[DatabaseName]`

Dynamic link libraries

The Action Request System requires certain DLLs. If they are not in the system path, an error message is displayed. To work around this problem, copy the .dll files listed below to the bin32 folder of the Connect-It installation folder. These files are, as a general rule, located in the sub-folders of the Action Request System installation folder.

ARS DLLs

	ARS 4.5	ARS 6	ARS 7	ARS 7.5
Connect-It requires the arapi DLL file:	arapi45.dll	arapi63.dll	arapi70.dll	arapi75.dll

	ARS 4.5	ARS 6	ARS 7	ARS 7.5
The <code>arapi</code> DLL file depends on other DLL files. The following are examples:	<code>arrpc45.dll</code>	<code>arrpc63.dll</code>	<code>arrpc70.dll</code>	<code>arrpc75.dll</code>
	<code>arutl45.dll</code>	<code>arutl63.dll</code>	<code>arutl70.dll</code>	<code>arutl75.dll</code>
	<code>arcatalog_eng.dll</code>		<code>icudt32.dll</code>	<code>icudt32.dll</code>
			<code>icuin32.dll</code>	<code>icuinbmc32.dll</code>
			<code>icuuc32.dll</code>	<code>icuucbmc32.dll</code>

Out-of-Box Scenarios

See [Action Request System Connector Scenarios](#).

Configuring the Action Request System Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Dynamic library to use

Enter in this field the full path to your dynamic library (for example, `arapi45.dll`) that enables you to connect to the Action Request System server. You must use a version 5.0 or higher of these libraries.

Format of the pointer

In this field, enter the name of the user format to apply to the pointer.

List of last-modification fields of the schemas (advanced mode)

The Action Request System schemas have a field indicating the last-modification date of a record in a schema. In most cases, the **Modified-date** field is used. Nevertheless, depending on your schemas and your version of Action Request System, the name of this field can vary. In this case, the editable zone that appears on this page enables you to indicate:

- The document type published by the connector corresponding to an Action Request System schema, for which you must specify a last-modification date field.
- The name of the last-modification date field used in the Action Request System schema. The default value of the last-modification date field is **Modified_date**. This value is used for all document types in which you do not enter values in the editable zone. You can specify a different name for the last-modification date field in each document type published by the Action Request System connector.

Caution: Do not use the name of the last-modification field indicated in the database. Use the name in the Action Request System scenario.

To enter a new association schema/ last-modification date field:

1. Click [✎](#).
2. Click the activated text zone in the **Diagram** column and enter the name of the document type (schema) published by the Action Request System connector.

3. Click in the corresponding **Field** column and enter the name of the last modification field used in this schema.

To delete an existing field:

1. Select the line corresponding to this field.
2. Click **x**.

IBM Websphere MQ Connector

Connector type: **CONSUMPTION and PRODUCTION**

Websphere MQ is an IBM application that enables the unique, asynchronous and stable transmission of data on numerous hardware and software platforms. It is an infrastructure for the communication between applications, either on the same machine or on different machines separated by one or more networks.

Websphere MQ can handle all of the most popular communication protocols and provides routes between networks using different protocols. With the integration scenarios, it enables us to exchange XML documents. The bridges and gateways of Websphere MQ enable you to easily access numerous application systems and environments, such as Lotus Notes, Web browsers and Java applets.

The numerous functions of Websphere MQ guarantee the transmission of data, even in the case of underlying system or network infrastructure failure. Websphere MQ data circulates in the form of **messages** containing the data that is exchanged between different applications. The messages are stored in data structures called queues. Under normal circumstances, messages can be placed in a queue or can be taken from the queue by applications or by a **queue manager**. External applications can put XML documents in an Websphere MQ queue. When connected to the appropriate queue, the Websphere MQ connector reads the messages and sends them to the HP applications specified in the scenario.

The exchange of data between Websphere MQ and HP applications can be bi-directional.

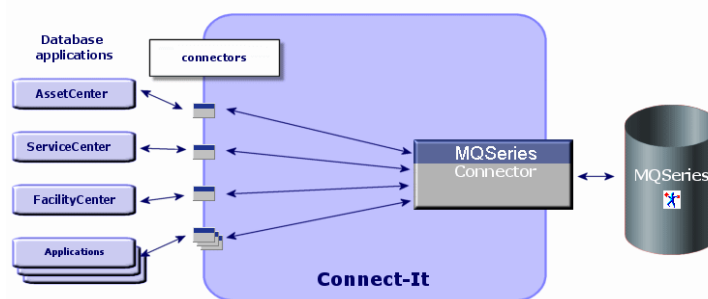


Figure 7-1: IBM Websphere MQ connector

Prerequisites

The Websphere MQ application client must be installed on the computer on which Connect-It is installed.

Out-of-Box Scenarios

See [IBM Websphere MQ Connector Scenarios](#).

Configuring the Websphere MQ Connector (read)

The basic configuration of the Websphere MQ connector in read mode enables you to specify the Websphere MQ queue in which it must read the data that it then transforms into Connect-It documents.

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Choosing a processing mode

The second page of the Basic configuration wizard enables you to choose in which mode you want to use the Websphere MQ connector. Select the **Read** mode.

Configure the queue manager connection

You must specify the following parameters in order for the connector to connect to a queue manager:

- **Use an extended connection**

This option is selected by default. Clear this option to use a standard connection. If you do not select this option, only the **Queue manager name** field can be populated.
- **Server name**

Enter the DNS name or IP address of your Websphere MQ server on your network.
- **Connection port**

Enter the listening port of your queue manager. By default, the value of this port is **1414**.
- **Connection channel name**

Enter the name of the connection channel that the queue manager uses as a communication path. The default value of the channel is **SYSTEM.DEF.SVRCONN**.
- **Queue manager**

Enter the name of your queue manager. In order to use the scenarios provided with the Websphere MQ connector, the name of this manager is **connectit**. If this field is not populated, the Websphere MQ connector will connect to the default queue manager of Websphere MQ.
- **Alternate user ID**

Specify an alternate user ID, if required, for opening the queue manager. You can specify different alternate user IDs for each queue (main, success, failure).

Choosing a queue

This page enables you to enter the name of the queue from which the Websphere MQ connector must read the data. To use the scenarios provided with the Websphere MQ connector, the name of this queue is **PEREGRINE.IN**.

Advanced configuration (advanced mode) option

This option enables you to define, for a queue:

- The name template for creating dynamic queues

Enter the format template for creating the dynamic queue name that will be used: The character '*' is replaced by the queue manager in order to guarantee the uniqueness of the dynamic queue's name.

You can choose from three options for closing dynamic queues:

- **Delete permanent dynamic queue if it is empty (MQCO_DELETE)**
Temporary dynamic queues are also physically deleted.
- **Retain dynamic queue if it is permanent and delete it if it is temporary**
- **Purge messages from permanent dynamic queue and delete it (MQCO_DELETE_PURGE)**
Temporary dynamic queues are also physically deleted.
- Authentication
For each queue, you can specify an alternate user ID.

Define post-processing actions

This page enables you to indicate how to process the messages from the queue after having been read by the Websphere MQ connector. In case of failure or success in processing a message, you have the choice of the following options.

- Leave the message in the queue.
- Delete it.
- Move it to a queue, which you will specify using the available text zone each time that this option is selected. When you select this option, you can specify whether you wish to keep the whole context of the message (default behavior), just the identity context, or no context at all. A context is formed as follows:
 - ID context: Information on the user inserting the message into the queue
 - Original context: Information concerning the inserted message (date, application that inserted the message)

In order to apply a post-processing action to the documents consumed by the other connectors and mapping boxes of your scenario, you must use the processing reports that each one produces.

Using post-processing actions

To use post-processing actions, you must create a mapping between the **PutDate** and **MsgID** elements of the **MessageInfo** structure of the document type produced by the source connector and the same elements of the **MessageInfo** structure of the **SuccessReport** document type consumed by the same source connector.

Configuring the Websphere MQ Connector (write)

The configuration of the Websphere MQ connector in write mode enables you to specify the Websphere MQ queue in which messages must be written from the Connect-It documents that it consumes.

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Also see [Configuring the Websphere MQ Connector \(read\)](#) for additional instructions.

Choosing a processing mode

The second page of the Basic configuration wizard enables you to choose in which mode you want to use the Websphere MQ connector. Select the **Write** mode.

Configure the queue manager connection

You must specify the following parameters in order for the connector to connect to a queue manager.

Choosing a queue

This page enables you to enter the name of the queue from which the Websphere MQ connector must read the data. To use the scenarios provided with the Websphere MQ connector, the name of this queue is **PEREGRINE.OUT**.

Specify the identity context option

Select this option to be able to specify the identity context of the messages you will write in the queue. Selecting this option modifies the structure of the produced document type **MessageInfo** and displays the following elements: **UserID**, **AccountingToken**, **APppIDentityData**. This option is not selected by default.

Production Directives of the Websphere MQ Connector

The production directives of the Websphere MQ connector enable you to filter the messages that the connector reads in the queue, specified during the configuration.



Figure 7-2: Websphere MQ connector production directives

These directives are:

- The **WHERE** clause
- Series of recovery options

To enter these directives:

1. Double-click the Websphere MQ connector.
2. Select a document type from the **Produced document types** tab in the dialog box that appears.
3. Select the root element of the produced document type.
4. Enter your production directives.

The WHERE clause

This clause enables you to filter the messages in the queue read by the Websphere MQ connector. This clause uses the following Websphere MQ syntax:

```
[key word]=[value]
```

The following key words are available:

- **MsgID** (Message ID)
- **GroupID** (ID of message group)
- **CorrelID** (ID of correlation)

The value of key words must be entered in a hexadecimal form.

In the Websphere MQ-Asset Manager scenario provided with the Websphere MQ connector, the following CorrelIDs are used for the produced document types:

- CONNIT.MQAM.REQUEST.ACK
(434f4e4e49542e4d51414d2e524551554553542e41434b) for the **ExtRequestAcks** document type
- CONNIT.MQAM.RECEIPT.ACK
(434f4e4e49542e4d51414d2e524543454950542e41434b) for the **ExtReceiptAcks** document type
- CONNIT.MQAM.VENDOR
(434f4e4e49542e4d51414d2e56454e444f52) for the **Vendors** document type
- CONNIT.MQAM.COSTCENTER
(434f4e4e49542e4d51414d2e434f535443454e544552) for the **CostCenters** document type

In the Websphere MQ-ServiceCenter scenario provided with the Websphere MQ connector, a CorrelID is used for the **External Contacts** produced document type:

CONNIT.MQSC.CONTACT
(434f4e4e49542e4d51414d2e434f535443454e544552)

These CorrelIDs must be applied in the Websphere MQ messages placed in the **PEREGRINE.IN** queue by external applications.

Type of messages to recover

This list enables you to select the type of messages recovered by the connector.

Conversion of messages

For all produced document types, the Enable conversion of messages to option is available. This option enables the conversion of queue messages to the following code pages:

- ISO-8859-1
- ISO-8859-2
- UTF-8

Get messages only if all messages in the group are available option

This option enables you to specify how the Websphere MQ connector recovers messages from the queue.

Consumption Directives of the Websphere MQ Connector

The consumption directives of the Websphere MQ connector enable you to specify the manner in which the connector will write messages in the queue, specified during the configuration.



Figure 7-3: Websphere MQ connector consumption directives

These directives are:

- The activation or not of the automatic segmentation of messages.
- The specification of particular IDs.
- The specification of the type of message to send.

To enter these directives, follow one of these procedures:

1. Double-click the Websphere MQ connector.
2. Select a document type from the **Consumed document types** tab in the dialog box that appears.
3. Select the root element of the consumed document type.
4. Enter your consumption directives.

Or:

1. Double-click a mapping box linked to the Websphere MQ connector in write mode.
2. Choose to edit or create a mapping in which the Websphere MQ connector is the destination connector.
3. Select the **Message options** tab.
4. Enter your consumption directives.

Automatic segmentation

If you authorize the segmentation of messages, the messages exceeding the maximum size limit in your queue will be cut into several physical messages. These physical messages are then regrouped into one logical message. If you prohibit the segmentation of messages, the messages exceeding the maximum size limit in your queue will be rejected by the Websphere MQ connector.

Specification of the IDs

These options enable you to specify particular IDs:

- Group ID
- Correl ID

The IDs must be entered in a hexadecimal form. In the Websphere MQ-Asset Manager scenario provided with the Websphere MQ connector, the following CorrelIDs are used for the consumed document types:

- **CONNIT.MQAM.REQUEST**
(434f4e4e49542e4d51414d2e52455155455354) for the **Request** document type
- **CONNIT.MQAM.RECEIPT**
(434f4e4e49542e4d51414d2e52454345495054) for the **Receipt** document type

In the Websphere MQ-ServiceCenter scenario provided with the Websphere MQ connector, a CorrelID is used for the Contacts from ServiceCenter consumed document type: CONNIT.MQSC.CONTACT (434f4e4e49542e4d51414d2e434f535443454e544552) The specified IDs identify your messages corresponding to purchase orders or receiving slips in the **PEREGRINE.OUT** queue. The external applications that read these messages must use the same IDs to recover these messages.

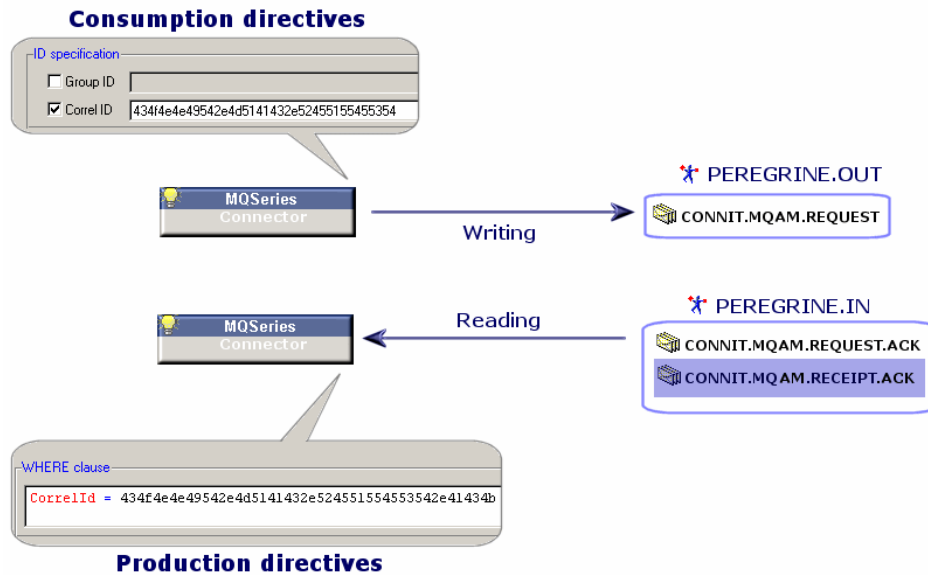


Figure 7-4: Websphere MQ connector IDs

Type of message

This frame enables you to:

- Select the type of message sent by the connector.
- Choose the queue containing the responses to the **Request** type messages.

Additional Information About the Websphere MQ Connector

MessageInfo structure in the consumed document types

- A **MessageInfo** structure is available in the consumed document types published by the connector in write mode. This structure can be used to specify certain parameters of the message to be sent. This structure contains the following fields:
 - **Expiry**
This field enables you to define the expiration delay of a message, after which, the message is deleted by the queue manager. The value of this field is expressed in tenths of a second. If a message does not expire, the value of this field is **-1**.
 - **Persistence**
This field concerning the persistence of the message can be set to the following three values:
0: If the queue manager is restarted, the message is not persistent.
1: If the queue manager is restarted, the message is persistent.
2: Default persistence of queue. This is the default value of the field.
 - **Priority**
This field enables you to specify the priority of the message. The value of this field must be an integer equal or greater than 0. If this field does not have a value, the priority of the queue is used by default.

- **CorrelationID**
 This field specifies the value of the document's **CorrelationID**. If you map this field, this value overrides the value of the **CorrelationID** specified in the connector's directives.
- **GroupID**
 This field specifies the **GroupID** of the message to send. If you map this field, this value overrides the value of the **GroupID** specified in the connector's directives.
- **AccountingToken**
 This field is an element of the identity context of the message. It only appears if the **Specify the identity context** option is selected in the connector configuration wizard.
- **AppIdentityData**
 This field is an element of the identity context of the message. It only appears if the **Specify the identity context** option is selected in the connector configuration wizard.
- **UserID**
 This field is an element of the identity context of the message. It only appears if the **Specify the identity context** option is selected in the connector configuration wizard.

MessageInfo structure in the document types produced in read mode

In the produced document types published by the connector, a **MessageInfo** structure is available. This structure contains some of the parameters of the message read. For more information about processing reports, refer to the Connect-It User guide. This document type contains the following fields:

• AccountingToken	• Persistence	• Report
• AppIdentityData	• Priority	• UserID
• BackoutCount	• PutDate	• MsgID
• Expiry	• ReplyToQ	• CorrelationID
• MsgType	• ReplyToQMgr	• GroupID

Using the MessageInfo document type

In write mode (sending messages), the connector publishes a document type, **MessageInfo**, produced each time a document corresponding to a message to be sent is consumed. This document type has the following fields, which correspond to the parameters of the message sent:

• AccountingToken	• Report
• AppIdentityData	• UserID
• MsgType	• MsgID
• PutDate	• CorrelationID
• ReplyToQ	• GroupID

Lotus Notes Connector

Connector type: **CONSUMPTION and PRODUCTION**

The Lotus Notes connector enables you to:

- In production mode, insert data from a Lotus Notes database into an external application (Example: a ServiceCenter database).
- In consumption mode, insert data from an external application into a Lotus Notes database.

Note: *To use the Lotus Notes connector, you must correctly install and configure the Lotus Notes client on the machine where Connect-It is installed. In particular, the PATH in Windows must include the path of the Lotus Notes folder.*

Known limitations

The Lotus Notes connector enables you to process the following field types only:

- Text
- Text list
- Number
- Date
- Lotus Notes Rich Text Format
This format enables you to preserve the formatting options used in Lotus Notes documents. In this case, all links to Notes documents and attachments are lost. Only the page format is conserved.
- Attachments

Out-of-Box Scenarios

See [Lotus Notes Connector Scenarios](#).

Configuring the Lotus Notes Connector

Before modifying the connector, you must first declare in the system environment variables the access path of the vim32.dll file in the Lotus folder.

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Define the connection parameters

The second page of the **Basic configuration of the connector** wizard enables you to configure the Lotus Notes connection. The four fields of this page enable you to specify the following:

- **Lotus Notes ID**
Indicate the .id file used to connect to your Lotus Notes database. This file contains your Notes ID and your password. Example: C:\Program Files\notes\data\user.id. This identification file must allow you to perform the required actions (read, write, delete) on the database selected in the **Database** field.
Warning: *This file must be the one used on the Lotus Notes client machine on which Connect-It is installed.*
- **Password**
Enter the password needed to connect to a Lotus Notes database. This password is not visible

when being entered and is stored in an encrypted format. The `nextpwd.dll` DLL file enables this password to be memorized and keeps you from having to enter it each time a scenario that includes the Lotus Notes connector is launched.

After installing the Lotus Notes connector, this DLL is located in the `bin 32` folder of the Connect-It installation folder. In order not to request the password each time the connector starts, follow these steps:

- a. Add the Connect-It `bin32` folder to the Windows environment variables,
- b. Add the following parameter to the `notes.ini` file after the `[Notes]` section:
`EXTMGR_ADDINS=nextpwd`
- c. Save your changes and restart Connect-It.
If you encounter conflicts with other programs, rename the `nextpwd.dll` file and update the `notes.ini` file as described previously.

- **Server name**

Select the server on which your database is installed. This field should display the servers to which your computer has access. However, if the name of the server that you want to use does not appear in this list, you can enter its address manually into the field. If you want to connect to a database installed on your computer, select **Local**.

Note: *If several Notes connectors are used in one or more scenarios executed on the same machine, they must be connected to the same Notes server and must use the same Notes ID.*

- **Database**

Enter the Lotus Notes database that you want to use in your Connect-It scenario. The entries in the drop-down list of the **Database** field:

- Depend on the chosen server name.
- If you have selected **Local** in the **Server name** field.

If the database of your choice is not shown in this field, you can enter it manually. In this case, you must indicate the **full path** of this database.

- **View all database fields option**

This option enables the connector to expose all the fields of a Lotus Notes database. If you do not select this option, only those fields declared in the forms are present in the document types published by the connector. This option exposes the **\$FILE** collection, which allows you to manage attachments.

Advanced options (advanced mode)

This page enables you to give specific information for certain produced document type elements. The status of a text field in Lotus Notes can change depending on what it contains:

- A text type field containing a character string is interpreted as an text type attribute.
- A text field that contains a character string containing semi-colons as separators is interpreted as a collection of attributes.

This page of the Lotus Notes connector configuration wizard enables you to identify the attributes that will be exposed as a collection of attributes:

- The **Name** column contains the full path in Connect-It (to which you must add the name of the document) that identifies the attribute in question.

- The **Value** column contains the NotesList string identifying a list and not a single attribute.

Additional Notes

- The case of binary Lotus-Notes fields
A Lotus Notes database contains documents with binary fields. They contain text and formatting information such as color, font, etc. In the document types published by the Lotus Notes connector, each of these fields is represented by two distinct fields:
 - A variable-length binary field
This field contains text and formatting information. This field is prefixed by LNRTB_ (Lotus Notes Rich Text Binary). In your mappings, these fields can only be mapped to other binary Lotus Notes fields (in a Lotus Notes to Lotus Notes scenario).
 - A long text field
This field contains text only. This field is prefixed by LNRTM_ (Lotus Notes Rich Text Memo). In your mappings, these fields can be mapped to other text type fields.
Note: *In the document types consumed by the Lotus Notes connector, only variable-length binary fields are available.*

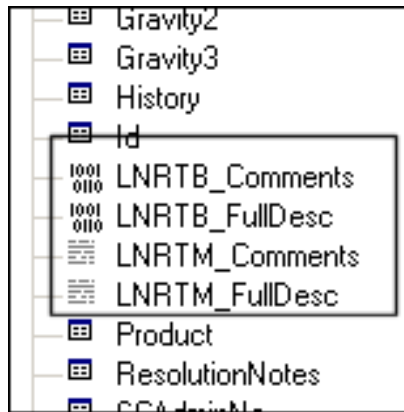


Figure 7-5: Lotus Notes LNRTM long text field

- Data processing in scheduled mode
If you use a local copy of a Lotus Notes database, the connector will not process the data modified between two sessions. In order for the connector to process your Notes documents, you must synchronize your local copy with the Notes server between two scheduled sessions. In the properties of a Notes document, there are two available modification dates:
 - An initial modification date (**initially**).
 - A modification date in this file (**in this file**).

Because the connector uses the initial modification date (updated each time the server is synchronized), it does not process the files that only had their 'in this file' date modified when the document was edited.

- Attachments
You can handle attachments by using the **\$FILE** collection. This collection contains two elements:
 - **Content:** Contents of the attached file in binary format
 - **Name:** Name of the attachment

Production Directives of the Lotus Notes Connector

For general instructions, see [Defining production Directives](#).

Design-form name

Each document type published by the Lotus Notes connector corresponds to a document from your Lotus Notes database. Each of these documents is associated with a form. The name of this form is included as a field of this document.

Sometimes several documents contain the same data but are associated with different forms. (example: a **Supplier** form and a **Contact** form). In order for your connector to produce the document corresponding to the form of your choice, you must select this name from the drop-down field in the **Design-form name** field.

The **@All form** is used to display data that is associated with each of the Lotus Notes forms. Using the **@All** form is the same as not applying a WHERE clause for a specific form.

Consumption Directives of the Lotus Notes Connector

The production directives of the Lotus Notes connector enable it to reconcile the records contained in the Lotus Notes database with the values of the documents consumed by the connector.

Design-form name

Each document type published by the Lotus Notes connector corresponds to a document from your Lotus Notes database. Each of these documents is associated with a form. The name of this form is included as a field of this document.

Sometimes several documents contain the same data but are associated with different forms. (example: a **Supplier** form and a **Contact** form). In order for your connector to create the document corresponding to the form of your choice, you must select this name from the drop-down field in the **Design-form name** field. Font This field enables you to select the character font used to format the data written by the connector.

Font

This field enables you to select the character font used to format the data written by the connector.

Reconciliation key

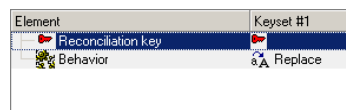


Figure 7-6: Lotus Notes connector reconciliation key

Use this option to indicate whether the current selection is used as a reconciliation key. To indicate that a field or a structure is a reconciliation key, you must:

- Select this element in the detail window.
- Then, select the **Reconciliation key** option or click directly on the transparent key in the pane showing the consumed-document type.

Case-sensitive reconciliation

[See Reconciliation](#).

Key set

[See Reconciliation.](#)

Reconciliation type

The options in the **Reconciliation type** field allow you define the reconciliation type for each parent node of your document type.

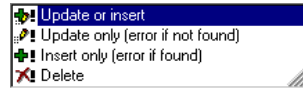


Figure 7-7: Lotus Notes reconciliation type

You can determine the reconciliation type for each non-terminal node of your consumed document-type. To do this:

1. Select a non-terminal node in your consumed document-type (root node, structure or collection).
2. Select the option of your choice in the **Reconciliation type** field. The four available options are:

- **Update or insert**

In this case, the data from the document consumed by the connector makes it possible to insert or update records in the destination Lotus Notes database.

- **Update only**

In this case, the data from the document consumed by the connector makes it possible to update records in the destination Lotus Notes database.

- **Insert only**

In this case, the data from the document consumed by the connector makes it possible to insert new records in the destination Lotus Notes database.

- **Delete**

In this case, the data from the document consumed by the connector makes it possible to delete records from the destination Lotus Notes database.

NT Security Connector

Connector type: **PRODUCTION**

This section describes the NT Security connector. This connector enables you to recover information concerning the NT domains of the computer on which Connect-It is installed. This information concerns the computers, users, and workgroups of the NT domains. This connector produces but cannot consume documents.

Compatibility of the NT Security connector

The NT Security connector is compatible with the 32-bit Windows network operating system. If you decide to use the new MS Active Directory security model, we recommend that you use it with the LDAP connector, since it is not supported by the NT Security connector.

Out-of-Box Scenarios

See [NT Security Connector Scenarios.](#)

Configuring the NT Security Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Enter an NT domain

Populate the **Domain** field with the name of the domain from which the NT Security information is recovered. By default, the domain name corresponds to the domain of the NT session in which Connect-It is started.

Note: *If you want to recover information coming from several domains, you need to enter the names of the domains to be processed and separate them with semi-colons (;). If you want to recover all the NT domains accessible via your computer, enter an asterisk (*). Be warned, though, that this operation can overload your computer's memory and stop the execution of Connect-It.*

Caution: *The short description that appears under the connector's name in the Scenario diagram is always the current name of the domain and the user of Connect-It. If you enter another domain in this page, it will not appear in the connector's short description.*

The **Use the domain controllers to recover a domain's list of computers** option lets you recover the full list of computers available on a given NT domain.

Document-types published by the NT Security connector

The NT Security connector publishes three document types:

- Machine
- NtDomain
- User

Note: *The NtDomain document type corresponds to the NT domains specified during the configuration of the connector.*

The different sub-nodes (structures and collections) of these document types published by the NT Security connector are:

- Computers of the domain (**Machine** collection)
- Users of the domain (**User** collection)
- User groups of the domain (**UserGroup** collections)

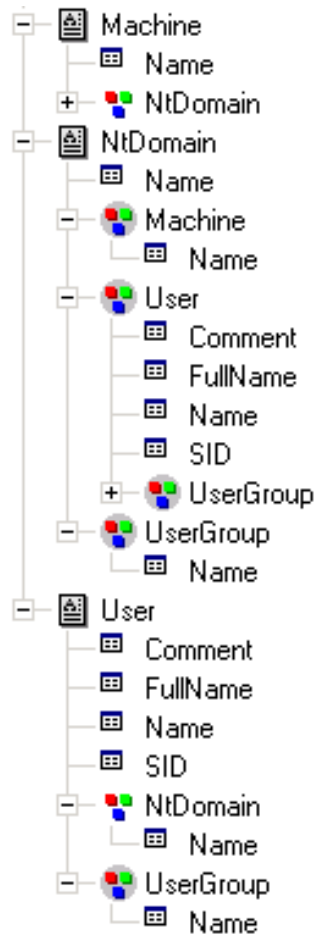


Figure 7-8: NT Security connector document types

Special notes for NT Security connector production directives

Populating the **Filter items by name** field enables you to filter members and their collections in produced document types. While creating this filter, you must use the special characters * and ?

Caution: To create a file, you must select a collection in the document type produced by the connector.

NT Security connector - Examples of production directives

Collection	Filter	Enables to recover
Machine	Platform 1	The Platform 1 computer
Machine	a*	All the computers from the domain whose name starts with a .
User	Doe?	All the users from the domain whose name starts by Doe and ends with an unknown letter.

Scheduling the NT Security connector

Asset Manager Server enables you to schedule the recovery of users declared in an NT domain. For further information on scheduling scenarios, refer to the Asset Manager documentation.

Tivoli Enterprise Console Connector (sending)

Connector type: **CONSUMPTION**

The Tivoli Enterprise Console connector (TEC) in sending mode enables you to send events to Tivoli Enterprise Console (TEC). This connector is a TEC adapter using the TEC EVD APIs to send events.

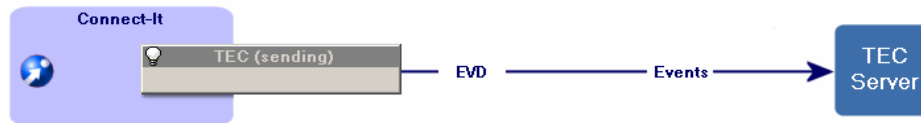


Figure 7-9: The TEC Console connector (sending)

Prerequisites

The Tivoli Enterprise Console connector uses Java Libraries from the EFI (Event Integration Facility) API which are supplied with the Tivoli Enterprise Console installation. These libraries are:

- [TEC installation folder]/bin/generic_unix/TME/TEC/evd.jar
- [TEC installation folder]/bin/generic_unix/TME/TEC/log.jar

To use the connector, you can:

- copy the libraries in the 'lib' subfolder in the Connect-It installation folder.
- modify the Connect-It global 'classpath' to add these libraries (**Java > Configure JVM** menu).
- add these libraries individually to the 'classpath' (Connector Configuration Wizard, **Configure JVM** page).

You must add the `eif.jar` library to the Connect-It class path. The `eif.jar` library is supplied with TEC 3.7.1 in the following folder: [TEC installation folder]/bin/generic-unix/TME/TEC. If you are using a previous version of TEC, please contact Tivoli customer support to get the latest library.

Out-of-Box Scenarios

There are no out-of-box scenarios for this connector.

Configuring the Tivoli Enterprise Console Connector (sending)

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Configure TEC server configuration


This page enables you to enter the configuration parameters to the TEC server.

- **Server**
Specify the name or address of the TEC server. This field is mandatory.
- **Port**
Specify the server port: The default port is **5529**. This field is mandatory.
- **The server does not support UTF-8**
By default, all the events sent to the TEC server are coded in UTF-8. Select this option if your server does not support UTF-8 (versions earlier than the version 3.7 of TEC). If this option is selected, the code page used by the server must be indicated in the **Server code page** field.
 - **Server code page**
If the previous option was selected, you must indicate in this field the code page used by the server.


Specifying the event classes

- **BAROC files**
You need to specify the BAROC files that contain the TEC event classes. The event classes of these files must correspond to those in the TEC server, which are defined in the BAROC files. The Tivoli Enterprise Console BAROC files contain a hierarchy of event classes. The order in which the BAROC files are declared is significant and reflects this hierarchy. Thus, if the two.baroc file defines an event A that inherits event B defined in the one.baroc file, then the one.baroc file must be listed before the two.baroc file.
Note: The scenarios based on a TEC connector prior to version 3.4 must be reconfigured. The fields contained in the event classes can be extended by an extension file.
- **Extension file**
The BAROC files defining the event classes handle the following data types: INTEGER, INT32, STRING and REAL. The Extension file field enables you to specify a BAROC extension file. This is an XML file that extends the definitions of the event classes contained in the BAROC files. This file:
 - Specifies a description for each event class and for each event-class field. This description is shown in the Connect-It document type editor.
 - Extends the event class field types. This makes it possible to declare fields as 'Date and time' type for easier handling in Connect-It.

To create / modify the extension file:

- a. Specify the path of the extension file.
- b. Click the magnifier icon .
- c. Define the regional settings used to convert event class fields, in particular for 'Date and time' type fields (Locale field). By default, the regional settings of Connect-It are used.
- d. Populate the list of event classes that will be extended.
- e. Click Next to extend the definitions of the event class fields.

***Note:** The event classes and their fields must be already defined in the BAROC fields otherwise an error will be recorded in the document log when the connector is opened. However, you are not obligated to extend all event classes or all fields of an event class. The event classes and fields that are not extended remain as defined in the BAROC files.*

- f. Click  to extend a field and populate its type. For 'Date', 'Time' and 'Date and time' type fields, you can specify the format used. The default format used for this field type is the number of seconds elapsed since January 01, 1970.
- g. Click **Finish** once all the fields have been defined.

Advanced configuration (advanced mode)

- **Activate the event buffer**

By selecting this option, the events sent to the server are stored in a buffer file in case the connection to the TEC server is interrupted.

Note: *If you do not select this option, any interruption to the connection with the TEC server will result in the rejection of the documents by the server.*

- **Buffer file**

Indicate the full path of your buffer file. This field is mandatory if the **Activate event buffer** option is selected.

- **Maximum size (KB) of the buffer file**

Here you specify the maximum size of the buffer file. If the maximum size is reached, and the connection to the server is interrupted, the TEC connector will reject any new document that it must consume.

Tivoli Enterprise Console Connector (receiving)

Connector type: **PRODUCTION**

The Tivoli Enterprise Console connector (TEC) in receiving mode is an event-driven connector that enables you to receive events from Tivoli Enterprise Console. A listening port specified in the connector's configuration enables it to listen for events sent by TEC. Rules defined in TEC enable you to send events to the TEC connector.

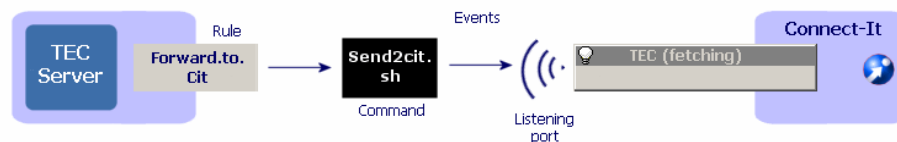


Figure 7-10: The TEC Console connector (receiving)

Prerequisites

The Tivoli Enterprise Console connector uses Java Libraries from the EFI (Event Integration Facility) API which are supplied with the Tivoli Enterprise Console installation. These libraries are:

- [TEC installation folder]/bin/generic_unix/TME/TEC/evd.jar
- [TEC installation folder]/bin/generic_unix/TME/TEC/log.jar

To use the connector, you can:

- copy the libraries in the 'lib' subfolder in the Connect-It installation folder.
- modify the Connect-It global 'classpath' to add these libraries (**Java > Configure JVM** menu).

- add these libraries individually to the 'classpath' (Connector Configuration Wizard, **Configure JVM** page).

In order for the TEC connector to receive events, you must create rules in TEC. For information about how to create rules in TEC, refer to the TEC Rules Builder guide.

Out-of-Box Scenarios

There are no out-of-box scenarios for this connector.

Configuring the Tivoli Enterprise Console Connector (receiving)

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Also see [Configuring the Tivoli Enterprise Console Connector \(sending\)](#).

Creating Event-Sending Rules in TEC

In order for TEC to send the events of an event class to Connect-It, you need to create a rule in TEC that launches the `send2cit` command. This command enables you to send back the events created in TEC to the TEC connector (receiving). For example, it enables TEC administrators to receive error warnings and thus implement specific problem-resolution procedures.

Note: *The `send2cit` command is a Unix shell script (.sh). However, this script can also be used with a Windows installation of TEC; a Unix shell is automatically provided with TEC.*

To send events to the connector, you must configure **send2cit** to give it the location of the connector, i.e. the name or address of the server hosting Connect-It and the listening port of the connector.

Location of the `send2cit` file

The **send2cit** command is available in the following folder:

```
[Connect-It installation folder]/datakit/tecevent_package.exe
```

The compressed auto-executable folder `tecevent_package.exe` contains the following files:

- `getvalue.exe`
- `tecevent.jar`
- `tecevent_lang.jar`
- `send2cit.sh`
- `tecevent.config`
- The Windows folder contains the `getvalues.exe` file.
- The Solaris, AIX and HP-UX folders each contain a `getvalues` file.

All these files (and the files `getvalues.exe` or `getvalues` corresponding to the operating system that is used) must be copied on the TEC server inside a unique folder.

Example - Sending events from the `TEST_CLASS` event class to Connect-It

The following example is a simple example of creating a rule in TEC to send an event to the TEC (receiving) connector when it is received by TEC. To create a rule enabling you to send events from the `TEST_CLASS` event class:

1. Launch Tivoli Desktop.
2. Create a new base rule.
3. In this base rule, create a new set of rules called, for example, **Forward**.
4. In this set of rules, create a new rule simply called, for example, **fwd_test_class**.
5. Specify that this simple rule applies to the **TEST_CLASS** event class.
6. Edit the rule actions.

To edit the rule's actions:

1. Create an action when you receive an event.
2. Select Launch a command.
3. Select the `send2cit.sh` command.
4. Edit the command's arguments:
 - **-p XXX**
XXX corresponds to the listening port of the TEC connector (receiving). This argument is mandatory.
 - **-s citserver**
citserver is the name of the server on which Connect-It is installed. This argument is optional. If this argument is not defined, the server is the computer on which TEC is installed.
 - **-l logfile**
Logfile is the full path of the log file. This argument is optional. If this argument is not defined, a log file will not be created.
5. Save the base rule (Forward).
6. Build the base rule.
7. Load the base rule.

After you load the base rule, each `TEST_CLASS` event received by TEC is resent to Connect-It and processed by the TEC connector (fetching). For more information about creating rules in TEC, refer to the TEC documentation.

Configuring the `send2cit.sh` File

The `send2cit.sh` file may be configured:

- from the command line
- using a configuration file

If you use a configuration file to configure the `send2cit.sh` file, this overrides the command line options.

Note: If the `send2cit.sh` script in the `tecevent datakit` does not work on UNIX, enter this command to make it executable:
`chmod 755 send2cit.sh`

Configuring from the command line

Prior to Connect-It version 3.4.0, the `send2cit` command could be configured from the command line. This functionality is still supported, however we now recommend using a configuration file. The command parameters are case-sensitive.

Note: The `-h` command-line parameter displays help on the **`send2cit`** command.

The location of the TEC (receiving) connector is given by two parameters:

- The name or the address of the server hosting Connect-It. This is specified using the `-s` parameter. The default value is 'localhost'.
- The listening port of the connector. It is specified using the `-p` parameter.
For example: `send2cit -s connectitsvr -p 12345`

Using a log file from the command line

The **`send2cit`** command enables you to write to a log file. You can specify the log file using the `-l` parameter. You can also specify a maximum size, in kilo-bytes, for a log file by using the `-m` parameter. For example: `send2cit -s connectitsvr -p 12345 -l /tec/connectit/send2cit.log -m 2048`

Configuration using a configuration file

The `send2cit` command can read its settings from a configuration file. The configuration file makes it possible to use advanced functionality of `send2cit`. The `send2cit` configuration file uses the same syntax as the property files:

```
# Comment  
Property = Value  
OtherProperty = Other value
```

A sample configuration file is provided with Connect-It.

Note: The `-listCfgProps` command-line parameter displays the list of properties of the configuration file and their descriptions.

Defining the configuration file to use

By default, `send2cit` uses the `tecevent.config` file if found next to the `send2cit.sh` file. If this file does not exist, or if you wish to use different one, you can use the `-cfg` command-line parameter. For example: `send2cit -cfg /tec/connectit/myConfig.conf`

Location of the TEC (receiving) connector

The location of the TEC (receiving) connector is given using by two properties in the configuration file:

- **Server:** Name or address of the server hosting Connect-It. The default value is 'localhost'.
- **Port:** Listening port of the TEC (receiving) connector.
Note: The `-s` and `-p` command line parameters, if they are used, override the values given the configuration file.

Using a log file

Two properties in the configuration file enable you to specify a log file:

- **LogFile:** Full path of the log file to use.
- **MaxLogSize:** Maximum size in kilobytes, of the log file. The default size is '2000'.

The **-l** and **-m** command line parameters, if they are used, override the values given the configuration file.

Defining generic event classes

In a Tivoli Enterprise Console configuration, an event class is often used as a parent class for set of event classes. For example, an event class, NetworkProblem, will be used as a parent class for event classes NP1, NP2, NP3, NP4... The NPx event classes don't clog up the fields of the NetworkProblem parent class, however their names identify the network problem.

To process these events in Connect-It, use the same mapping for all event classes deriving from NetworkProblem, to open a ticket in ServiceCenter, for example. To achieve this, define a generic event class, NetworkProblem, in the send2cit configuration file that groups the NPx classes together. When send2cit is invoked to send an event from the NPx class to Connect-It, it will send a NetworkProblem event. When generic event classes are defined, the real event class remains available in Connect-It in the 'EVENT_CLASS' element of the document type produced by the TEC (receiving) connector. Generic event classes are defined in the configuration file using the prefix 'G_':

```
G_MyGenericClass = SpecificClass1 SpecificClass2 SpecificClass3
```

In our example, to define the generic event class, NetworkProblem, the configuration file must contain the line:

```
G_NetworkProblem = NP1 NP2 NP3 NP4
```

Handling connection problems

It is possible to specify a connection timeout for the TEC (receiving) connector. The connection is considered as having failed beyond this time. This delay is specified using the TimeOut property and its value is expressed in milliseconds. By default, when the connection to the TEC (receiving) connector fails and the event log is not activated, an error is raised and an error code is returned by send2cit.

It is also possible to tell send2cit to send an event to TEC when the connection fails. This event thus enables the TEC administrator to rectify the problem. The class of the event sent is CIT_TecEvent_ConnectionFailure. To activate this functionality, the following properties are used:

- **SendEventOnCnxFailure:** The value of this property must be set to 'true' to enable sending the event.
- **CnxEventFailureSeverity:** Severity of the event sent. The default value is 'WARNING'.

Sending events to TEC requires additional configuration of send2cit and TEC.

Using an event cache

The cache folder stores events that could not be sent if an error connection occurs between the TEC server and the TEC (receiving) connector. When the connection is restored, the contents of the cache is sent back to the server. In this way, no events are lost.

Note: *If an environment has more than one firewall, the TEC host might not be able to cache events.*

Two properties are used to specify the use of a cache:

- **UseCache:** The cache is enabled if the value of this property is set to 'true'.
- **CacheDir:** This property specifies the folder to be used to store the event cache

If you enable the cache, you must specify a folder to use.

When send2cit fails to send an event to the TEC (receiving) connector, this event is stored in the cache in the form of a file in the cache folder. The name of the folder is formed as <server>_<port>_XXXX.cache, where XXXX is a sequence guaranteeing the uniqueness of the file name. When the next call to send2cit is made, if the connection to the connection is reestablished, send2cit starts by purging the cache before sending the new event; for example, it starts by sending back the events stored in the cache.

Limiting the size of the cache means limiting the number of events saved in the cache folder. By default, the size of the cache is limited to 100 events. To change this value, use the **MaxCacheSize** property. The **MaxCacheSizeAction** property determines the behavior of send2cit when an event cannot be sent and the cache has already reached its maximum size.

The possible values are:

- **Drop:** The event is rejected, an error message is generated and an error code is returned by send2cit.
- **Error:** The event is placed in the cache, but an error message is generated and an error code is returned by send2cit.
- **Warning:** The event is placed in the cache, and a warning message is generated.
- **None:** The event is placed in the cache, without any other action. This value is the equivalent of disabling the cache.
- **SendEvent:** The event is placed in the cache but an event is sent to the TEC server.

The default behavior is 'None', i.e. the size of the cache is not limited by default. When the MaxCacheSizeAction property is set to SendEvent, an event of the CIT_TecEvent_CacheFull class is sent to Connect-It. The CacheFullEventSeverity property enables you to specify the severity level of the event sent. The default severity is 'MINOR'. Sending events to TEC requires additional configuration of both send2cit and TEC.

Sending events to TEC

In order for send2cit to be able to send events to TEC if a connection problem occurs or if the cache reaches its maximum size, send2cit and TEC must be configured. The following properties are used to send events to TEC:

- **TECServer:** Name or address of the TEC server to send the event to. The default value is 'localhost'.
- **TECPort:** Connection port of the TEC server to send the event to. The default value is '5529'.
- **SendEventCommand:** Command used to send the event. The default value 'postemsg'; In this case, the postemsg command must be in the PATH. It is possible to specify the full path of the command.
- **EventSource:** Source of the event sent

The event classes sent by send2cit are defined in the BAROC file `cit_event.baroc` provided with send2cit. In order for TEC to receive events sent by send2cit, you must therefore import this file into TEC. The send2cit event classes derive from the same parent class **CIT_TecEvent_Error**, which defines the four following fields:

- `tecevent_error_code`
- `src_event_handle`
- `src_date_reception`
- `src_server_handle`

The **tecevent_error_code** field contains the send2cit error code that provoked the sending of the event. The three other fields makes it possible to correlate the event with the event at the origin of the send2cit error. The source of the events sent by send2cit, and defined by the **EventSource** property in the send2cit configuration file, must also be declared in TEC.

send2cit error codes

1. Error while recovering the environment variables containing the event data to send.
2. Bad command-line parameters.
3. Connection problem while sending the event to the TEC connector (receiving).
4. Error while reading the configuration file.
5. Error while processing the command-line parameters.
6. Error while processing the environment variable: A variable is missing.
7. Unknown error.

JMS for IBM WebSphere MQ Connector

Connector type: **CONSUMPTION and PRODUCTION**

This connector implements Java Message Service describes the API and required semantics for IBM WebSphere MQ. The following communication modes are supported by the connector:

- Point to point (queues)
The message is consumed by a single client.
- Publication / subscription (distribution lists)
All clients that have subscribed to the distribution list consume the message.

The JMS connector enables XML documents to be exchanged via JMS. The connector is used to define the following behavior:

- Interaction with the JNDI provider:
Connection address, root class that enables access to ConnectionFactory objects, Queue, Topic
- Definition of the connection with the JMS server
- Interaction with a proxy server

Out-of-Box Scenarios

See [IBM Websphere MQ Connector Scenarios](#).

Configuring the JMS for IBM WebSphere MQ Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Configure the connection to the JNDI provider

The JNDI (Java Naming and Directory Interface) extension is a Java standard that enables a client to access a directory. Information for the connection to the JMS server is saved in a directory. Access to this directory requires information about the JNDI provider.

- **Class:** Select or enter the JNDI provider's initial context factory implementation class
(`javax.naming.spi.InitialContextFactory`)
File `com.sun.jndi.fscontext.RefFSContextFactory`
LDAP `com.sun.jndi.ldap.LdapCtxFactory`
This class must be present in the connector's classpath. You must add the JNDI client's Java libraries (jar) that correspond to the JNDI provider to the connector's classpath (**Configure the JVM** wizard page).
- **URL:** Enter the URL of the JNDI directory. This connector uses the IBM WebSphere MQ file system and LDAP JNDI providers. The URL has the following format:
`file:///C:/JNDI-Directory` or `ldap://yourserver/cn=jms, dc=root, dc=hp, dc=com`
- You can enter additional properties in the Name and Value fields to create the initial JNDI context. The name and the manner in which these additional properties are interpreted depend on the JNDI provider. For example: The security properties can be configured for LDAP JNDI provider as following:
 - **java.naming.security.principal** `cn=root, dc=hp, dc=com`
 - **java.naming.security.credentials** `root`

Configure the connection to the JMS server

The connection to an IBM WebSphere MQ server is made using the JMS connection factory (**javax.jms.ConnectionFactory**). This page enables you to configure the connection to the MQ server. The following fields must be populated:

- **ConnectionFactory:** Name of the JNDI context containing the JMS connection factory.
- **Destination:** Enter the JMS destination (queue or distribution list) to read or send XML documents.
- **Enable MQMD properties:** If the option is selected, the MQMD structure will be shown in the `JMSMessageInfo` document type. MQMD properties display as document attributes where you can set values under the MQMD structure and the documents can be sent in a JMS message or read from a JMS message. For more information about how to use these properties, see the relevant IBM topic [Message object properties](#).

Define post-processing actions

After a file is read by the JMS connector, Connect-It proposes the following options, which depend on processing success or failure:

- **Do nothing**
- **Copy the message to another destination**

Consumption of a JMS message destroys the message. Once consumed, the message is no longer delivered (and will be destroyed if the destination is a queue). If the Copy the Message to another destination option is selected, a copy of the message will be sent to the selected destination. The value of the correlation identifier (**JMSCorrelationId**) for this destination will be set to the value of the originating message's identifier (**JMSMessageID**). This function is linked to the use of processing reports.

Specify the XML schema of the messages

Messages are transmitted in XML format. This page lets you define the URL of the xml schema that is used to process the messages.

JMS for IBM WebSphere MQ Connector Published and Consumed Documents

Published Documents

- In read mode:
Documents that correspond to the XML schema plus a JMSMessageInfo structure that contains information about the JMS message.
Directives:
Message filter: JMS filter used to filter received messages depending on the properties of the JMS messages.
- In write mode:
A document type produced on JMSMessageInfo consumption containing information about the message that is sent.

Consumed Documents

- In read mode:
Nothing
- In write mode:
Documents corresponding to the XML schema plus a JMSMessageInfo structure used to specify information about the JMS message.
Directives:
 - Codepage:
UTF-8 by default
 - Priority:
Priority of the JMS message that is sent
 - Life cycle:
Life cycle of the JMS message that is sent (0=unlimited)
 - Persistence:
Send or do not send a persistent JMS message

Chapter 8

ERP Connectors

The ERP (Enterprise Resource Planning) connectors enable you to process data from Enterprise Resource Planning applications.

RFC (Remote Function Calls)

The RFCs are used to the SAP BAPI connectors. They enables the SAP or non-SAP applications to access the functions of an SAP R:3 server in a network architecture.

The RFCs:

- Enable you to call on and process procedures predefined on an SAP server.
- Manage communications, parameter transfers, and error messages.
- Are controlled by the standard SAP authentication procedures. These authentication procedures verify that the application data is read and edited in a secured and consistent way.

An SAP R/3 server provides thousands of functions, most of which can be remotely accessed. These are usually written using the **ABAP/4** (Advanced Business Application Programming) language. The RFCs are bi-directional, which enables an SAP client application to access the functions in the non-SAP applications.

RFC libraries and Java archives

To use the SAP connectors, you must use Java archives (.jar) and native Java libraries (.dll, .so or .o) that you obtain using the following procedure:

1. Launch your Internet browser.
2. Enter the following address: **http://service.sap.com/connectors**.
Note: To access this site, you must obtain an authorization certificate from SAP support.
3. Select the **SAP Java Connector** section.
4. Select the **Tools & Services** section.
5. Click **Download SAP JCo Release 2.1.2**.
This download enables you to obtain a compressed file (.zip or .tgz) that contains the files necessary for the SAP connectors to work.
Note: Other versions of SAP JCo are supported.
6. Uncompress the compressed file on your disk.
7. Copy the `sapjco.jar` archive to the `lib` sub-folder of the Connect-It installation folder.
8. Install the native Java libraries. This step depends on your operating system.

Copy the following libraries to the `bin32` or `bin` sub-folder of the Connect-It installation folder:

- `librfc32.dll` and `sapjcorfp.dll` for Microsoft Windows operating systems
- `librfccm.so` and `libsapjcorfc.so` for Linux and Oracle Solaris operating systems
- `librfccm.o` and `libsapjcorfc.o` for IBM AIX operating systems

SAP ALE Connector

The SAP ALE connector processes the IDoc files like the SAP IDoc connector. However, it uses the ALE (Application Link Enabling) communication layer to send and receive these files to and from the SAP R/3 server. The events that it receives (corresponding to produced document types) can also use the SAP R/3 server RFCs. Unlike the SAP IDoc, it supports both synchronous and asynchronous data processing (event-driven mode).

Prerequisites

The following elements must be defined for the SAP server:

- logical system
- RFC destination
- tRFC port
- distribution model
- partner profile

Out-of-Box Scenarios

There are no out-of-box scenarios for this connector.

Configuring the SAP ALE Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

See [Configuring the SAP BAPI Connector](#).

SAP BAPI Connector

Connector type: **CONSUMPTION**

The SAP BAPI connector connects to an SAP R/3 server and obtains a list of (BAPI) services sorted by business object. The document types published by the SAP BAPI connector are organized as follows:

- Consumed document types
These document types let the connector send data to the SAP server.
- Consumed-Produced document types
These document types let the connector:
 - Send a request to the SAP server (consumption)
 - Receive a response from this request (production)

The way an SAP connector works when using consumed-produced document types is illustrated by the following diagram.

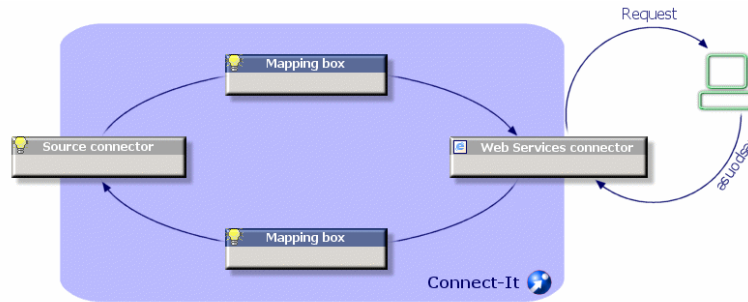


Figure 8-1: SAP BAPI Connector request and response

Limitations of the SAP BAPI connector

This connector does not work with versions earlier than 3.5 of SAP R/3, which introduced the use of BAPIs.

ABAP data types

The following chart shows how the ABAP data and the Connect-It data are equivalent.

ABAP data and Connect-It data

ABAP type	JCO data type	Connect-It data type
C	TYPE_CHAR	Text
D	TYPE_DATE	Date
P	TYPE_BCD	Double integer
T	TYPE_TIME	Date
X	TYPE_BYTE	Blob
N	TYPE_NUM	Integer
F	TYPE_FLOAT	Double integer
I	TYPE_INT	Integer
b	TYPE_INT1	Integer
s	TYPE_INT2	Integer
g	TYPE_STRING	Text
y	TYPE_XSTRING	Blob

Out-of-Box Scenarios

See [SAP BAPI and Web Service Scenarios](#).

Configuring the SAP BAPI Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that

section, they are available below.

Configure SAP server connection

This page enables you to indicate the connection parameters of the connector to the SAP server.

- **SAP server**
Indicate the name of your SAP server or the routing string to access the server. For example:
`/H/saprouter1/H/saprouter2/H/sapserver`
- **System number**
Indicate the system number that you want to use with your SAP server.
- **Client**
Indicate the client port. Example: **800**
- **Login**
Indicate your user name.
- **Password**
Indicate the password corresponding to the user name, specified in the previous field.
- **Language**
Indicate the language in which the information from the server should appear. Example: **en** for English, **de** for German, etc.
- **Enable Auto Commit/RollBack mode after each document**
This option enables you to automatically enable the **BAPI_TRANSACTION_COMMIT** BAPI call (or **BAPI_TRANSACTION_ROLLBACK**) after the BAPI user call in order to validate the changes. This option can only be used in write mode.

SAP IDoc Connector

Connector type: **CONSUMPTION and PRODUCTION**

Services and events for SAP IDoc use a flat file format (located in a folder shared by the network or on the FTP) to pass information from and to SAP R/3. The IDoc is a data structure defined by SAP that supports data exchange between any two applications in a common format. The SAP R/3 system provides hundreds of IDocs, but it also allows you to create your own IDoc types or expand existing ones.

An IDoc instance consists of three, distinct parts:

- **A header**
This header contains a string that lets the SAP R/3 server to uniquely identify an Idco file.
- **Data**
This data is organized in segments.
- **A series of statuses**
The file went through all of these statuses on its path to or from the SAP R/3 server.
Caution:For the IDoc files containing multiple documents, each document must be separated by a header (EDI_DC segment).

Known limitations of the SAP IDoc connector

For the connector to function correctly when using the HTTP and FTP protocols, the dlls `wininet.dll` and `shlwapi.dll` must be installed on the Connect-It client machine.

Out-of-Box Scenarios

See [SAP IDOC Scenarios](#).

Configuring the SAP IDoc Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

A consumed document type corresponds the sending of the data to an IDoc file. A produced document type corresponds reading of the data in an IDoc file.

If you wish to use multiple IDoc file types, you must deploy as many connectors as there are different IDoc files. Since each connector has a unique behavior (production or consumption), you must create as many connectors as there are behaviors.

IDoc type definition (read and write)

This page enables you to choose the IDoc document types and the way you want to get it:

- **IDoc types in local**
The **Local file** option lets you use the IDoc types already present on your computer.
- **IDoc types of the SAP server**
The **On the SAP server** option enables to dynamically obtain the IDoc types on your SAP server. The available fields change depending on which of these options you select.

IDOC type file

If you selected the option **Local IDoc types**, click **Browse** to find the location of the IDoc type on your computer.

SAP version (write)

Indicate the version of the SAP you use.

SAP server connection settings

The following fields enable you to indicate the following SAP server connection parameters:

- SAP server
- User
- Password
- Language
- Client
- System identifier

IDoc settings

- **Basic type**
Name of the SAP document you want to produce. For example, **MATMAS01**.
- **Extension**
This field enables you to enter:

- A CIM type for the IDoc types (version **2** or **3**)
- An SAP server CIM type (available versions: **3.1 G** to **4.6 C**)
- **Segment release**
SAP version on which the segment depends. The most recent version of the application is taken into account and is adapted to the oldest version of the segment.
- **Version of IDoc record type**
Version 2 record types correspond to SAP 3.x. Version 3 record types correspond to SAP 4.x.

SAP Web Service Connector

Historically, SAP business data and processes are externally available through ABAP-based SAP proprietary technologies (e.g., IDOC, ALE, BAPI). Since WebAS 6.20, SAP started to make business functions available through standard based web service technologies. This connector is used to access ABAP business functions exposed by SAP as web services.

Out-of-Box Scenarios

See [SAP BAPI and Web Service Scenarios](#).

Configuring the SAP Web Service Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Define the connection parameters

Use this page to indicate the connector's connection parameters to the SAP Web Services.

- **Server**
Name or IP address of your SAP server.
- **Port**
Port of the SAP server. Default port is **80**.
- **Client**
Specify the SAP client. Default client number is **000**.
- **ServiceType**
Three types of service are available:
 - **Out-of-the-Box RFC Web Services (default)**
Since WebAS 6.20, all out-of-the-box RFC-enabled function modules are exposed as web services by default. Choose this type if you want to invoke these oob modules.
Note: All RFC-enabled function modules can be accessed through the following URL :
`http://<Server>:<Port>/sap/bc/bsp/sap/WebServiceBrowser/search.html`
 - **User-defined Services**
Since WebAS 6.40, web services can be defined for RFC-enabled function modules, function groups, business objects. You can build your own function modules, or add additional logic to

existing function modules, and create your own ABAP web services for them. Choose this type if you want to invoke these user-defined web services.

- **XI Web Services**

Choose this type if you want to invoke web services provided by SAP XI (Exchange Infrastructure) .

- **ServiceName**

Name of the function module you want to invoke.

- **Login**

Indicate your user name.

- **Password**

Indicate the password corresponding to the user name, specified in the previous field.

Chapter 9

Protocol Connectors

Protocol-type connectors are those which enable you to process data when the processing uses a particular communication protocol.

Command-line Connector

Connector type: **CONSUMPTION and PRODUCTION**

This section describes the Command-line connector.

- In consumption mode, the Command-line connector enables you to execute a command line on the computer on which Connect-It is installed.
- In production mode, the Command line connector enables you to recover from a command line executed on a computer where Connect-It is installed:
 1. Return values
 2. Standard output
 3. Error output

The connector produces documents automatically. In a scenario, each time the Command line connector consumes a document (coinciding with the launch of a command line sent by another connector), it automatically produces the corresponding document.

Compatibility of the Command line connector

The Command line connector enables you to launch a command line on any operating system on which Connect-It can be installed. In Unix, the connector only recovers the return code of commands. The standard output and the error output cannot be recovered.

Out-of-Box Scenarios

There is no out-of-the-box scenario for this connector.

Configuring the Command-line Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Execution frequency

This page enables you to specify a pause between the processing of a determined number of documents. This prevents your computer, or an involved server, from being overloaded. This page contains the following fields:

- **Number of documents**
You enter the number of documents processed by your computer between each pause.

Example: If you enter the number **100** and a pause of **30 s**, your computer will process 100 documents followed by a 30-second pause. After this pause, another 100 documents will be processed, and so on.

Note: *The values of the **Pause duration** and **Number of documents** fields depend upon your computer's performance level. You should conduct some tests before setting values for these fields.*

- **Pause duration**

This pause must use the following conventions: s for seconds; m for minutes; h for hours.

Example: If you enter **30 s** for **100** documents, your connector will process the first 100 documents, stop after 30 seconds and then process the following 100 documents.

- **Timeout**

In order for this field to be available, you need to have already selected the **Kill the process if it has not finished before timeout** option. Enter a value in seconds.

Caution: *In Unix, the **Kill the process if it has not finished before timeout** option is not available.*

Command-line Connector Published Document-Types

The Command-line connector publishes two document types:

- The **Command** document type

This document type corresponds to a command line executed on the computer on which Connect-It is installed. This published document-type enables you to define the document types consumed by the Command-line connector.

This document type has three fields:

- The **Arg** field

This field corresponds to your command line arguments.

- The **Name** field

This field corresponds to the full path of the executable (the target). For example:

```
C:\Program Files\HP\Asset Manager 4.40 en\bin
```

If your executable is defined in your operating system's Path environment variable, you can directly specify the name of the executable.

Example: `explorer.exe`.

- The **Path** field

This field corresponds to the command-line execution folder. It corresponds to the field **Start in**.

- The **CommandReturn** (production) document type

This document type corresponds to the return values of a command line executed on the computer on which Connect-It is installed. This published document-type enables you to define the document types produced by the Command-line connector.

This document type has three fields:

- The **ReturnCode** field

This field corresponds to the return code of the executed command line.

- The **StdErr** field

This field enables you to obtain data on the standard error output.

- The **StdOut** field
This field enables you to obtain data from the standard output.
In Unix, the **StdErr** and **StdOut** fields are not available.

BIOS Commands

Certain commands are executables and others are BIOS commands. To use a BIOS command in the command-line connector, you must use the following syntax with the `cmd.exe` executable:

```
cmd /C move d:\HR\Employee.xml d:\HR\done\Employee.xml
```

In Connect-It, the **Name** element is mapped with "cmd /C move" and the **arg** element mapped with "d:\HR\Employee.xml d:\HR\done\Employee.xml".

Command-line Connector Consumption Directives

Only one consumption directive is available for the Command-line connector. This directive corresponds to the **Synchronous execution** option available in the edit window of the document types consumed by the connector. In synchronous mode (by default), a command line is executed after the previous command line has finished executing. All messages resulting from this execution (including error messages) will be processed by the connector.

Database Connector

Connector type: **CONSUMPTION and PRODUCTION**

The Database connector enables you to process data from an ODBC data source or directly from an Oracle, Sybase, DB2 or MySQL database. Starting with Connect-It version 4.10, you can set multiple owners for a databases (ODBC, Oracle, Sybase, and DB2). Then you can configure every owner's schema in a connector. If there is a duplicate table name when you set multiple owners for a database connector, the display-name will append to the table name and appear as "table name(display-name)". The default display name is the owner name. You can use advanced options to set the display name and use the new name to define a document type or consume data just like the others.

To use the ODBC connections, the ODBC Administrator must have been previously installed.

Known Limitations

We highly recommend using the Database connector with an ODBC driver and ODBC administrator using identical version levels of DLL files. When you set multiple owners, you will not be able to define duplicate display names or add a new owner when the owner has a table name duplicated with a defined document type.

For example: the Database connector will not work if you use ODBC Administrator version 3.0 and an Access 4.00 ODBC driver.

Note: *The limitations of the database engine (number of tables in the SQL query, etc.) and the ODBC driver also apply to the Database connector.*

Type of supported fields

The document types published by the Database connector are made of fields. Each of these fields has a particular type: text field, lob field, Blob field, etc. Depending on the database driver used, the field types supported by the Database connector are different.

DB2 MVS - Microsoft ODBC driver for DB2

The following configurations have been used:

- DB2 Connect DB2 SDK 8.1.10
- ODBC IBM DB2 driver 8.01.00.36
- Z/OS V1.2 DB2 V7.2 server

DB2 MVS - Microsoft ODBC driver for DB2

Field type	Read-only support	Write support
Blob	No	No
character(n)	Yes	Yes
date	Yes	Yes
decimal(p,s)	Yes	Yes
float(p)	Yes	Yes
integer	Yes	Yes
long varchar	Yes	Yes
timestamp	Yes	Yes
varchar	Yes	Yes

DB2 - Native connection

Field type	Read-only support	Write support
bigint (64-bit integer)	Yes (for 32 bit integers only)	Yes (for 32 bit integers only)
blob	No	No
character	No	No
clob	Yes	Yes
date	Yes	Yes
datalink	No	No
decimal	Yes	Yes
dbclob	No	No
double	Yes	Yes
char	Yes	Yes
graphic	No	No
integer	Yes	Yes

Field type	Read-only support	Write support
real	Yes	Yes
time	Yes	Yes
smallint	Yes	Yes
timestamp	Yes	Yes
varchar	Yes	Yes
varchographic	No	No

Oracle 10g and 11g

Oracle 10g/11g - Native connection			Oracle10g/11g - Microsoft ODBC driver for Oracle	
Field type	Read-only support	Write support	Read-only support	Write support
bfile	No	No	No	No
blob	Yes	Yes	Yes	Yes
char	Yes	Yes	Yes	Yes
clob	Yes	Yes	Yes	Yes
date	Yes	Yes	Yes	Yes
decimal	Yes	Yes	Yes	Yes
float	Yes	Yes	Yes	Yes
integer	Yes	Yes	Yes	Yes
interval year	No	No	No	No
interval day	No	No	No	No
long	Yes	Yes	Yes	Yes
long raw	Yes	Yes	Yes	Yes
nclob	Yes (unicode version: Windows and Linux)	Yes (unicode version: Windows and Linux)	Yes	Yes
nchar	Yes (unicode version: Windows and Linux)	Yes (unicode version: Windows and Linux)	Yes	Yes

Oracle 10g/11g - Native connection			Oracle10g/11g - Microsoft ODBC driver for Oracle	
Field type	Read-only support	Write support	Read-only support	Write support
nvarchar2	Yes (unicode version: Windows and Linux)	Yes (unicode version: Windows and Linux)	Yes	Yes
number	Yes	Yes	Yes	Yes
raw	Yes	Yes	Yes	Yes
rowid	No	No	No	No
smallint	Yes	Yes	Yes	Yes
timestamp	No	No	Yes	Yes
timestamp with time zone	No	No	No	No
timestamp with local time zone	No	No	No	No
varchar2	Yes	Yes	Yes	Yes
xmltype	No	No	No	No

Microsoft SQL Server

Field type	Read-only support	Write Support
bigint	Yes	Yes
binary	Yes	Yes
bit	Yes	Yes
char	Yes	Yes
date	Yes	Yes
datetime	Yes	Yes
datetime2	Yes	Yes
datetimeoffset	Yes	Yes
decimal (numeric)	Yes	Yes

Field type	Read-only support	Write Support
float	Yes	Yes
image	Yes	Yes
int	Yes	Yes
money	Yes	Yes
nchar	Yes	Yes
ntext	Yes	Yes
nvarchar	Yes	Yes
real	Yes	Yes
smalldatetime	Yes	Yes
smallint	Yes	Yes
smallmoney	Yes	Yes
text	Yes	Yes
time	Yes	Yes
timestamp	Yes	No
tinyint	No	No
uniqueidentifier	Yes	Yes
varbinary	Yes	Yes
varchar	Yes	Yes

Notes:

- For the *bigint* type, Connect-It uses *double* to store *int64* data.
- For the *decimal (numeric)* type, if *scale=0* and *precision<20*, it is represented as *integer* without lost precision; otherwise it is represented as *floating point number* with lost precision.

MySQL - Native connection

Field type	Read-only support	Write support
bigint	Yes	Yes
bit	Yes	Yes
blob	Yes	Yes
bool	Yes	Yes

Field type	Read-only support	Write support
char	Yes	Yes
date	Yes	Yes
decimal	Yes	Yes
double	Yes	Yes
double precision	Yes	Yes
enum	No	No
float	Yes	Yes
int	Yes	Yes
longblob	Yes	Yes
longtext	Yes	Yes
mediumblob	Yes	Yes
mediumtext	Yes	Yes
numeric	Yes	Yes
set	No	No
smallint	Yes	Yes
text	Yes	Yes
time	Yes	Yes
timestamp	Yes	Yes
tinyblob	Yes	Yes
tinyint	Yes	Yes
tinytext	Yes	Yes
varchar	Yes	Yes

Sybase12

Sybase12 - Sybase System 11 driver version 3.11.00.01		Sybase12 - Native connection	
Field type	Supported	Read-only support	Write support
binary	Yes	No	No
bit	---	No	No

Sybase12 - Sybase System 11 driver version 3.11.00.01		Sybase12 - Native connection	
Field type	Supported	Read-only support	Write support
char	Yes	Yes	Yes
datetime	Yes	Yes	Yes
decimal	Yes	Yes	Yes
float	Yes	Yes	Yes
image	Yes	Yes	Yes (version 2.70 and higher)
int	Yes	Yes	Yes
money	Yes	Yes	Yes
nchar	Yes	No	No
ntext	Yes	No	No
numeric	Yes	Yes	Yes
nvarchar	Yes	Yes	Yes
real	Yes	Yes	Yes
smalldatetime	Yes	Yes	Yes
smallint	Yes	Yes	Yes
smallmoney	Yes	Yes	Yes
text	Yes	Yes	Yes
tinyint	Yes	Yes	Yes
varbinary	Yes	No	No
varchar	Yes	Yes	Yes

Out-of-Box Scenarios

There is no out-of-the-box scenario for this connector.

Configuring the Database Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Certain options concerning this connector can be enabled using the **Edit > Options > Connector** menu.

Prerequisites

Before configuring your connector, make sure a valid connection to your ODBC database has been declared in ODBC Administrator. A valid ODBC declaration must use an appropriate login and passwords.

Advanced configuration (advanced mode)

- **SQL92 supported:** Select this option if you use the MySQL extension for the ANSI SQL92 standard.
- **Execute an initial import:** This option lets you avoid having to use SELECT queries in reconciliation scripts for initial imports (for example, when the database is empty or the items to be inserted do not yet exist). This shortens processing time.

Advanced options (advanced mode)

- **ODBC 3 compatibility with Oracle:** Add the **IsODBC3Compliant** advanced option to ensure compatibility. When the ODBC driver is not ODBC 3 compatible, define the value as follows: **IsODBC3Compliant=0**.
- When you set multiple owners, you must set the display name for duplicated table names. Use **ownerdiaplsynname.ownername** as the option name (**ownerdispalyname** is a keyword and **ownername** is the database owner that you are using).
- **QuoteChar:** This option is for the Oracle (native) connection type. Set **QuoteChar** value as " when a field name is the same as the reserved word of the Oracle database or the field/table names in the database are in lower case.

For more advanced options of the Database connector, see [Configuring Additional Advanced Options](#)

Production Directives

For this connector, these directives involve writing **WHERE** and **ORDERBY** clauses, which enable you to filter records from the source database.

The **ORDERBY** clause enables you to indicate the field used to define the order in which documents are produced by the database connector. You can enter a list of fields separating them with commas each time.

***Note:** The **WHERE** and **ORDERBY** clauses apply at the root level of the document type produced and at the level of the collection.*

Sysdate field

In the **WHERE** clauses written in AQL, the use of the **sysdate** field must be replaced with the function **getdate()**.

NULL

When a "Numeric" type field is not populated (its value is NULL), Connect-It sets its value to "0". Similarly, the absence of a link is comes out as "Link = 0" or "foreign key = 0". Example: "Location=0" or "lLocald=0".

Consumption Directives

To set a connector's consumption directives, you must enter reconciliation parameters in the **Reconciliation** and **Advanced reconciliation** tabs.

Additional Information

- **Additional information about declaring your ODBC data source**

Additional information about declaring your ODBC data source. When a scenario using a Database connector is launched as a service in 32-bit Windows, the ODBC data source must be declared as a system data source (system DNS) and not as a user data source (user DNS).

- **Verifying that an ODBC data source is a system data source**

To verify that a data source is a system data source:

- Start ODBC Administrator.
- Check that the ODBC data source used by your connector is located in the **System DNS** tab. If it is not, delete it, and then create a new ODBC data source.

- **Avoiding database reserved words**

For the connection types of MS SQL (ODBC), Oracle (native) and MySQL (native), pay attention that the following situations will cause data processing failure and using reserved words should be avoided:

- A table name is identical to any reserved word of the database.
- A field name is identical to one of these reserved words: DISTINCT, FIRST_ROWS, FROM, HAVING, INTO, NOT, NULL, SELECT, VALUES, and WHERE.

Note: The Oracle database requires additional configuration of QuoteChar when using reserved words. For more information, see "Advanced options for the inventory and database connectors" in this guide.

Delimited-text Connector

Connector type: **CONSUMPTION and PRODUCTION**

The Delimited-text connector processes files containing a flat representation of database records. Field values are either aligned in columns of fixed width (expressed as a number of characters) or separated by a delimiter character (a punctuation mark, tab, or any specified character).

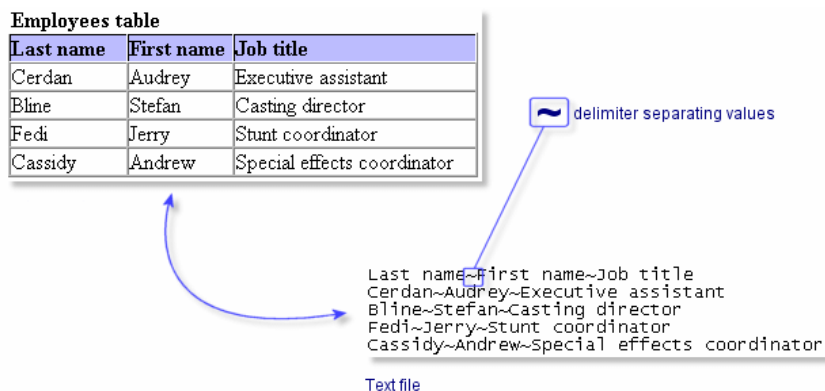


Figure 9-1: Delimited-text connector separating values

- In read mode, data is read from a text file so that it can then be imported into a database via a Connect-It connector (Database, Asset Manager, ServiceCenter connectors, etc.)

- In write mode, text files are created. The data contained in these file originates from a database via a Connect-It connector (Database, Asset Manager, ServiceCenter connectors, etc.).

The text enables the:

- Processing of text files on the computer or the network on which Connect-It is installed. These files can be compressed using the gzip command.
- Processing of text files on FTP sites and Web sites (read-only for the latter).

Known limitations of the Delimited-text connector

The only limitation of the Delimited-text connector concerns the format of **Date** and **Date and Time** type fields. In order to be processed correctly, these fields must be formatted according to the regional settings of the computer on which Connect-It is installed. This limitation can be bypassed by using a string type field and associating it with a user format during the mapping.

FTP and HTTP protocols

For the connector to function correctly when using the HTTP and FTP protocols, the dlls `wininet.dll` and `shlwapi.dll` must be installed on the Connect-It client machine.

Using IPv6

If you access a file through HTTP on a remote server using IPv6, you will need version 7 of the `wininet.dll` file.

You can do either of the following:

- install Internet Explorer 7
- download and save version 7 of this file to the Connect-It computer (default path: `C:\Windows\System32`)

Out-of-Box Scenarios

There is no out-of-the-box scenario for this connector.

Configuring the Delimited-text Connector (read)

Certain options concerning this connector can be enabled using the **Edit > Options > Connector** menu.

Path of a file or folder

Often when configuring a connector, you must indicate the path of a file or folder in the configuration wizard. If this file or folder is in a folder associated with a network drive on your computer, you must never indicate the letter of this drive; only indicate the folder associated with this letter. Indicating the network drive in the path stops the scenario from working correctly when it is associated with a service.

Choosing a processing mode

This page enables you to indicate if the connector is used in read or write mode.

Selecting a connection protocol

The three available options are:

- HTTP Web site
- FTP server
- Local or network file(s)

Choose a file or folder

If you choose to read local or network files, you must:

1. Choose a file or folder
2. Select an action to perform after the files have been processed.

You can choose between two options:

Read a file

If you select this option, the wizard will ask you to enter the path of the text file on your computer or your network.

Read files in a folder

If you select this option, the wizard will ask you to populate the following fields:

- **Folder name**
Enter the path of the folder that contains your files.
- **Extension**
Indicate the file name extension that the connector must read. For example, 'txt'.

Post-processing actions

After the text connector has read a file, Connect-It will give you three options:

- Leave it in the folder
- Delete it from the folder
- Move it to another folder
If you choose this last option, you must indicate the path of the folder to which you want to move the processed file.

You must specify one of these options in the case of failure, and one in the case of success, in processing the text files by your connector. In order to apply a post-processing action to the documents consumed by the other connectors and mapping boxes of your scenario, you must use the processing reports that each one produces.

Note: *The post-processing action (in the case of success or failure) does not apply unless the processed file contains data.*

In scheduled mode, when you select the "On successful processing of a file: Leave in the folder" option, all the files are processed, with no regards to their modification date (there is not schedule pointer).

Connecting to an HTTP Web site

If you have chosen to read files on a Web site, you must indicate the HTTP connection parameters:

- **Address**
Enter an address of the type: **[protocol]://[address]:[port]/[path]**. The **Address** field in must begin with **http://** when using text connector to access a HTTP server. The usual port number for an HTTP server is **80**.

- **Login**

Enter the login that enables you to access the chosen site.

- **Password**

Enter the password corresponding to your login.

Secured connection (HTTPS protocol)

This option enables you to indicate if you want to connect to your site via a secured connection (HTTPS).

***Important:** If the [protocol] part of the Web site's address does not correspond to the HTTPS protocol, the selection of the Secured connection (HTTPS protocol) option forces the use of an HTTPS protocol. Typically, the port number for an HTTPS server is **443**.*

Client certificate

This field enables you to select an HTTPS certificate from among those present on your computer.

***Important:** If you change or delete a client certificate specified in this field after the connector has already read or written to these documents, you must close and reopen the Scenario Builder in order for this modification to be accounted for.*

Manage the list of client certificates

The list of client certificates proposed by the configuration wizard corresponds to the list of certificates in your `Root console/ Certificates - current user/ Personal/ Certificates` folder of the Microsoft Management application.

To add certificates to this folder using Windows XP:

1. Select **Run** from the Windows **Start** menu.
2. Enter **MMX** in the **Open** field.
3. Select **File > Add/Remove Snap-in**.
4. Click **Add** in the window that is displayed.
5. Select **Certificates** in the window that is displayed.
6. Click **Add**.
7. Select **My user account in the dialog box** that is displayed.
8. Click **Finish**.
9. Click **OK**.
10. Add or delete the files in the `Certificates - Current user/ Personal/ Certificates` folder.

Do not verify the server identity

Use this option to support a proxy server in a secure connection. If a proxy server is installed, the address shown in the server certificate does not match the one shown for the proxy server. Selecting this option disables the automatic verification of the server's identity.

Connect via a proxy

This option is available for HTTP and FTP connections. Use this option to access a proxy server for a given connection. When you select this option, enter the name (or IP address) of the proxy server, its port and, if necessary, select the **Use authentication** option.

Cascading proxies are supported if the same authentication is used for all proxies.

Do not use a proxy server

For HTTP type connections, you can specify conditions when the proxy server should be bypassed (**Do not use a proxy server for** field). You must enter an HTTP type address in this field depending on the connection that is used. Names must be separated by a space.

FTP Server

FTP server If you have chosen to read the text files on an FTP site, you must:

1. Select the FTP connection parameters
2. Select an action to perform after the files have been processed.

After configuring all FTP server items, you can test your connection by clicking **Test**.

In the next page of the wizard, select:

Read Files

If you select this option, the wizard will ask you to enter the path of one or more text files on your computer or your network.

Read Folders

- Path of the folder on the FTP site.
- File extension names. By default, the value of this field is **txt**.

Read the sub-folders

If you select this option, the connector will also read the files in the sub-folders of the selected folder.

Note: *During a connection in FTP mode, you might see the following error: **Error: 12015**. This error means that the directory is not available, usually because of too many simultaneous connections.*

Define post-processing actions

After the text connector has read a file, Connect-It will give you three options:

- Leave it in the folder
- Delete it from the folder
- Move it to another folder

If you choose this last option, you must indicate the path of the folder to which you want to move the processed file. You must specify one of these options in the case of failure, and one in the case of success, in processing the text files by your connector.

Using post-processing actions

To use post-processing actions, you must create a mapping between the **UriFileInfo.Path** structure of the document type produced by the source connector and the **UriFileInfo.Path** element of the **SuccessReport** document type.



Note:

The post-processing action (in the case of success or failure) does not apply unless the processed file contains data.

In scheduled mode, when you select the "On successful processing of a file: Leave in the folder" option, all the files are processed, with no regards to their modification date.

Choosing a Description File

In order to process a text file, the connector must use a description file. Two scenarios are possible:

- The description file already exists.
Indicate its path in the **DSC file** field. Click  to create a location for the file.
- The description file does not exist.
Indicate the path and the name of the description file you want to create in the **DSC file** field and click  to launch the description file creation wizard. This wizard is presented in the Creation of the description file (DSC file).

Note: *If you use two delimited text connectors in the same scenario, a backup copy is created for the first DSC file in order not to be overwritten by the second connector. If you are using a UNICODE version, an additional option enables you to select the encoding type to use.*

Options associated with the Delimited-text connector

You can access the options of the Text connector using the menu **Edit > Options > Connector > Delimited text and XML**.

The options associated with the Text connector in FTP protocol are the following:

- Copy locally the files to read from FTP server.

Select this option to make a local copy of the files on an FTP server and to read the data from the local file. This option must be selected if the network configuration does not enable a connection to be maintained on the FTP server long enough for a file to be processed.

Configuring the Delimited-text Connector (write)

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

HTTP Web site

If you choose to write files on an HTTP site, you must:

- Select the HTTP connection parameters.
- Select an action to perform after the files have been processed.

HTTP connection parameters

On this page, you must populate three fields, which enable you to connect to your HTTP server.

- **Address**
Specify the connection address to the HTTP server. The **Address** field in must begin with **http://** when using text connector to access a HTTP server.
- **Login**
Enter the login that enables you to access the chosen site.
- **Port**
The usual port number for an HTTP server is **80**.

- **Password**

Enter the password corresponding to your login.

Write command

There are two write commands used to write to an HTTP server:

- **POST**

Sends data to the program located at the indicated address. It is different from the PUT method in that the sent data must be processed.

- **PUT**

Sends data so that it can be saved at the indicated address. Select the command used by your HTTP server.

HTTPS protocol

Select this option if you want to connect to a server using this secure protocol. The usual port number for an HTTPS server is **443**. The usual port number for an HTTP server is **80**.

For further instructions, see [Configuring the Delimited-text Connector \(read\)](#).

Behavior between two sessions

This page enables you to select how documents are processed between two data-writing sessions. The behavioral options are different if you write data to one single file or if you write it to several files in a folder.

Write the documents to a single file

To write to a single file, you have the following options:

- **Append to the same file**

If you choose this option, the connector starts writing data to the open file again the moment the previous session is interrupted.

- **Overwrite the last file**

The connector deletes the file in which it wrote data during the previous session.

- **Number the different files**

The connector creates a file whose name is the same as the previous file, but incremented by 1 unit. Example: file.xml, file1.xml, file2.xml, etc.

Write to a different file for each document


To write in several files, you have the following options:


- **Delete all previous files and start numbering them from the beginning** The files in the folder are all deleted (all the files dating from all the previous session; not just from the last one). The connector starts incrementing all the files it writes.

- **Continue numbering files** The connector starts writing files again and incrementing them without deleting the existing files.

Choosing a Description File

In order to process a text file, the connector must use a description file. Two scenarios are possible:

- The description file already exists.
Indicate its path in the **DSC file** field. Click  to create a location for the file.
- The description file does not exist.
Indicate the path and the name of the description file you want to create in the **DSC file** field and

click  to launch the description file creation wizard. This wizard is presented in the Creation of the description file (DSC file).

Note: *If you use two delimited text connectors in the same scenario, a backup copy is created for the first DSC file in order not to be overwritten by the second connector.*

Blind mode

If you select Activate blind mode, the Delimited-text connector will create text files without a description file. The values contained in these text files are separated by one of the chosen delimiters. The delimiters to choose from are:

- Tab
- Comma
- Semi-colon
- Space
- Other

Enter the symbol that you want to use as a separator in the **Other** field.

Operation of the Delimited-text connector in blind mode

To operate the Delimited-text connector in blind mode:

- Configure your Delimited-text connector by selecting the **Activate blind mode** option.
- Choose a database-type connector (for example: the Asset Manager connector) for which you create produced document-types whose fields contain values that you want to create in the text file or files.

In blind mode, only the values of fields directly under the root node of produced document-types are written in the text files. If the document type includes fields of structures or collections under the root node, their values are not written to the text file or files created.

- Directly link the database-type connector to your Delimited-text connector in your scenario without passing through a mapping box. (To create a direct link, just hold down the **Shift** key.)
- Put your database-type connector into production (**Produce now** command in the **Tools** menu).

Text files created in blind mode

If the **Write documents to one single folder** option is selected in the **Basic configuration** wizard, the file created in blind mode:

- Will be composed of lines, all of which correspond to the documents consumed by the Delimited-text connector.
- Will have a name that corresponds to that of a document type produced by the source connector.

If the **Write to a different file for each document** option is selected in the **Basic configuration** wizard, the files created in blind mode:

- Will contain values that are found in one of the documents consumed by the text connector.
- Will have a name composed of:
 - The name of the document type produced by the connector source (example: **Test**).
 - A number corresponding to the document production order (example: **Test_01**, **Test_02**, **Test_03**, etc.).

Writing the DSC file

The Write DSC file option enables you to create a description file in blind mode. If you select **Enable blind mode**, then this option is also selected by default. If you use the name of an already existing file, Connect-It will automatically save it, and attach to it a number: **_01**. This description file uses the delimiter chosen for blind mode.

Encoding to use

If you are using a UNICODE version, an additional option enables you to select the encoding type to use. Please refer to the Compatibility Matrix for the list of available encoding types.

Creation of the description file (DSC file)

A description file (file name extension .dsc) describes the way that the text-file data is organized. This data corresponds to the field values in the database tables. The Text connector publishes the document types created in the description file. Example: A description file specifies that:

- The text file contains data from one single table (Employees table).
- The first line of the file contains the column headers (each column corresponds to a field in a table).
- The "~" is used to separate values.

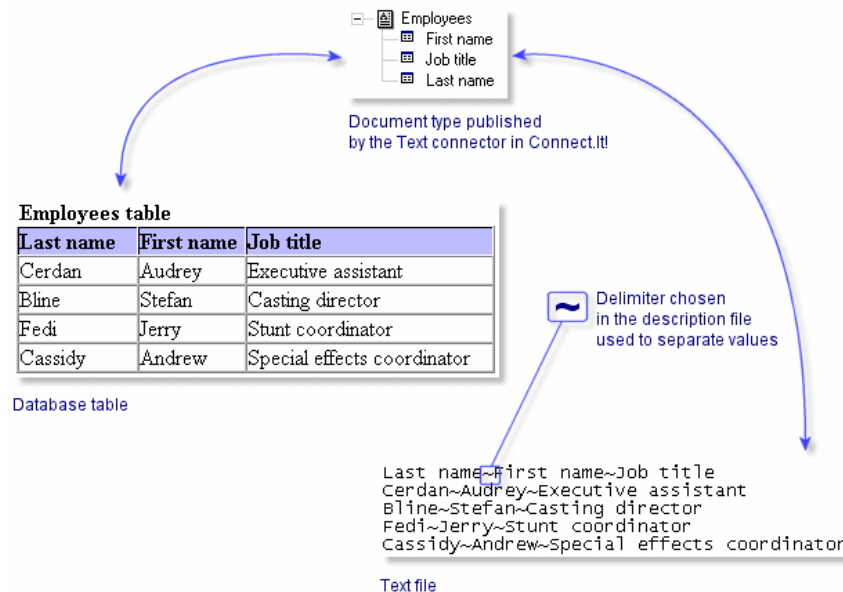


Figure 9-2: Document published by text connector

To access the wizard that enables you to create a description file, click on the **Choose a description file** page in the Configure the connector wizard.

Select a document type

This page helps you create document types published by the Delimited-text connector. Each published document-type corresponds to a database table. To create a document type:

1. Click [✎](#).
2. Click the highlighted text in the Document type column and enter the name of the document type you want to create.
3. Click **Next** to continue on the next page.

Select a file for the preview

This page enables you to indicate a file containing delimited text. The wizard uses this file to let you view how the description file is processing data.

File to preview

Indicate a file containing data delimited by spaces, tabs or special characters. This file must correspond to the files processed by the connector in the scenario.

Tip: To optimize the display of data for previewing, we recommend using as your preview file:

- A local copy of one of the files that the connector will process when you put your scenario into production.
- A small file, although there are no limits pertaining to size.

Number of lines to preview

Indicate the number of lines that the wizard will display in the Data previewing zone.

Specify the column delimiters

This page helps you define how your text file data will be separated. Two options are available:

- **Fixed width**
This option helps you define the fixed width for each of your text-file columns in the **Width** field on the Columns page of the wizard. To create columns, click directly inside the preview zone. To resize a column, click on this column and pull it until you obtained the desired size. To delete a column, click it and drag it outside of the preview zone.
- **Delimiter**
This option helps you define the delimiter that will separate the values in your text file. The available delimiters are:
 - Tab
 - Comma
 - Semi-colon
 - Space
 - List of characters used as delimiters

Indicate the number of characters that you want to use as delimiters. Do not use any delimiters to separate the entered characters.

Specify the data-processing options

This page lets you specify the different ways you can process data from your delimited-text files. The following text delimited by a semi-colon illustrates the different ways data can be processed:

```
Name;Brand;ID
Inspiron;Dell;Comp111
"Inspiron";Dell;Comp112
'Inspiron';Dell;Comp113
"Inspiron" ;Dell ;Comp114
\\"Inspiron";Dell;Comp115
Inspiron\;;Dell;Comp\;116
```

Enter the column names

Select this option to enter the column headers in the produced document.

Import column titles from the first line

Select this option so that the values of the first line of the delimited-text file are used as the column title. The first line used as a column header is also processed in the document type produced.

	Name	Brand	ID	Column 4	Column 5
1	Inspiron	Dell	Comp111		
2	"Inspiron"	Dell	Comp112		
3	'Inspiron'	Dell	Comp113		
4	"Inspiron"	Dell	Comp114		
5	\\"Inspiron"	Dell	Comp115		
6	Inspiron\		Dell	Comp\	116

Figure 9-3: Column titles in a description file

Do not generate errors if a line contains a number of columns different than what is indicated in the description

Select this option to ignore lines that do not match the description of the delimited text. If the line contains a number of columns different than what is indicated in the description, it is ignored and no error is generated.

Quotation character

Indicate which kind of quotation marks to use in your text:

- Single quotation marks (')
- Double quotation marks (")
- All other symbols except for the chosen delimiter(s)

If you choose double quotation marks, you will obtain the following result:

	Name	Brand	ID	Column 4	Column 5
1	Inspiron	Dell	Comp111		
2	Inspiron	Dell	Comp112		
3	'Inspiron'	Dell	Comp113		
4	Inspiron	Dell	Comp114		
5	\\"Inspiron	Dell	Comp115		
6	Inspiron\		Dell	Comp\	116

Figure 9-4: The quotation character in a description file

Pay attention to the quotation conventions for some special data strings. Suppose you have selected **Comma** as the delimiter and **Double quotation marks** as the quotation character:

- If a value contains one or more delimiters, a pair of quotation marks will be around the value.
Example: test1,test2 -> "test1,test2"
- If a value contains one or more quotation characters, a pair of quotation marks will be around the value and the quotation characters within will be doubled. **Example:** test1"test2 -> "test1""test2"

Select the **Keep quotes around values (preview only)** option to have quotation marks that you selected appear in the preview zone.

Commentary starting-line

Indicate the character string signaling commentary in your delimited-text files. The default value is // . In the data preview zone, the commentary is highlighted. If you choose the value \ , for our example text, you will obtain the following result:

	Name	Brand	ID	Column4	Column5
1	Inspiron	Dell	Comp111		
2	"Inspiron"	Dell	Comp112		
3	'Inspiron'	Dell	Comp113		
4	"Inspiron"	Dell	Comp114		
5	\\ "Inspiron" ;	Dell ; Dell ; Dell ;	Comp115		
6	Inspiron\		Dell	Comp\	116

Figure 9-5: Commentary in a description file

Escape character

Enter a escape character. If you choose the character \ for your example text, you will obtain the following result:

	Name	Brand	ID	
1	Inspiron	Dell	Comp111	
2	Inspiron	Dell	Comp112	
3	'Inspiron'	Dell	Comp113	
4	Inspiron	Dell	Comp114	
5	\\ "Inspiron"	; Dell ; Dell ; Dell ;	Comp115	
6	Inspiron ;	Dell	Comp ; 116	

Figure 9-6: The escape character in a description file

Specify the column header and type

The values in each column of a text file correspond to the fields in a database table and, consequently, to the created document-type fields for each description file. This page enables you to:

- Create and name each column of your text files.
- Enter the expected value-type for each created column.
To enter a value type, click the **Type** column and select one of the choices from the drop down

list that appears. The proposed value types are **Text**, **Number**, **Date**, **Date and time**, **Currency**.

- Set the width of each column when the **Fixed width option** is selected on the Delimiters or fixed width page. To enter a value for the width, click the Width column and enter the number of characters required.

Delimited Text-Connector Published Document Types

The delimited text connector publishes the following structure for all document types:

UriFileInfo

This structure is mainly used for post-processing actions. The **UriFileInfo** structure contains the following fields:

- **creationdate**
This field corresponds to the creation date of the document
- **lastmodificationdate**
This field corresponds to the last modification date of the document
- **line**
This field corresponds to the line number of the document
- **path**
This field corresponds to the path of the document

Consumption Directives of the Delimited-Text Connector (write)

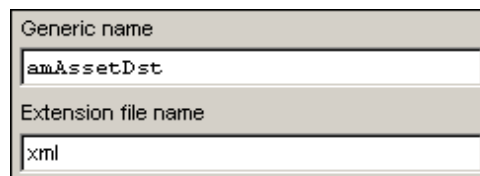
The consumption directives indicate for each document type consumed by the text connector:

- **A generic name**
By default, the value of this field is the name of the document type consumed by the Text connector.
- **A file name extension**
Enter this name without the period mark. Example: Enter **txt** instead of **.txt**; **"txt"** is the default value.

These two fields will only be used if you have chosen the **Write to a different file for each document** option during the configuration of your Text connector.

When you run your scenario, the files that are written will have a composite name, which is created from the generic name, a number corresponding to the file's creation order (**_01**, **_02**, **_03**, etc.) and from the file-name extension indicated. For example, you enter the values **employee** and **txt**. The created files have the following names: **employee_01.txt**, **employee_02.txt**, **employee_03.txt**, etc.

Caution: *If your Text connector consumes several document types, do not use the same generic name twice: The most recently created files will overwrite the previously created files having the same generic name.*



Generic name
amAssetDst
Extension file name
xml

Figure 9-7: Generic name and extension file name**Dynamic population by UriFileInfo.Path**

If the **UriFileInfo.Path** is mapped, this value will be used as the consumption directives. The **UriFileInfo.Path** has high priority to the generic name and file name extension.

The **UriFileInfo.Path** could be used whether you chosen Write the document to a single file or Write to a different file for each document option during the configuration of your Text connector. When you run your scenario, the files that are written will have a dynamic name based on the mapping info of **UriFileInfo.Path**. And more, depend on the option of Behavior between two sessions, a number corresponding to the file's creation order (**_01**, **_02**, **_03**, etc.) could be added to the file name, but not to the file extension.

E-mail Connectors

Connector type: **CONSUMPTION and PRODUCTION**

The e-mail connectors enable you to send and receive e-mails messages. In the case of messages received by the e-mail (fetching) connector, their contents can be routed to an external application after processing in Connect-It (mapping). Example: An **E-mail-ServiceCenter** type scenario enables you to use the information from a message received by your e-mail server in order to create a ticket in ServiceCenter.

When messages are sent by the e-mail (sending) connector, data from an external application can be mapped to the different fields of an e-mail message in Connect-It. For example, in the provided Asset Manager to E-mail scenario (*finreque.scn*), each record in the requests table gives rise to a message sent to the requester's supervisor.

If using Microsoft MAPI or Microsoft Extended MAPI, Microsoft Outlook must be installed on the same machine as Connect-It. There is no requirement for Lotus Notes to be installed on the same computer as Connect-It.

Secure Encrypted Communication Through SSL and TLS

The email connectors now support secure encrypted communication:

- **Email sending connector security options:**
 - SSL for SMTP
 - STARTTLS for SMTP
- **Email fetching connector security options:**
 - SSL for POP3
 - STARTTLS for POP3
 - SSL for IMAP
 - STARTTLS for IMAP

Other Notes

To support multiple language other than English, note the following:

Code pages

- With POP3: make sure that the corresponding language pack for your system is installed.
- With IMAP: check <http://download.oracle.com/javase/6/docs/technotes/guides/intl/encoding.doc.html> to see supported codepages. If your code page belongs to the Extended Encoding Set, such as GB2312, make sure that the international version installed. This version includes the `lib/charsets.jar` file.

Please refer to the Compatibility Matrix available from Hewlett-Packard Development Company, L.P. Support.

Caution: *An English-language UNICODE version is available from Hewlett-Packard Development Company, L.P. Support. Specify the connectors and code pages you need to process in order for them to determine whether the UNICODE version is compatible with your requirements.*

Various limitations

The following list details the different limitations of the e-mail connector:

- The 'Priority' e-mail flag can only be processed with an SMTP server.
- The connector only supports Base64 and Quoted-Printable encoding.
- The **Detect HTML messages** option is not supported with the MAPI protocol.

IMAP4 over SSL

Connect-It's SSL implementation does not support Microsoft NTLM authentication.

Out-of-Box Scenarios

See [Email Connectors Scenarios](#).

Configuring the E-mail connector (fetching)

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Connector type: **CONSUMPTION**

Define the connection parameters

This page enables you to select the messaging protocol of the e-mail (fetching) connector.

The number of fields in this page varies according to the protocol selected in the **Messaging type** field. The protocols available for the e-mail (fetching) connector are:

- POP3 (Post Office Protocol)
- Microsoft MAPI (Messaging Application Program Interface)
- Microsoft Extended MAPI (Extended Messaging Application Program Interface)
- IMAP

POP3

For the POP3 protocol, four fields must be populated:

- **POP3 server**
Enter the name of your POP3 server.
- **Connection security**
Select **None** for no security, or select **SSL** or **STARTTLS** to provide encrypted email communication.
- **Connection port**
Indicate the connection port used by the server on your computer. By default, the value of this field is **110** when no security or when **STARTTLS** is selected, and **995** when **SSL** is selected.
- **Login**
Indicate the login that allows you to access your POP3 server.
- **Password**
Enter the password associated with the login. This password is not visible when being entered and is stored in an encrypted format.
- **Test**
Test the connection to the server.
- **Use DOS/Windows text formatting**
Carriage return (CR) and line return (LF). This option must be selected if the destination application used DOS or 32-bit Windows as operating systems. In these two systems, carriage returns (CR) are followed by a line feed (LF). If the destination application uses UNIX as operating system, this option must be cleared.
- **IPv6**
If the server address is specified by a host name and supports both IPv4 and IPv6, the IPv4 protocol is used by default. Select the **Prefer IPv6** checkbox to use IPv6 protocol.

MAPI

For the MAPI protocol, two fields must be populated:

- **Profile**
Indicate the profile allowing you to access the MAPI messaging installed on your computer.
- **Password**
Enter the user password of your operating system (example: your Windows NT password). This password is not visible when being entered and is stored in an encrypted format.

Note: *Whether a password is required for you to log on depends on the operating system which is running on the computer. If the operating system integrates messaging system credential checking into the initial system logon, you need not provide a password to the MAPILogon function. If not, a password is required from the user or from a cache location implemented by the client. Microsoft Windows NT and Microsoft Windows 95 do not require passwords for MAPILogon since the user's initial logon to the operating system suffices for both. Clients running with Microsoft Windows for Workgroups are required to provide a password for MAPILogon.*

E-mail processing options

When using the MAPI protocol, the following options are available:

- Retrieve unread e-mail messages only
- Use DOS/Windows text formatting: Carriage return (CR) and line return (LF). This option must be selected if the destination application used DOS or 32-bit Windows as operating systems. In these two systems, carriage returns (CR) are followed by a line feed (LF). If the destination application uses UNIX as operating system, this option must be cleared.

Extended MAPI

For the Extended MAPI protocol, the following fields must be populated:

- **Profile**
Indicate the profile allowing you to access the Extended MAPI messaging system installed on your computer.
- **Password**
Enter the user password of your operating system (example: your Windows NT password). This password is not visible when being entered and is stored in an encrypted format.
- **Directory**
Enter the path of your directory. You can enter a path to a local folder or to a folder on the e-mail server. For example, for a folder on the server:
`/Mailbox - Smith E/Inbox`
For example, for a local folder:
`/Personal Folders/Inbox`

E-mail processing options

When using the Extended MAPI protocol, the following options are available:

- Retrieve unread e-mail messages only
- Use DOS/Windows text formatting: Carriage return (CR) and line return (LF). This option must be selected if the destination application uses DOS or 32-bit Windows operating systems. In these two systems, carriage returns (CR) are followed by a line feed (LF). If the destination application uses UNIX, this option must be cleared.

IMAP

For the IMAP protocol, the following fields must be populated:

- **IMAP Server**
Enter the name of your IMAP server.
- **Connection security**
Select **None** for no security, or select **SSL** or **STARTTLS** to provide encrypted email communication.
- **Connection port**
Enter the connection port. If you select **None** or **STARTTLS** for the Connection security, the default port is **143**. If you select **SSL**, the default port is **993**.
- **Login**
Indicate the login that allows you to access your IMAP server.

- **Password**

Enter the password corresponding to your login. This password is not visible when entered and is stored in an encrypted form.

Options for all messaging types

Processing message content

The e-mail connector enables you to parse the text and attachments of e-mails received according to a DTD file. This file describes how the text and attachments of the e-mail are organized in XML elements.

The content of e-mails is analyzed, and all the text contained before the first XML tag (`<?xml version="1.0">`) and the last XML tag is deleted automatically. This analysis allows, for example, to isolate the content added automatically to XML elements by a mail server.

To process message content using a DTD file:

1. Select the **Process contents of e-mail body** option.
2. Indicate the path of the DTD file enabling you to process the body of the e-mail in the **DTD file** field.

To process contents of the attachments using a DTD file:

1. Select the **Process contents of e-mail attachments** option.
2. Indicate the path of the DTD file enabling you to process the attachments of the e-mail in the **DTD file** field.

Define post-processing actions

This page enables you to specify which actions the connector will take after the documents that it produced have been processed or if the processing has failed:

- **Apply actions just after fetching messages**

If select this option, the actions are applied according to the processing reports sent the scenario's other connectors. For more information about processing reports, refer to User's guide, chapter Processing reports.

- **Available post-processing actions**

In the two frames In case of successful processing of message and On failure in processing a file select one of the two following options:

- Do not modify e-mail messages from server
- Flag e-mail messages as read (not POP3)
- Delete e-mail messages from server

Using post-processing actions

To use post-processing actions, you must create a mapping between the **MailInfo.UniqueID** structure of the document type produced by the source connector and the **MailInfo.UniqueID** element of the **SuccessReport** document type.

Document Types Produced by the E-mail (fetching) Connector

The e-mail (fetching) connector receives e-mails. It then produces instances of the **InMailMessage** document type containing the different components of the e-mail message. You can then map this

information to a document type consumed by another connector.

InMailMessage document-type produced by the E-mail (fetching) connector

The following table presents the different components of a document type produced by the e-mail (fetching) connector.

E-mail fetching connector document type components

Part of the produced document-type	Elements
InMailMessage root node	<ul style="list-style-type: none"> • Message body (Body field) • Message date (Date field) • Message priority (Priority field) • Subject of the message (Subject field)
From structure	<ul style="list-style-type: none"> • Electronic address of the author of the message (Address field) • Name of the author of the message (Namefield) • Message author type (Type field)
MailInfo structure	<p>Contains the following field:</p> <p>UniqueID: This field is used by the processing reports sent by the other connectors. This field is the unique identifier of the mail message. It enables you to uniquely identify the messages produced by the connector.</p>
Attachment collection	<p>Contents of attachments represented by the following fields:</p> <ul style="list-style-type: none"> • BlobFromMail field containing the attached data • FileFromMail field containing the file name • MimeType indicating the attachment type
Cc collection	<ul style="list-style-type: none"> • Electronic addresses of the persons to whom the message was sent as a carbon copy (Addressfield) • Names of the persons to whom the message was sent as a carbon copy (Name field) • Carbon copy type (Type field)
To collection	<ul style="list-style-type: none"> • Electronic addresses of the persons to whom the message was sent (Address field) • Names of the persons to whom the message has been addressed (Name field) • Sending type of message recipients (Type field)

Processing contents of messages produced by the e-mail (fetching) connector

When you specify a DTD file for the connector the bodies of messages produced by the e-mail (fetching) connector can be parsed. This process includes new elements in the produced document-type. Each element corresponds to an element in the DTD file.

In the sample E-mail (fetching) to Asset Manager scenario (`newemplo.scn`), the `newemplo.dtd` DTD adds a new structure (`amEmpIDept` structure) to the `InMailMessage` document.

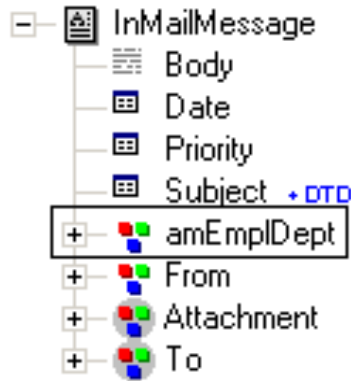


Figure 9-8: Email fetching connector `amEmpIDept` structure

Analysis of the DTD of the E-mail (receiving) connector using MAPI protocol

The Address element of the **From** structure and the **To** and **Cc** collections contain the E-Mail addresses of the sender or the recipient of the message. These addresses may be SMTP addresses, but they can also be internal addresses specific to the E-Mail server when the sender and recipient both belong to the same server.

For example, for a Microsoft Exchange server, the addresses will be SMTP addresses or Microsoft Exchange addresses such as:

- `SMTP:john.doe@company.com` for an external user
- `EX:/O=HP/CN=RECIPIENTS/CN=JDOE` for an internal user

The **SmtAddress** element has been added to the **From** structure and the **To** and **Cc** collections. It contains the SMTP address, regardless of whether the message was sent internally or not. Thus, using the previous example, the corresponding values of the **SmtAddress** element will be:

- `john.doe@company.com` for an external user
- `john.doe@company.com` for an internal user

Configuring the E-mail Connector (sending)

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Connector type: **PRODUCTION**

Configuring the connection

The second page of the wizard enables you to select the messaging protocol of the e-mail (sending) connector. The number of fields in this page varies according to the protocol selected in the

Messaging type field. The protocols available for the e-mail (sending) connector are:

- SMTP (Simple Mail Transfer Protocol)
- Microsoft MAPI (Messaging Application Program Interface)
- Microsoft Extended MAPI (Extended Messaging Application Program Interface)

SMTP

For the SMTP protocol, four fields must be populated:

- **SMTP server**
Enter the name of your SMTP server.
- **Connection security**
Select **None** for no security, or select **SSL** or **STARTTLS** to provide encrypted email communication.
- **Connection port**
Enter the connection port. The default port is **25**.
- **Displayed name**
Specify the name to appear in the **From** field of the person receiving your message.
- **Reply address**
Enter the e-mail address (such as **Reply@hp.com**) that sends the message.
- **Code page to use**
By default, Connect-It uses the ISO-8859-1 code page. You can change the code page depending on your need. For example, if you are in a multi-language environment, use the **UTF-8 page** code page; if you want to use the code page corresponding to the current language in Connect-It, select **automatic**. To switch language, go to **Edit > Options > Advanced options > General > Current language**.
- **Test**
Test the connection to the server.
- **IPv6**
If the server address is specified by a host name and supports both IPv4 and IPv6, the IPv4 protocol is used by default. Select the **Prefer IPv6** checkbox to use IPv6 protocol.

MAPI / Extended MAPI

For the MAPI and Extended MAPI protocols, these two fields must be populated:

- **Profile**
Indicate the login that allows you to access the messaging system installed on your computer.
- **Password**
Enter the password associated with the login. This password is not visible when being entered and is stored in an encrypted format.
- **Request a receipt**
Select this option to receive confirmation that the email message has been read.
- **Detect HTML messages** (Extended MAPI & SMTP)
This option enables you to send messages in HTML format, not Plain Text. Enable this option if

the body of email messages begin with an HTML tags and the message will be sent in HTML format. Otherwise, the message will be sent as Plain Text. If the body of email messages do not begin with an HTML tag, the messages will automatically be sent as Plain Text.

Document Types Produced by the E-mail (sending) Connector

The e-mail (sending) connector sends e-mail messages. It publishes an **OutMailMessage** document type containing the different components of the e-mail message. The information from this document type is mapped to fields in another document type produced by another connector. (The Asset Manager connector in the out-of-the-box Asset Manager to E-mail scenario (`finreque.scn`)).

The following table presents the different components of a document type consumed by the e-mail (sending) connector.

E-mail sending connector document type components

Part of the produced document-type	Elements
OutMailMessage root node	<ul style="list-style-type: none"> • Message body (Body field) • Message priority (Priority field) • Subject of the message (Subject field)
From structure (SMTP protocol only)	<ul style="list-style-type: none"> • Electronic address of the author of the message (Address field) • Name of the author of the message (Name field) • Message author type (Electronic address list for a reply-to address list (Reply-to field))
Attachment collection	<p>Contents of attachments represented by the following fields:</p> <ul style="list-style-type: none"> • BlobFromMail field containing the attached data • FileFromMail field containing the file name
Cc collection	<ul style="list-style-type: none"> • Electronic addresses of the persons to whom the message was sent as a carbon copy (Addressfield) • Names of the persons to whom the message was sent as a carbon copy (Name field) • Carbon copy type (Type field)
To collection	<ul style="list-style-type: none"> • Electronic addresses of the persons to whom the message was sent (Address field) • Names of the persons to whom the message has been addressed (Name field)

Part of the produced document-type	Elements
	<ul style="list-style-type: none"> • Sending type of message recipients (Type field)
Bcc collection	<ul style="list-style-type: none"> • Electronic addresses of the persons to whom the message was sent as a blind carbon copy (Address field) • Names of the persons to whom the message was sent as a blind carbon copy (Name field) • Blind carbon copy type (Type field)

Processing of attachments

Attachments are files (images, video, sound, executables, etc.) sent with a message. In the document types published by the e-mail connector, these files are represented by fields of the **Attachment** collection.

Mapping attachments

This section describes how to map received and sent attachments. Mapping received attachments
Received attachments are represented by three fields in the document type produced by the e-mail connector:

- **BlobFromMail** field containing the attached data.
- **FileFromMail** field containing the file name
- **MimeType** indicating the attachment type

To map the fields of the received attachments, map the **BlobFromMail** field to a binary field of the destination connector.

Example: In the sample E-mail (fetching) to Asset Manager scenario (*newemplo.scn*), the **BlobFromMail** field is mapped to the **Photo.blobData** field in the **amEmpIDept** document type of the Asset Manager connector.

Mapping sent attachments

Sent attachments are represented by two fields in the document type consumed by the e-mail connector:

- **BlobToSend** field containing the attached data.
- **FileToSend** field containing the file name. To map the fields of the attachments received:
 1. Map a binary field of the source connector to the **BlobToSend** field.
 2. Map a text file of the source connector to the **FileToSend** field. This field must contain the name of the attachment.

Example: The **Photo.blobData** and **Photo.Name** fields of the **amAsset** document type of the Asset Manager connector are mapped to the **BlobToSend** and **FileToSend** fields.

HTTP Service Connector

Connector type: **CONSUMPTION and PRODUCTION**

When receiving HTTP requests, the HTTP Service connector enables you to do the following:

- Read documents through mapping, and then expose the consumed data in RESTful format of XML/ATOM/JSON.
- Convert HTTP requests to produced document type for the mapping engine to decide what operations to perform, such as creating, updating and deleting documents.

How it works

- To read data from a source connector
 The listening port of the HTTP Service connector receives an incoming HTTP request of which data format depends on the document type defined by XML schema. The connector processes the request to drive a connector (for example, the Database connector) to produce data through mapping logic. The processed data produced by the Database connector goes back to the HTTP Service connector, and then the result data can be exposed in RESTful format as response of the HTTP request.

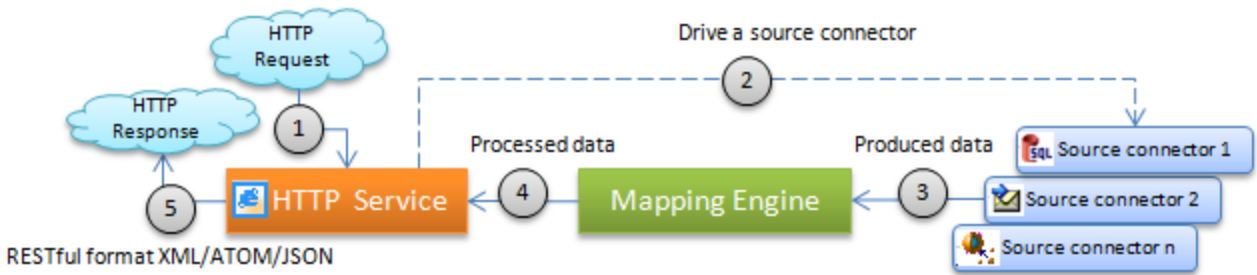


Figure 9-9: Reading data from a source connector

- To transfer data from a source connector to a destination connector
 In this situation, the Status Reports connector is used to trace reports to indicate whether reading data from a source connector is successfully finished or not. The process starts with the HTTP Service connector receiving an HTTP request and driving a source connector (for example, the Database connector) to produce documents. Then a destination connector consumes these documents through mapping logic. The Status Report connector monitors the reading process and collects all the status data that is sent to the HTTP Service connector as the response of the HTTP request.

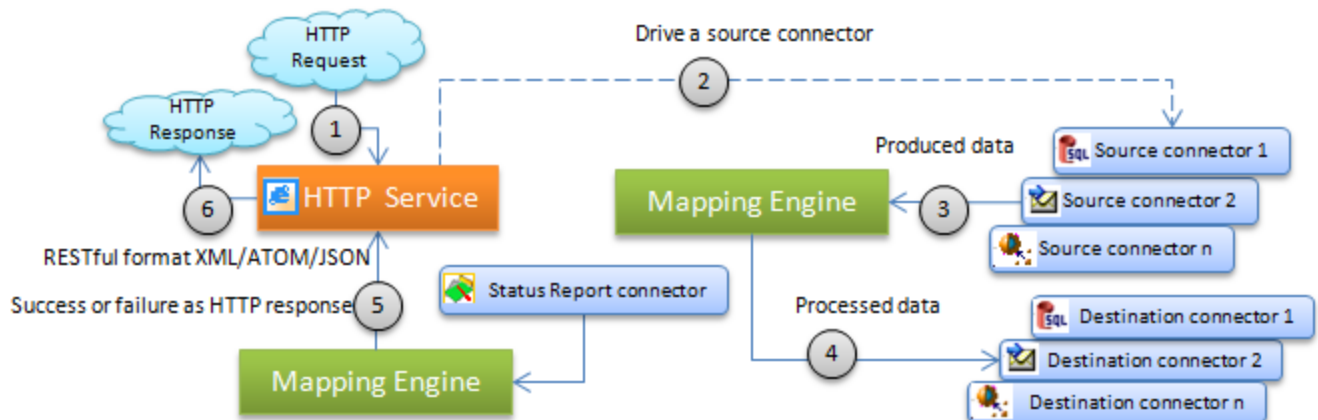


Figure 9-10: Transferring data from source to destination

- To write data to a destination connector

The HTTP Service connector receives an HTTP request with writing operations (PUT/POST/DELETE) and sends it to the mapping engine. A destination connector consumes these documents through the mapping logic, and its processing reports returns consumption feedback to the HTTP Service connector via mapping. Response is then exposed as data of RESTful format.

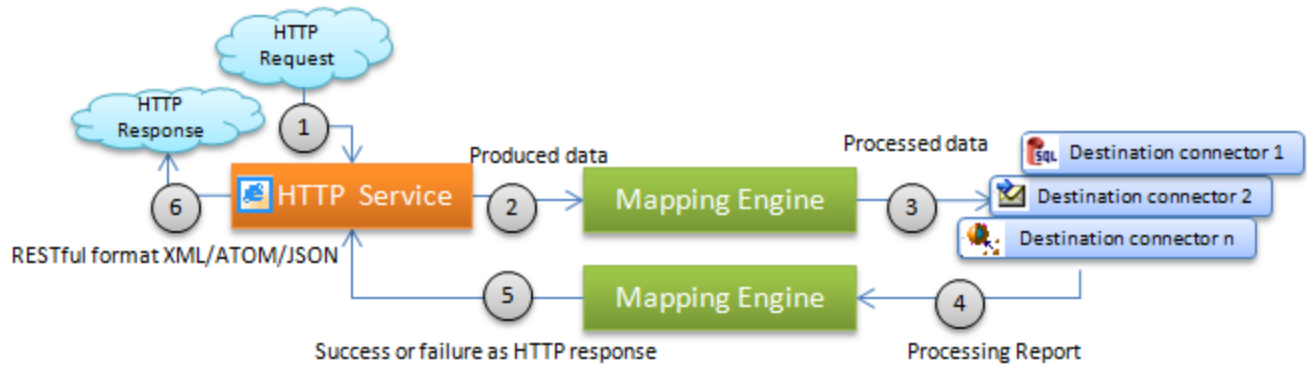


Figure 9-11: Writing data to a destination connector

Known limitations of the connector

- Concurrency is not supported. The HTTP Service connector is not able to accept a HTTP request until the connector finishes returning the response of the previous request.
- The HTTP Service connector cannot read or write binary data.
- When reading data from a source connector, the HTTP Service connector doesn't support pagination query.
- When using WHERE Clause to query a record in a source connector, you must follow the format: `http://{server:port}/cit-rest/service/{servicename}/{dataId}`

Out-of-Box Scenarios

See [HTTP Service Connector Scenarios](#).

Configuring the HTTP Service Connector

The HTTP Service Connector requires both the wizard configuration and layer configuration.

Wizard configuration

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

- **Server Port**

Specify the listening port used by the HTTP Service connector. The port enables a HTTP server to create the communication channel for processing HTTP requests.

- **Schema Location**

Provide the full path of the HTTP Service connector XML schema that defines document types.

- **Extended Schema Location**

This is an optional field. Specify the Extended XML schema path that can be used to define an extended structure for every document type of the HTTP Service connector.

For example, if you want to return an error message in the HTTP Service response for writing operation, you can define a response structure, and then the Process report fills it with data. The result will be put in every document type.

Layer configuration

The layer configuration provides you with configuration options to be used for the HTTP Service connector in the edit window of produced or consumed document types. You can specify the source connector name and its document type when you want to read data from a source connector, or configure supported HTTP methods such as HTTP GET/POST/DELETE when you want to insert, update or delete a document.

The layer configuration frame consists of the RESTful exposure tab and the Service configuration tab. RESTful Exposure exposes data managed by a connector directly as XML resources through HTTP protocol using a RESTful design. You may find more details about [RESTful exposure](#) in this guide. The configuration steps in the Service configuration tab vary depending on the use of the connector.

Reading Data

When using the HTTP Service connector as a destination connector to read data, the connector is the one to consume documents and you can see the following configuration fields:

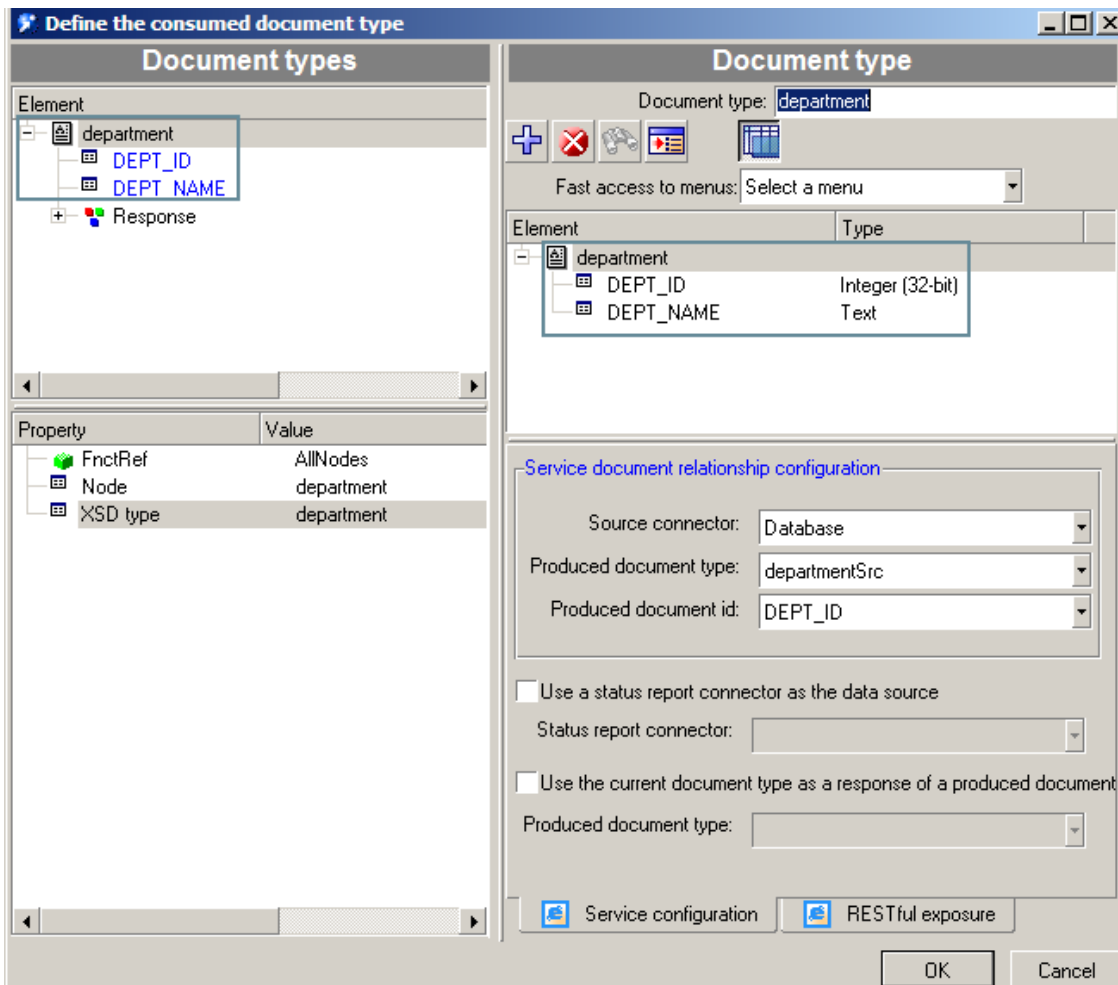


Figure 9-12: Layer configuration for reading data

- **Document type**
Name a service, for example **department**.
- **Source connector**
All the connectors that produce documents in the scenario are listed in the drop-down list. Select the one from which you want to read data.
- **Produced document type**
Select a produced document type in the source connector. This produced document type is used when the source connector is triggered to start producing documents.
- **Produced document type id**
Select a field as the query condition when the source connector supports WHERE clause. For example, if you want to use the URL <http://localhost:8080/cit-rest/service/department/10001> to query a record, **10001** will be used as the condition id. The source connector uses this value as the "Produced document type id" in the WHERE clause.
- **Use a Status Report connector as the data source**
Select this check box when the Status Report connector is used for operation results.
- **Status Report connector**
Select a Status Report connector if you need a status report as the response of data reading.

- **Use the current document type as a response of a produced document type**
This field is only for writing data through the HTTP Service connector. Find the related information in the Writing Data section.
- **Produced document type**
This field is only for writing data through the HTTP Service connector. Find the related information in the Writing Data section.

Writing Data

As a source connector, the HTTP Service connector can convert a HTTP request to a produced document, and then enable the destination connector to perform writing operations through mapping reconciliation. You need layer configuration for both the produced document type and consumed document type.

To define the produced document type:

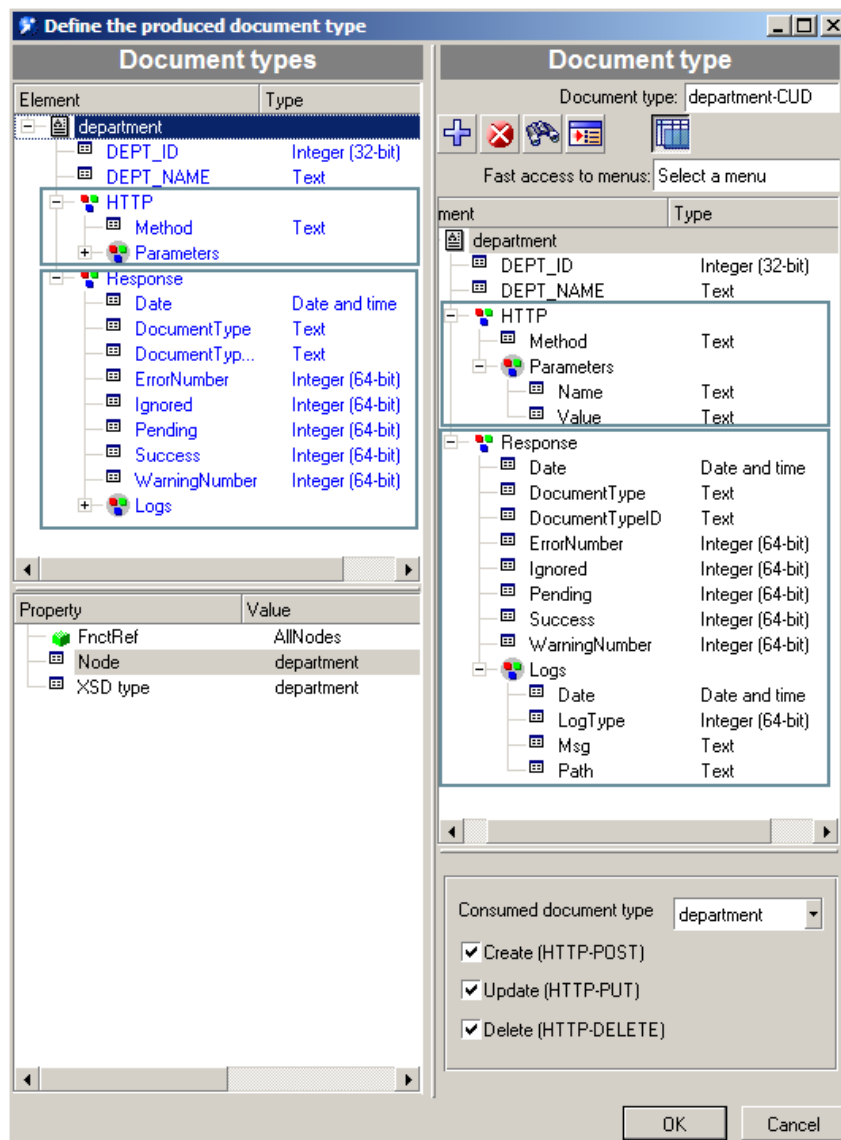


Figure 9-13: Defining the produced document type

- Consumed document type**
 Select a consumed document type for the destination connector.
- Create (HTTP-POST)**
 Select the check box to support the HTTP POST method for data inserting.
- Update (HTTP-PUT)**
 Select the check box to support the HTTP PUT method for data updating.
- Delete (HTTP-DELETE)**
 Select the check box to support the HTTP DELETE method for data deleting.
- HTTP structure**
 This structure is filled with values automatically, and it must be added to the right panel. Add the HTTP structure to the selected document type so that the HTTP method and parameter information can be used in the Basic language script in the mapping box for reconciliation operations.
- Response structure**
 If you want a returned response for writing operations, you must add the Response structure to the selected document type. It is used to receive the Process Report data that indicates whether writing operations are finished successfully or not. This field is defined by [Extended Schema Location](#) configured in wizard.

To define the consumed document type:

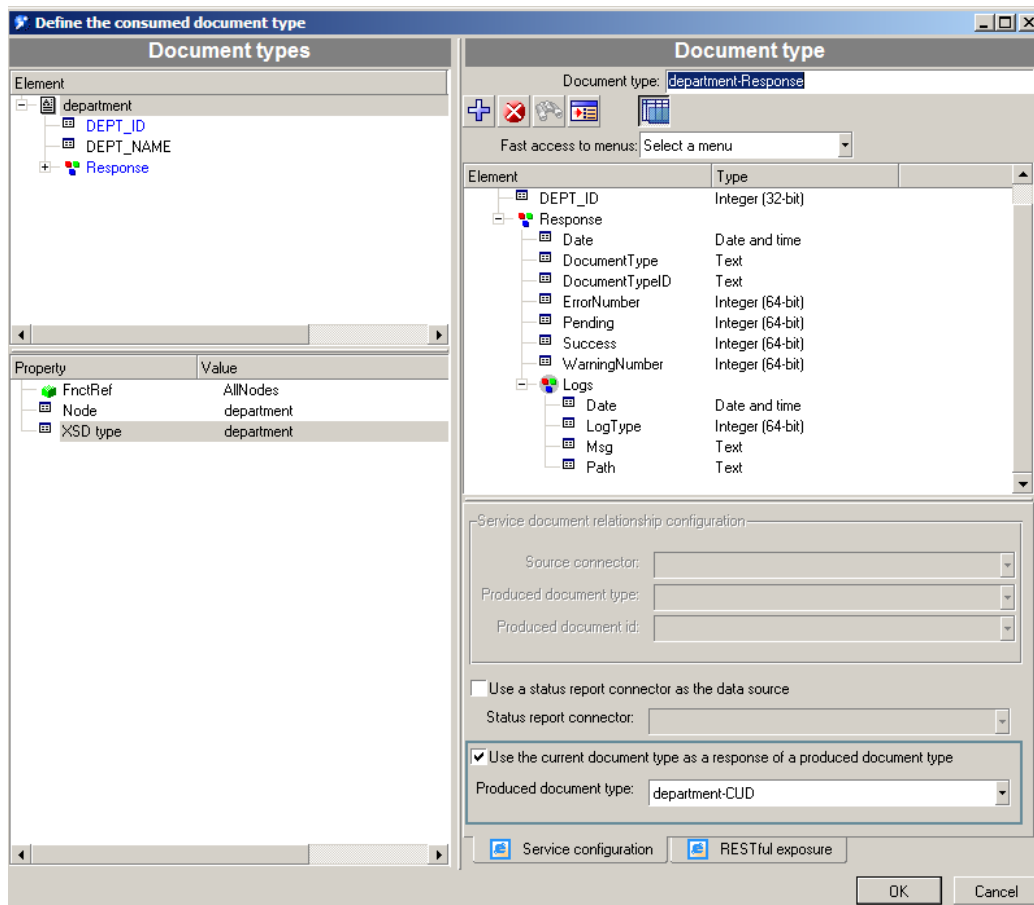


Figure 9-14: Defining the consumed document type

- **Use the current document type as a response of a produced document type**
This check box must be selected if the current document type is used as response.
- **Produced document type**
It's a mandatory field. It can convert HTTP requests with method (POST/PUT/DELETE) to a data record. In this example, the field can automatically identify the current document type "department-Response" as the response structure for "department-CUD".

JMS Connector

Connector type: **CONSUMPTION and PRODUCTION**

Java Message Service describes the API and required semantics which enable a Java client to use an asynchronous message bus. The following communication modes are supported by the connector:

- **Point to point (queues)**
The message is consumed by a single client.
- **Publication / subscription (distribution lists)**
All clients that have subscribed to the distribution list consume the message.

The JMS connector enables XML documents to be exchanged via JMS. The connector is used to define the following behavior:

- Interaction with the JNDI provider: Connection address, root class that enables access to ConnectionFactory objects, Queue, Topic
- Definition of the connection with the JMS server
- Interaction with a proxy server

Out-of-Box Scenarios

There is no out-of-the-box scenario for this connector.

Configuring the JMS Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Configure the connection to the JNDI provider

The JNDI (Java Naming and Directory Interface) extension is a Java standard that enables a client to access a directory. Information for the connection to the JMS server is saved in a directory. Access to this directory requires information about the JNDI provider.

- **Class:**
Enter the full name of the JNDI provider's initial context factory implementation class (**javax.naming.spi.InitialContextFactory**). For example, for the JBoss JNDI provider:
`org.jnp.interfaces.NamingContextFactory`
This class must be present in the connector's classpath. You must add the JNDI client's Java libraries (`jar`) that correspond to the JNDI provider to the connector's classpath.
- **URL**
Enter the URL of the JNDI directory. The URL format depends on your JNDI provider. For

example, for the JBoss JNDI provider, the URL has the following format:

```
jnp://myserver:1099
```

- You can enter additional properties in the **Name** and **Value** fields to create the initial JNDI context. The name and the manner in which these additional properties are interpreted depend on the JNDI provider.

Configure the connection to the JMS server

The connection to a JMS server is made using the JMS connection factory (`javax.jms.ConnectionFactory`). A connection factory must be provided by the JMS administrator and made available in the JNDI directory. The implementation classes of the JMS client must be present in the connector's classpath. You must add the JMS client's Java libraries (jar) that correspond to the JMS provider to the connector's classpath ([Configure the JVM](#) wizard page).

This page enables you to configure the connection to the JMS server. The following fields must be populated:

- **ConnectionFactory:** Name of the JNDI context containing the JMS connection factory.
- **Destination:** Enter the JMS destination (queue or distribution list) to read or send XML documents.

Define post-processing actions

After a file is read by the JMS connector, Connect-It proposes the following options, which depend on processing success or failure:

- Do nothing
- Copy the message to another destination Consumption of a JMS message destroys the message. Once consumed, the message is no longer delivered (and will be destroyed if the destination is a queue). If the **Copy the message to another destination** option is selected, a copy of the message will be sent to the selected destination. The value of the correlation identifier (**JMSCorrelationId**) for this destination will be set to the value of the originating message's identifier (**JMSMessageID**). This function is linked to the use of processing reports.

Specify the XML schema of the messages

Messages are transmitted in `.xml` format. This page lets you define the URL of the XML schema that is used to process the messages.

Configure the JVM

This page lets you define all options and classpaths required by the JVM. On this page, enter all options and classpaths that are used for the client. The implementation classes of the JNDI and JMS clients must be specified in the **Classpath** frame.

Type of published documents

- In read mode:
Documents that correspond to the XML schema plus a `JMSMessageInfo` structure that contains information about the JMS message.
Directives: Message filter: JMS filter used to filter received messages depending on the properties of the JMS messages.
- In write mode:
A document type produced on `JMSMessageInfo` consumption containing information about the message that is sent.

Type of consumed documents

- In read mode:
Nothing
- In write mode:
Documents corresponding to the XML schema plus a `JMSMessageInfo` structure used to specify information about the JMS message.
- Directives:
 - **Codepage:** UTF-8 by default
 - **Priority:** Priority of the JMS message that is sent
 - **Life cycle:** Life cycle of the JMS message that is sent (0=unlimited)
 - **Persistence:** Send or do not send a persistent JMS message

Note: *Connect-It specified JVM options are applicable to this connector. See the [Configuring the JVM](#) topic for more information.*

LDAP Connector

Connector type: **CONSUMPTION and PRODUCTION**

The LDAP connector enables you to process data from servers that use the LDAP protocol. It enables you, for example, to access the X500 electronic directories. The actual version of the LDAP connector enables you to read and write the data coming from an LDAP source.

The LDAP connector uses the auto-description capacities of the LDAP v3 protocol and furnishes vast information to the user: Definitions of object classes, naming contexts, supported controls, etc.

Behavior specific to Microsoft Active Directory

Writing an empty string generates an error on a Microsoft Active Directory server.

Precautions of use for the LDAP connector

Limitations of the LDAP connector depend on the limitations of your LDAP resources (CPU allotted, memory allotted, etc.). If you pass the limits of your LDAP resources while testing your connector or running a scenario, it is possible that the data processing failed partially or completely.

Customizing operational attributes

For a Red Hat Directory LDAP server, a certain number of operational attributes are used to save information related to security or named users. The attributes used by the LDAP connector are saved in the `config/ldap/ldapball.cfg` file.

For example:

```
{ STRUCT VIRTUAL
{ STRUCT AllTables
TABLE = self
TIMESTAMP modifyTimestamp
TIMESTAMP createTimestamp
{ STRING distinguishedName
INSERTONLY = 1
KEY = 4
}
{ ATTRIBUTE AllFields
FIELD = self
Exception = modifyTimestamp, createTimestamp, distinguishedName
```

```
CIRCULAR = AllTables
}
}
```

This file can be edited and configured to add a list of attributes. For example, to add the nsaccountlock attribute for a Novell Directory server:

```
{ STRUCT VIRTUAL
{ STRUCT My_Dummy_Object
ObjectClass = *
STRING nsaccountlock
}
{ STRUCT AllTables
TABLE = self
TIMESTAMP modifyTimestamp
TIMESTAMP createTimestamp
{ STRING distinguishedName
INSERTONLY = 1
KEY = 4
}
{ ATTRIBUTE AllFields
FIELD = self
Exception = modifyTimestamp, createTimestamp, distinguishedName
CIRCULAR = AllTables
}
}
```

These attributes must be prefixed with an asterisk (*) so that they are not taken into account during an LDAP search.

Out-of-Box Scenarios

See [LDAP Connector Scenarios](#).

Configuring the LDAP Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Certain options concerning this connector can be enabled using the **Edit > Options > Connector** menu.

LDAP server name

Indicate the name of your LDAP server. You can either enter the server name (for example: ldap-server.unil.ch) or its numeric IP address (for example: 207.68.137.42).

Server connection port

Indicate the connection port of your LDAP server. The default value of this field is **389** and should work in most cases. (This value is proposed by default by the protocol.)

Login

Indicate the login that enables you to access your LDAP server. When you select the Windows Active Directory option, this login must use the following syntax: full name@domain.name. For

example, john.doe@hp.com. The LDAP connector is restricted to users who have permissions to retrieve a data schema from LDAP. For example, if the login allows the user to view the **Schema** tab in the JXplorer browser, that login can be used to configure the LDAP connector.

Note: For iPlanet LDAP server, the login is the Distinguished Name of the user.

Password

Enter the password associated with the login. This password is not visible when being entered and is stored in an encrypted format.

Indicate the server type

The available server types are:

- Novell Directory Service, Netscape Directory Server, etc.
- Microsoft Exchange
- IBM z/OS Security Server
- Microsoft Active Directory
- OpenLDAP

Note: If Resource Access Control Facility (RACF) is installed on your IBM z/OS Security Server, the RACF-related table names must start with the characters **racf**. For example, **racfuser**, **racfgroup**, or **racfconnect**.

SSL connection

Select this option if you use a secure connection protocol. According to the operating system used, proceed as follows:

- **Solaris:** for all versions prior to Solaris 9, you must apply patch 112438 for Solaris 8. This patch is available at the following site: sunsolve.sun.com. Corrections are available for Solaris versions 2.6 and 2.7.
- **AIX:** for all versions prior to AIX 5.2, you must download the **prngd** package available at the IBM Web site (<http://www.ibm.com/servers/aix/products/aixos/linux/download.html>), as the peripheral files **/dev/random** and **/dev/urandom** are not installed by default.

Referrals

Select this option if you would like to track the "LDAP referrals" sent by the LDAP server

Specify the codepage of the server

The drop-down list enables you to indicate the codepage of your LDAP connector.

Advanced configuration (advanced mode)

This page enables you to specify:

- The size of the page
The default value is 500. The value of this option is only used by the Microsoft LDAP servers.
- The DN (Distinguished Name) of the schema
In rare cases, a schema's distinguished name is not automatically recovered by the LDAP connector. In this case, you must specify it in this field. A warning in the Connect-It log will tell you if the distinguished name was not recovered. To obtain this name, refer to the subschemaSubentry entry of the DSE root (Directory Service Entries).

- The format of the dates.
The default value of this field is `%4Y%2M%2D%2H%2N%2SZ`. The following symbols are those specified for the date and time:
 - Y
Year
 - M
Month
 - D
Day
 - H
Hour
 - N
Minute
 - S
Second
 - Z
Mandatory symbol specifying that the date is in GMT 0 format.

The `%[number][symbol]` string specifies how many digits are used to express the indicated symbol. For example: `%4Y` indicates that the year must be expressed in four digits: 2003, 1997, etc.

If your server supports milliseconds, these are expressed by a decimal point. The number after the point expresses how many digits follow the decimal point. Example:
`%4Y%2M%2D%2H%2N%2.1SZ` indicates that milliseconds can be expressed after the decimal point.

Current date of the server

- The **Attribute Name** field is used to enter the attribute that has the current date of the server.
- The **DN** field is used to enter the attribute's identifier (Distinguished Name). The default entry is **currentTime**.

Configure schedule pointers (advanced mode)

Last modification field

Certain document types have a field indicating the last modification date of a record in the database schema. In most cases, the **Modified-date** field is used. However, depending on your database schemas, the name of this field may differ. The editable zone in this field enables you to indicate the field of the DSE used as the schedule pointer for the LDAP connector. The default value of this field is **modifyTimestamp**.

LDAP Connector Consumption and Production Directives

For information about how to enter a connector's consumption directives, refer to [Defining consumption directives](#). A connector's consumption directives involve:

- Entering reconciliation parameters in the **Reconciliation** and **Advanced reconciliation** tabs.
- Writing SCOPE and SEARCH DN clauses.

SCOPE clause

This clause determines the scope of your request depending on your entry point, identified by a DN (**Distinguished Name**) in an LDAP directory tree. Three options are available:

- **Base**
By selecting this option, your request applies to the data contained under your entry point without exploring the sub-trees of this entry point.
- **First level**
By selecting this option, your request applies to the data contained under your entry point, as well as under the direct sub-nodes of this entry point.
- **Recursive**
By selecting this option, your request applies to the data contained under your entry point and under all the sub-nodes of this entry point.

SEARCH DN clause

This clause enables you to indicate the DN (**Distinguished Name**) of your entry point in the tree structure of the directory to which you connected your LDAP server. When you launch your connector, the directory proposes several entry points. These are the naming contexts exposed by the LDAP connector. These naming contexts appear in the drop-down list next to the **SEARCH DN** clause. You can, however, choose another entry point in the directory by editing this field manually. Each DN is made up of RDNs (Relative Distinguished Names), a list of the most frequent is given below.

Relative Distinguished Names used by LDAP

RDN (Relative Distinguished Name)	Key
CommonName	CN
LocalityName	L
StateOrProvinceName	ST
OrganizationName	O
OrganizationalUnitName	OU
CountryName	C
StreetAddress	STREET

*Note: Do not leave the **SEARCH DN clause** field empty. This would mean querying the entire directory and would probably overrun the LDAP server's download limitations.*

For Production Directives

In addition to the SCOPE and SEARCH DN clauses mentioned above:

FILTER clause

This clause enables you to filter records under the entry point that you selected. The **FILTER** clause must respect the syntax used by the LDAP requests. This syntax is presented in the RFC 2254. Example: If you want to obtain the list of all people whose common name starts with 'A', you must enter the following clause: (&(ObjectClass=person)(cn=A*))

If you only want to obtain the list of people beginning with 'Ar', you must enter the following clause:
 (&(ObjectClass=person) (&(cn=A*) (cn=Ar*)))

To verify that your LDAP server manages the dates that entries were modified in your directory:

- Create an LDAP document type.
- Enter the following value for the FILTER clause:
 (modifyTimestamp=*)

This returns you all the entries in the directory whose **modifyTimestamp** field is populated. If this field is not populated for all the entries, then the connector needs to be configured to recover all entries at each startup.

LDAP filter syntax

The following presents the syntax used to filter the LDAP data:

LDAP filters

filter= "(" filtercomp ")"
filtercomp = and / or / not / item
and = "&" filterlist
or = " " filterlist
not = "!" filter
filterlist = 1*filter
item = simple / present /substring / extensible
simple = attr filtertype value
filtertype = equal / approx / greater / less
equal = "="
approx = "~="
greater = ">="
less = "<="
extensible = attr [":dn"] [": matchingrule] ":@" value / [":dn"] [": matchingrule ":@" value
present = attr "=*"
substring = attr "=" [initial] any [final]
initial = value any = "*" *(value "**")
final = value
attr = AttributeDescription of section 4.1.5 of the RFC 2251
matchingrule = MatchingRuleId of section 4.1.9 of the RFC 2251
value = AttributeValue of section 4.1.6 of the RFC 2251

The following table indicates how to obtain certain characters in the values of the documents processed by the LDAP connector.

Special characters processed by the LDAP connector

Desired character	ASCII value
*	0x2a
(0x28
)	0x29
\	0x5c
NULL	0x00

The character must be encoded like the "\" character (ASCII 0x5c), followed by two numbers representing the ASCII value of the encoded character. The case of the two hexadecimal characters is not important. Example: To verify that the cn RDN contains the * character, write the following filter:

"(cn=*\2a*)"

Examples of filters

Example LDAP filters

Objective of the filter	To write
Look through all classes.	(objectclass=*)
Filter the people whose name starts with A .	(&(objectclass=person)(cn=A*))
Filter all types of people in LDAP.	((objectclass=person)(objectclass=organizationalPerson)(objectclass=inetOrgPerson)(objectclass=residentialPerson)(objectclass=newPilotPerson))
Filter the people whose name starts with A while ignoring those whose name starts with Ar .	(&(objectclass=person)(&(cn=A*)(!(cn=Ar*))))
To filter people whose name does not start with H, Y or E, you cannot write:	(&(objectclass=person)(!(cn=H*)(cn=Y*)(cn=E*))) "not" (!) is an unary operator.
To filter the people whose name does not start with H, Y or E, you must write:	(&(objectclass=person)(&(!(cn=H*)(!(cn=Y*)(!(cn=E*))))))

Additional Information for the LDAP Connector

This section presents additional information about the connector.

Sources of information concerning the LDAP protocol

For more information about the LDAP protocol, refer to the following RFCs (Requests for comments):

- RFC 1274: The COSINE and Internet X.500 Schema
- RFC 1777: Lightweight Directory Access Protocol
- RFC 1778: The String Representation of Standard Attribute Syntaxes
- RFC 1617: Naming and Structuring Guidelines for X.500 Directory Pilots
- RFC 2253: Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names
- RFC 2251: Lightweight Directory Access Protocol (v3)
- RFC 2252: Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions
- RFC 2254: The String Representation of LDAP Search Filters
- RFC 2256: A Summary of the X.500(96) User Schema for use with LDAP v3
- RFC 1823: The LDAP Application Program Interface
- RFC 1798: Connection-less Lightweight X.500 Directory Access Protocol
- RFC 2259: Internet X.509 Public Key Infrastructure Operational Protocols - LDAP v2
- RFC 2279: UTF-8, a transformation format of ISO 10646
- RFC 2116: X.500 Implementations Catalog-96
- RFC 2255: The LDAP URL Format

Mapping the sole field of a collection to a field in another application

LDAP-directory entries are often single-field collections (each member of the collection corresponds to a field). Each attribute represents an information type containing one or more values (multiple-value attribute). Example: A collection for which the value of the attributes is an e-mail address. Ideally, you would map this field to the field of a collection in the destination application. If there are no available collections, you have two solutions:

1. Map one single member of the collection to the field of the other application.
 2. Use a Basic script in the mapping. This script is used to concatenate the values of all the fields in a collection. The concatenated value obtained can then be mapped to the single field in the external application.
- Map one single member of the collection to the target field:
 1. In the **mapping edit** window, map the field of the collection of the LDAP-document type to the target field.
 2. Enter the number of the field that you want to map to this target field in the Mapping script field. (Reminder: Collection members in Connect-It are numbered sequentially from 0; 0 being the first member, 1 the second, etc.) For example: To indicate the first attribute of a single-field collection, Person . cn (common name of a person), enter the following line in the mapping script: [cn (0) . cn].
 - Using a Basic script
 1. In the mapping edit window, map the sole field of the collection of the LDAP-document type to the target field.

2. Enter the mapping script in the **Mapping script** field.

The first part of the script counts the number of members in the collection whereas the second part concatenates them in order to produce a single string. The syntax is as follows:

```
Dim iCollectionCount As Integer
iCollectionCount = PifGetItemCount("cn")
Dim strCollapse As String
Dim iItem As Integer
For iItem = 0 to iCollectionCount - 1
  strCollapse = strCollapse + [cn(iItem)]
Next iItem
RetVal = strCollapse
```

Identifying the LDAP elements containing the most data


The LDAP connector publishes document types containing the following information:

- Object classes
- Fields of these classes
- Heritage between these classes

The most important information when using an LDAP data source is:

- The object classes containing the data that interests you. For example, the tables containing records.
- The heritages between the classes enable you to write high-performance requests. For example, the **Top**, **Person** or **OrganizationalPerson** class.

To obtain this information:

1. Select the LDAP connector in the scenario that you created or edited.
2. Select the **Document types** tab.
3. Create a produced document-type whose root element is **Top**. All the LDAP directories publish a **Top** object class that serves as a root element for all the server classes.
4. Click  to view the data in the LDAP source.
5. Search the classes returning the most data in order to use this data in the definition of your document types.

LDAP date formats

The format of **timestamp** fields for the LDAP directories respect the following syntax:

```
[year] [month] [day] [hour] [minute] [second] Z
```

Example: 22 hours 40 minutes 34 seconds, April 5, 2003 is expressed by the following string:
20030405224034Z.

The final **Z** indicates that the date is on the GMT 0 (Greenwich Meridian Time) time zone. Certain LDAP servers use other formats for timestamps. Example: Certain servers add a digit corresponding to milliseconds (this digit is expressed by a decimal point.) Example:
20030405224034.5Z.

Web Service Connectors

Connector type: **CONSUMPTION and PRODUCTION**

Web Service connectors enable you to interact with Web services. A Web service provides a service or a set of services composed of operations. SOAP, MTOM, HTTP GET/POST and MIME are the main communication protocols used by these Web services.

Lists of Web services are available on numerous sites. For example: <http://www.xmethods.com>.

In an integration scenario, a connector consumes a document that it sends as a request to a Web service. Then it receives a response that it automatically transforms into a produced document type. On receiving a document, this connector sends a query to the Web service and produces a document on receiving the response from the Web service.

The group of Web Service connectors consists of the following:

- RESTful (Deprecated) Client Connector
- RESTful Client Connector
- SOAP Connector

RESTful (Deprecated) Client Connector

The RESTful (Deprecated) Client connector:

- Is part of the base connectors
- Acts as a REST client to issue HTTP requests (POST, PUT, DELETE, and GET)
- Is based on Atom Publishing Protocol (APP)
- Uses data that is exchanged between CIT and external systems is XML based.
- Is compliant with the Hewlett-Packard products that provide REST Web service endpoints based on Atom protocol/XML data.

Out-of-Box Scenarios

There is no out-of-the-box scenario for this connector.

Configuring the RESTful (Deprecated) Client Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Specify the XMLSchema for the resources

URL of the schema: XML Schema containing the description for all the resources to be retrieved/added/updated/deleted from the feed. Choose the representation for the resources: Either Atom entry: The document types that will be exposed by the connector are the Atom entry documents. The Atom entry's content will correspond to the XML representation (given by the XML Schema) of the resource itself. or XML content: The document types that will be exposed by the connector correspond directly to the XML representation of a resource.

Consumption directives

HTTP method to apply: choice between POST, PUT or DELETE to select the HTTP method that will be performed Name of the field used as the ID of the resource. If configured to expose directly XML content, this directive lets you to specify what field will be used to represent the Atom entry identifier. It is mandatory to provide a name if issuing a PUT or a DELETE because it has to know what resource is to be updated or deleted.

Production directives

To obtain a specific subset of the entire feed, RESTful web service endpoint may support additional query parameters to pass on the HTTP GET request URL. The production directive consists in an optional array of parameter name/value pairs that forms the query part of the URL.

Schedule pointer support

When a scenario is launched in scheduled mode, the RESTful (Deprecated) client connector is able to limit resources production to only those that have been created or updated since the last session. To do so, it relies on a specific query parameter called 'watermark' of type datetime that must be supported/understand by the endpoint.

RESTful Client Connector

Connector type: **CONSUMPTION and PRODUCTION**

The RESTful Client connector acts as a REST client to issue HTTP requests and accepts HTTP responses. Compared with the RESTful (Deprecated) Client connector, the RESTful Client connector is more powerful, flexible, and featured by the following capabilities:

- Allows users to define the Path attribute and set it as a placeholder in a server URL.
- Supports the MIME multipart type.
- Provides suggested HTTP request and response schema (XSD) files.
- Provides options to send compressed HTTP requests and accept compressed HTTP responses.

Out-of-Box Scenarios

See [RESTful Client Connectors scenarios](#).

Configuring the RESTful Client Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Specify the server information

- **Server**

Specify the URL of the server using the format `[hostname]:[port]/[others]`, for example, `abc.hp.com:8080`. It's recommended not to add `http` or `https` at the beginning of the URL, because Connect-It can tell the protocol to be used based on the configuration information automatically.

You can also use a placeholder [Path] anywhere in the URL to provide a flexible server address. The RESTful Client connector contains a Path attribute in the document type (Text, File or Multipart), and the attribute value corresponds to the placeholder [Path] in the server URL. See the table below for the examples using the placeholder:

Suppose the complete URL is `abc.hp.com:8080/rest/user?eid=1000`

Num.	Server URL	The value for Path attribute	Description
A	abc.hp.com:8080	/rest/user?eid=1000	A commonly used example. The "/rest/user?eid=1000" part can be changed for different requests.
B	abc.hp.com:8080/rest	/user?eid=1000	A commonly used example. The "/user?eid=1000" part can be changed for different requests. Compare with the example A, the "/rest" part is fixed in the URL and unchangeable.
C	abc.hp.com:8080/[Path]	rest/user?eid=1000	The result is the same as the example A.
D	[Path].hp.com:8080/rest/user?eid=1000	abc	A special example. The "abc" part can be changed for different requests.
E	[Path]	abc.hp.com:8080/rest/user?eid=1000	A special example. The entire URL can be changed for different requests.

- **Login**

Enter the user name to log on to the server.

- **Password**

Enter the password to log on to the server.

Note: The RESTful Client connector uses HTTP basic authorization.

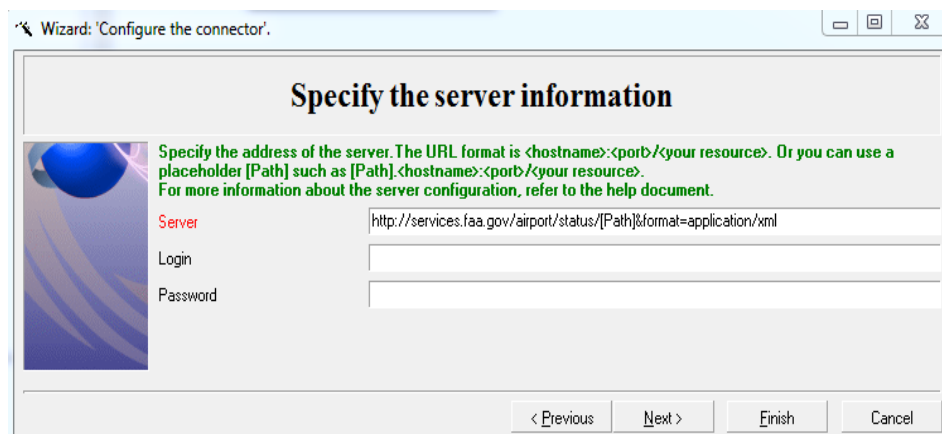


Figure 9-15: Specify the server information**Specify document types****• Learning XML to generate a suggested XML schema**

Select the check box if you want to generate an XSD file to be used as the HTTP request or response schema file, and then provide the directories for the following files:

- **Suggested HTTP request XML schema**
- **Suggested HTTP response XML schema**

Note: These are suggested schema files and you may need to validate files before using them.

• Choose the MIME type

Select one or more options:

■ Text

Subtypes include text/plain, text/html, application/xhtml+xml, etc.

■ File

Subtypes include image, audio, video and application files.

■ Multipart

The MIME standard defines various multipart-message subtypes to specify the nature of the message parts and their relationships. The subtype is specified in the Content-Type header of the entire message.

Note: The Multipart type is only for a HTTP or HTTPS request message.

• Use an XML Schema (XSD) to define the document types

Select the option if you want to use an existing XML schema file for document types. You can select only one from the **Learning XML to generate a suggested XML schema** and **Use an XML Schema (XSD) to define the document types** check boxes.

After selecting to define an XSD file, specify the directories:

- **Request XSD Location**
- **Response XSD Location**

• Compression support for HTTP request and response

The Restful Client connector supports sending compressed HTTP requests and accepting compressed HTTP responses.

■ Compress requests

If the option is selected, the connector sends the compressed HTTP request that contains an HTTP header Content-Encoding:gzip.

■ Accept compressed responses

If the option is selected, the connector sends the HTTP request that contains an HTTP header Accept-Encoding:gzip, which tells the server that the connector allows a compressed response.

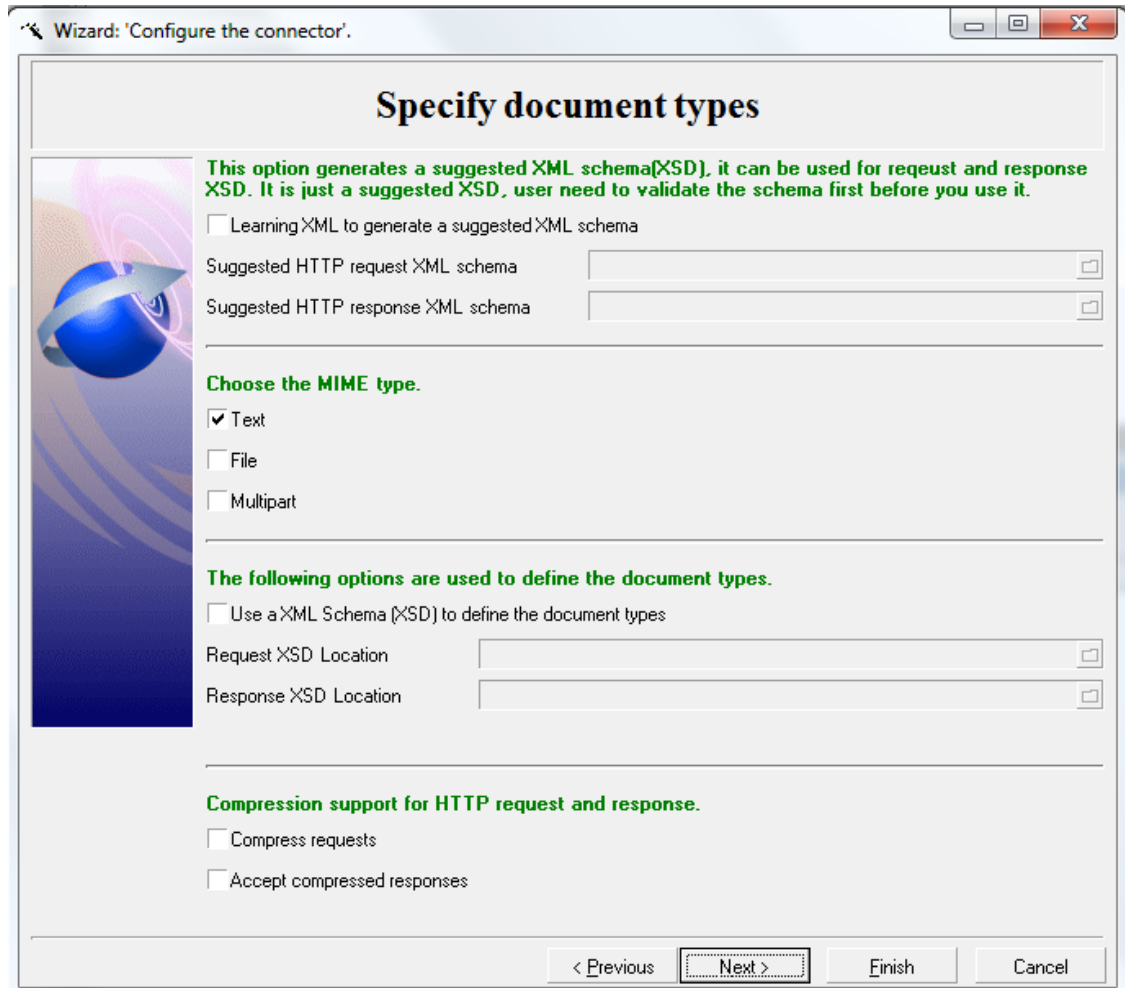


Figure 9-16: Specify document types

An example of using the RESTful Client connector

The RESTful Client connector provides a set of sample scenarios to query an airport status service and then converts the responded information to an XML file. The image below shows the scenarios in the Scenario diagram:



Figure 9-17: Sample scenarios of the RESTful Client connector

The steps are as follows:

1. Open and run `learn-xsd_initialize.scn`.
2. Open and run `airport-status.scn`.

Open and run learn-xsd_initialize.scn

The `learn-xsd_initialize.scn` is constituted of a single connector (RESTful Client Connector) and designed to generate a schema file `airport_status.xsd` that will be used later by the other scenario `airport-status.scn`. You must open and run `learn-xsd_initialize.scn` before querying a web service from a specific website. See the descriptions for each configuration wizard page as below.

In this sample scenario, the URL in the Server field indicates that the RESTful Client Connector sends a request to the FAA Airport Status REST service interface via `http://services.faa.gov/airport/status/`

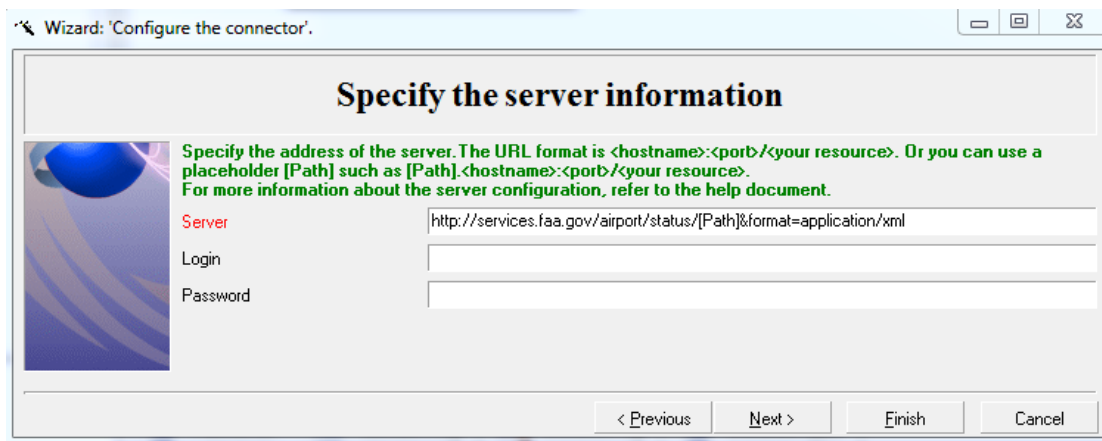


Figure 9-18: Specify server information in sample

The Specify document types page displays the necessary configuration settings for generating an XML schema file. You can view the Learning XML to generate a suggested XML schema check box selected, and the suggested directory of the schema (XSD) file for HTTP response. Change the file location if needed. Text is selected as the MIME type in this example.

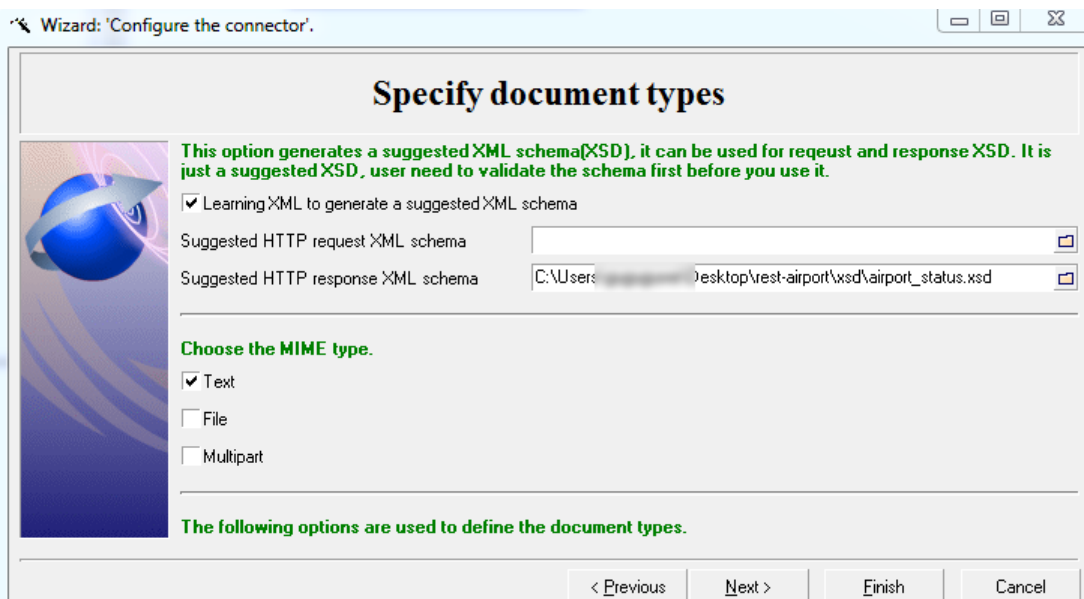


Figure 9-19: Specify document types in sample

A schema file `airport_status.xsd` is generated after you run `learn-xsd_initialize.scn`.

Open and run `airport-status.scn`

The scenario `airport-status.scn` consists of the following:

- **Get Airport Code (a Delimited Text connector)** - Contains an airport code list to request status information from the FAA Airport Status REST web service.
- **Request basic engine** - Creates a request with airport codes as parameters.
- **FAA Service (a RESTful Client connector)** - Sends the request to the FAA Airport Status REST interface, and converts the response to the data of Connect-It Document type.
- **Save response basic engine** - Saves the airport status information in the XML format.
- **XML (an XML connector)** - Output the airport status information in the XML format. You can see the updated airport status of the two airports in the file `AirportStatus_001.xml`.

In this scenario, with the XSD file generated by running `learn-xsd_initialize.scn`, the RESTful Client connector (named FAA Service) converts the response to data with Connect-It document type. See the image below for the produced document type.

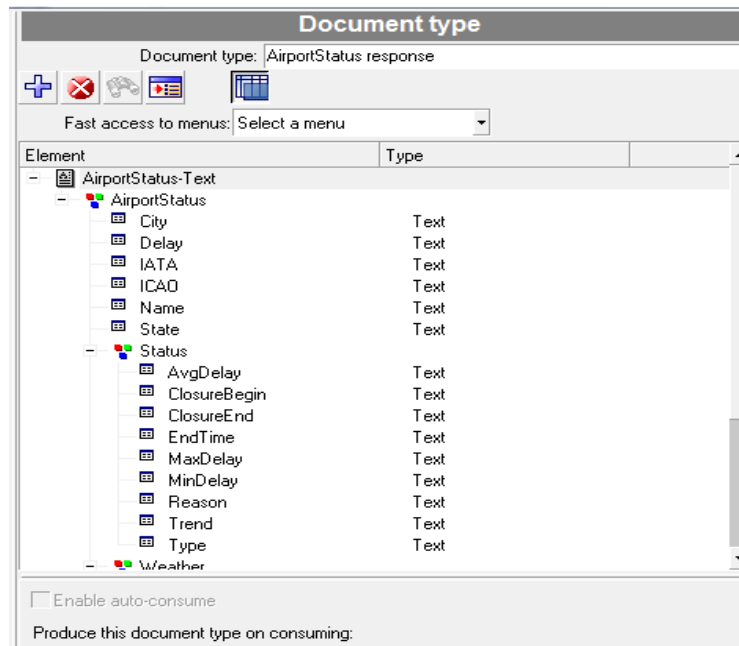


Figure 9-20: Produced document type of the RESTful Client connector

SOAP Connector

The SOAP message protocol (Simple Object Access Protocol) is a data-exchange standard for distributed environments based on the use of communication protocols (transport layers) such as HTTP, SMTP or other Internet protocols, and provides a data-exchange mechanism based on XML messages. Message Transmission Optimizing Technique (MTOM) is used to send binary data with

SOAP envelopes. Using the WSDL (Web Service Description Language), the SOAP connector obtains from the service an auto-description of a given Web service. This auto-description contains the protocols used by the service and the list of formats that an operation is capable of processing.

Example of using the SOAP connector

1. An external application produces a document containing a zip code. A mapping enables the SOAP connector to consume this document.
2. The connector sends a request by sending this zip code as a parameter to a Web service that provides temperatures.
3. The Web service sends a response containing the temperature corresponding to the zip code. In response to this, the SOAP connector spontaneously produces a document in Connect-It.
4. The produced document is used in a feedback loop that sends information to the external application that initially provided the zip code.

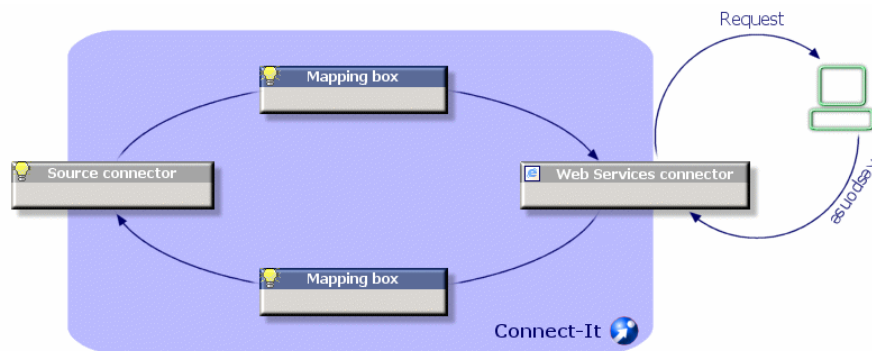


Figure 9-21: SOAP connector request and response

Note: Certain Web services do not function on the query/response model and cannot provide a response.

The notion of auto consumed documents is introduced to simplify the creation of scenarios that use SOAP connectors. Because SOAP connectors have to consume a document to produce the response in turn, the data generator connector was needed for the SOAP connector to act as a source connector. Now, with the auto consumed document types, the response of a SOAP connector can be produced without having to add a source connector. Of course, A source connector still can be used to fill the Web service request if needed but it is no more the only way.

How it works

We include the source connector code inside the SOAP connector so that we are able to fill a consumed document type in the SOAP connector exactly as we do for the Source connector. When creating a new auto produced document type: int the "Define the produced document type" window, you can select a produced document type. This document type contains 2 main nodes.

- **\$ParentDoc\$ n**
- **the Web service response document type**

Note the following:

- If you unfold \$ParentDoc\$, you will be able to see the request document type associated to the response.
- If you add to your document type a \$ParentDoc\$ child node, this will create or update the Web service request document type.
- When a Web service request document type is created on the fly, the layer property "Produce this document type on consuming" on the "Document produced on consumption" tab is automatically synchronized with the created document type Id.
- If you switch the layer property "Produce this document type on consuming" on the "Document produced on consumption" tab to an other existing value, the auto consumed node of the user document type will be updated according to the existing document type.
- You can set the auto consumed document type values in the "executing" tab. (see the source connector documentation to get more details on the control)

Limitations - SOAP connector

The SOAP connector does not support:

- **Operations with identical names**
The SOAP connector only supports operations with unique names in the same Web service. If a Web service references two operations with identical names, the connector will only publish the first one.
- **Queries with multiple responses**
The SOAP connector associates a response (produced document type) to every request (consumed document type). If a Web service associates more than one response to a request, the connector only publishes the first operation.
- **Responses with multiple requests**
If a Web service associates more than one request to a response, the connector only publishes the first operation.

Protocols

SOAP and SOAP 1.2 by HTTP protocol, MTOM attachments, HTTP GET, HTTP POST: supported

MIME: not supported

Out-of-box Scenarios

See [SOAP Scenarios](#).

Configuring the SOAP Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Specify the WSDL

This page enables you to enter the connector's connection parameters to a Web service.

- **WSDL address** field: The WSDL URL address enables you to retrieve for each Web service:
 - the communication protocols used
 - the supported operations
 - the format of these operations
 - the data consumed and the data produced
- **Login:** field: User identifier
- **Password** field

Example: <http://www.xmethods.net/sd/2001/TemperatureService.wsdl>

- **Choose the protocol you want to use**
Select from the list of protocols displayed the transport protocol to use. By default, SOAP is selected. If you choose **All**, all the listed protocols are used and operations are prefixed according to the protocol.

HTTP options

This page is available only if you select **SOAP**, **SOAP 1.2** or **All** in the previous **Specify the WSDL** page. You can choose to configure some HTTP options for the SOAP request and response interaction. By default the options are disabled.

- **User-Agent Header** Set the HTTP User-Agent Header. If none is specified, the default HttpClient header **Axis2** is used.
- **Socket Timeout (in seconds)** Specify the socket timeout for HTTP requests in seconds. The default value is **60**. The value of **0** means to turn off timeout functionality (never timeout).
- **Performance tuning options** The SOAP connector provides some options for performance tuning. Select to use these options depending on the web service context.
 - **Compress SOAP requests (gzip)** If the option is selected, the connector sends the compressed SOAP request that contains an HTTP header `Content-Encoding:gzip`.
 - **Accept compressed SOAP responses** If the option is selected, the connector sends the HTTP request that contains an HTTP header `Accept-Encoding:gzip`, which tells the server that the connector allows a compressed response.
 - **Close the HTTP connection after each SOAP request** Select the option if you want to disable the HTTP persistent connection and open a new connection after a SOAP request/response pair. HTTP persistent connection, also called HTTP keep-alive, is the idea of using the same TCP connection to send and receive multiple HTTP requests and responses. HTTP persistent connection has advantages, for example, less CPU and memory usage and reduced network congestion, versus disadvantages such as keeping unnecessary connections open for many seconds after a document is retrieved.
 - **Enable content-chunking for SOAP requests** Select the option if you want to use the Chunk Transfer Coding to transfer content in a series of chunks. It uses the `Transfer-Encoding` HTTP request header with value `chunked`.
 - **Enable MTOM attachments for SOAP requests** Select to activate or not the usage of MTOM attachments to send SOAP requests. MTOM is an efficient method of sending binary

data to and from Web services. By default the SOAP connector transmits binary data using Base64 encoding. Note that both the transporting methods have overheads. It is recommended to use MTOM for large binary data whereas Base64 is still convenient for small binary data (for example, file size < 1024 bytes).

- Use MTOM attachments only for binary data larger than (in bytes):** A numeric box will be activated if the "Enable MTOM attachments" check box is selected. You can specify the threshold above which binary data will be sent as MTOM attachments. The default value is 1024 bytes, which means binary data smaller than 1024 bytes is sent as Base64 strings while binary data larger than 1024 bytes is sent as MTOM attachments.
- Show the collection node 'HTTPHeader' in all document types** Select the option if you want to add user-defined HTTP headers for SOAP requests. As a result, the HTTPHeader collection node displays in each document type, and you can add or edit elements under the HTTPHeader node. See the images below for the comparison between selecting the option and unselecting the option.

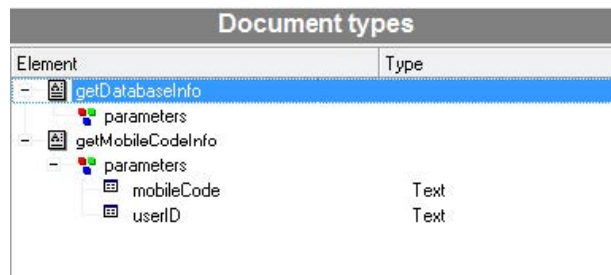


Figure 9-22: Option unselected - no HTTPHeader collection node displays

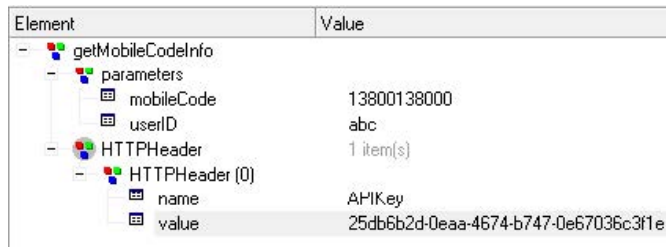


Figure 9-23: Option selected - HTTPHeader collection node displays in each document type (define an HTTPHeader for APIKey)

Sometimes the SOAP server requests specific HTTP headers such as APIKey and extra security ID. Here is an example of adding HTTPHeader elements in the request. Set the name as **APIKey** and the value as **25db6b2d-0eaa-4674-b747-0e67036c3f1e**, and the HTTP header messages display the following:

```
POST: http://www.webxml.com.cn/WebServices/MobileCodeWS.asmx HTTP/1.1
Content-Type: text/xml; charset=UTF-8
SOAPAction: "http://WebXml.com.cn/getMobileCodeInfo"
APIKey: 25db6b2d-0eaa-4674-b747-0e67036c3f1e
User-Agent: Axis2
Host: www.webxml.com.cn
Proxy-Connection: Keep-Alive
Content-Length: 386
```


UTF-8 encoding

The SOAP connector specifies the UTF-8 character set by default with the HTTP header “contentType”. For example, requests for protocol SOAP use “contentType=text/xml charset=utf-8”. If you need to use another character set to encode the requests, you have to specify it. To do so, add the following to the JVM options text box in the Configure the JVM configuration screen: –
`Dcit.ws.http.charset=iso-8859-1`

MTOM in SOAP responses

Connect-It is able to handle MTOM attachments in SOAP or SOAP 1.2 responses without any configuration. In other words, if the web service sends binary data to Connect-It’s SOAP connector, using either Base64 encoding or MTOM attachments, the binary data will be correctly handled in a transparent way for the Connect-It users.

Consumption directives - SOAP connector

Operations to apply Select the query operation so that the connector can send data to the Web service's server. If you select none in the **Operation to apply** field, the query operation will still be used. In this case, a warning appears in the Document log indicating that this operation has been performed.

Published document-types

The SOAP connector publishes document types in production corresponding to what is defined by WSDL. The document types available in consumption mode correspond to the Web service operations. For each operation there is a document type available for consumption (sending a request) and for production (receiving the response).

Naming the published document types

Note: *The SOAP connector does not publish operations that use an unsupported protocol.
 Example: the MIME protocol.*

The names you assign to document types published by the SOAP connector must have a prefix corresponding to the communication protocol and name of the operation concerned.

Consumed document types

SOAP connector - Consumed document types

Protocol	Consumed document types
SOAP Messaging	Soap-[Name of the operation]
SOAP1.2 Messaging	Soap12-[Name of the operation]
HTTP GET	HttpGet-[Name of the operation]
HTTP POST	HttpPost-[Name of the operation]

Note: *Connect-It specified JVM options are applicable to this connector. See the [Configuring the JVM](#) topic for more information.*

XML Connector

Connector type: **CONSUMPTION and PRODUCTION**

The XML enables the:

- Processing of XML files located on the computer or network on which Connect-It is installed
- Processing of XML files located on FTP or Web sites

Note: *The XML files processed can be compressed (gzip).*

Known limitations of the XML connector

In the document types processed by the XML connector, the following field types are not supported:

- **any**
- **PCDATA** (blob)
- **CDATA**
- **Namespaces**, except qualified names when XSD is selected.

UNIX environment

Under UNIX, the DTD specified in the configuration wizard of the XML connector must reference a local file, even if it is possible to specify the file on an FTP or HTTP server.

FTP and HTTP protocols

For the connector to function correctly using the HTTP and FTP protocols, the dlls wininet.dll and shlwapi.dll must be installed on the Connect-It client machine.

Using IPv6

If you access a file through HTTP on a remote server using IPv6, you will need version 7 of the `wininet.dll` file.

You can either of the following:

- install Internet Explorer 7
- download and save version 7 of this file to the Connect-It computer (default path: `C:\Windows\System32`)

Out-of-Box Scenarios

There is no out-of-the-box scenario for this connector.

Configuring the XML Connector (read)

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Certain options concerning this connector can be enabled using the **Edit > Options > Connector** menu.

The configuration steps for this connector are similar to the text connector. See [Configuring the Delimited-text Connector \(read\)](#).

Options associated with the XML connector

You can access the options of the XML connector using the menu **Edit > Options > Connector > Delimited text and XML**. The following options are associated with the XML connector in FTP protocol:

- Display the URL being processed in the Connect-It log
- Copy locally the files to read from FTP server.
Select this option to make a local copy of the files on an FTP server and to read the data from the local file.

This option must be selected if the network configuration does not enable a connection to be maintained on the FTP server long enough for a file to be processed.

Configuring the XML Connector (write)

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

The configuration steps for this connector are similar to the text connector. See [Configuring the Delimited-text Connector \(write\)](#).

Consumption Directive of the XML Connector (write) - FTP mode

The consumption directives indicate for each document type consumed by the XML connector:

- A generic name
By default, the value of this field is the name of the document type consumed by the XML connector.
- A file name extension
Enter this name without the period mark. Example: Enter **xml** instead of **.xml**; "**xml**" is the default value.

These two fields will only be used if you have chosen the **Write to a different file for each document** option during the configuration of your XML connector. When you run your scenario, the files that are written will have a composite name, which is created from the generic name, a number corresponding to the file's creation order (**_01**, **_02**, **_03**, etc.) and from the file-name extension indicated. Example: You enter the values **ebizz** and **xml**. The files created have the following names: **ebizz_01.xml**, **ebizz_02.xml**, **ebizz_03.xml**, etc.

If your XML connector consumes several document types, do not use the same generic name twice. The most recently created files will overwrite the previously created files having the same generic name.

Additional Information for the XML Connector

See [Additional Information for the uCMDB \(XML\) connector](#) and [Delimited Text-Connector Published Document Types](#).

XML Listening Connector

Connector type: **PRODUCTION**

Using the queries it receives, the XML listening connector can:

- Produce documents from these queries.
- Interact with another connector that processes the queries

The XML data is extracted from the contents of the HTTP requests sent to the servlet.

XML listening connector in publishing mode (document production)

The XML listening connector in publishing mode receives events that enable it to produce documents.



Figure 9-24: XML listening connector in publishing mode

XML listening connector in interaction mode

The connector in interaction mode listens to the queries sent by clients. The connector transmits these requests to a resource connector, who in turn produces documents.



Figure 9-25: XML listening connector in interaction mode

Compatibility of the XML listening connector

The connector is compatible with all Web servers supporting Java servlets.

About the Servlet

A servlet, integrated inside the Web server, has the task of:

- Handling incoming HTTP requests.
- Extracting the appropriate XML data from the incoming requests
- Creating a client socket if required
- Sending XML documents to the XML listening connector via the socket just created

Since the servlet must know the socket parameters (port, host) to be used when it communicates with the connector, using the XML listening connector creates a HTTP POST request for the servlet with the following form:

```
POST servletURL?socketHost=string?socketPort=string HTTP/1.1
Host: webServerHost
```

In fact, the servlet must support HTTP POST queries. For example, to specify that the URL `http://www.hp.com/conit/connector` is to create a socket for the 'localhost' host on port '1700' when it receives data from a client computer, the HTTP POST is as follows:

```
POST conit/connector?socketHost=localhost?socketPort=1700 HTTP/1.1
Host: www.hp.com
```

Using IPv6

If you are using IPv6, upgrade to JRE version 7.

Out-of-Box Scenarios

There is no out-of-the-box scenario for this connector.

Configuring the XML Listening Connector

A wizard enables you to configure the connector. You can find detailed instructions for these wizard screens under [Configuring Connectors](#). If there are any details that are not covered under that section, they are available below.

Configure connection

This frame enables you to enter the connection parameters for the XML listening connector:

- **Port**
Specify the listening port used by the XML listening connector. This port enables a Web server to create the communication channel (socket) between the Web server and the XML connector. The option **Link the connection to a Web resource** is selected by default. Disabling this option enables you to open a connector without it being linked to Web server.
- **URL**
Specify the URL of the servlet processing the HTTP requests (POST or GET) that generate XML events.
- **Maximum number of connections**
Indicate the maximum number of simultaneous connectors that can be processed by the connector. The default value of this field is 20. This field is only available if the configuration is made in advanced mode. By default, the **Refuse connection after maximum limit reached** option is cleared. In this case, as soon as the maximum number of connections is reached, the new connections are put into a queue until they can be processed by the connector.

Configure the processing of XML events

This frame enables you to enter request-processing parameters:

- **Policy of processing queries**
Indicate the connector's mode of processing queries. You can choose between **Publication** and **Interaction**.
- **Configuration file for the publications**
Indicate the path of the XML file that specifies the document types published by the connector.
- **Connector for interactions**
Indicate the connector processing interaction requests. You must be able to use this connector in production mode.

Using the XML Listening Connector

IPv6

If you are using IPv6, enter the following in the **JVM Options** field on the Configure the JVM screen:

```
-Djava.net.preferIPv6Addresses=true -Djava.net.preferIPv6Stack=true
```

Publication

When the XML listening connector is in publishing mode, the XML document received by the XML listening connector is considered as produced by the connector. It is then processed in the same way as any other document. If the processing is successful, the produced document is sent back. If the processing fails, an error message is returned.

The published document is formed as follows:

```
<DocToPublish>
<STRUCTURE name='Employee'>
<ATTRIBUTE name='FirstName'>John</ATTRIBUTE>
<ATTRIBUTE name='LastName'>Smith</ATTRIBUTE>
</STRUCTURE>
</DocToPublish>
```

The configuration file for publications or the 'SchemaFile' property is supplied to Connect-It in order to validate the publication.

For example:

```
<PublishingSchema>
<STRUCTURE name='Employee'>
<ATTRIBUTE name='FirstName' type='String' />
<ATTRIBUTE name='LastName' type='String' />
</STRUCTURE>
<STRUCTURE name='amAsset'>
<ATTRIBUTE name='AssetTag' type='String' />
<ATTRIBUTE name='dtLastModif' type='Date' />
</STRUCTURE>
<STRUCTURE name='amProduct'>
<ATTRIBUTE name='BarCode' type='String' />
</STRUCTURE>
</PublishingSchema>
```

Note: *The document sent must be fully compatible with the configuration file in order to be processed correctly.*

Interaction

When the XML listening connector is in interaction mode, the XML document must bind the target resource query (i.e. perform a Result Set type operation).

For example: recovering the 'AmAsset' type records from the default target database:

```
<operation>
<STRUCTURE name='AmPortfolio'>
<ATTRIBUTE name='AssetTag' type='String' />
<ATTRIBUTE name='FullName' type='String' />
```

```
</STRUCTURE>
</operation>
```

If a 'count' type attribute is specified, it set the maximum size of the resulting document.

For example, recovering the ten first records from the default target database:

```
<operation count='10'>
<STRUCTURE name='AmPortfolio'>
<ATTRIBUTE name='AssetTag' type='String' />
<ATTRIBUTE name='FullName' type='String' />
</STRUCTURE>
</operation>
```

If a 'target' attribute is specified, the name of the connector specified is used instead of that defined in the wizard.

For example, retrieve collection elements. If a collection is present in an .xml file received by the .xml listening connector, the configuration file for the .xml publications must have the following form:

```
<PublishingSchema>
....
<COLLECTION name='<COLLECTION_NAME>'>
<STRUCTURE name='<COLLECTION_NAME>'>
...
</STRUCTURE>
</COLLECTION>
...
</PublishingSchema>
```

In order to process data correctly, the name of the structure contained in the collection must be the same as the collection itself. Consequently, xml files that are processed must follow the form defined in the configuration file.

```
<Document>
....
<COLLECTION name='<COLLECTION_NAME>'>
<STRUCTURE name='<COLLECTION_NAME>'>
...
</STRUCTURE>
<STRUCTURE name='<COLLECTION_NAME>'>
...
</STRUCTURE>
<STRUCTURE name='<COLLECTION_NAME>'>
...
</STRUCTURE>
...
<STRUCTURE name='<COLLECTION_NAME>'>
...
</STRUCTURE>
</COLLECTION>
...
</Document>
```

For example, for produced xml documents:

```
<Document>
<STRUCTURE name='Employee'>
<ATTRIBUTE name='Name'>Bailly</ATTRIBUTE>
<ATTRIBUTE name='BarCode'>DEMO-U061</ATTRIBUTE>
<ATTRIBUTE name='Field1'>test</ATTRIBUTE>
<COLLECTION name='Group'>
<STRUCTURE name="Group">
<ATTRIBUTE name='BarCode'>EG000002</ATTRIBUTE>
</STRUCTURE>
<STRUCTURE name="Group">
<ATTRIBUTE name='BarCode'>DEMO-EG03</ATTRIBUTE>
</STRUCTURE>
</COLLECTION>
</STRUCTURE>
</Document>
```

```
<Document>
<STRUCTURE name='computers'>
<ATTRIBUTE name='test'>tt</ATTRIBUTE>
<COLLECTION name='computer'>
<STRUCTURE name='computer'>
<ATTRIBUTE name='AssetTag'>XX1</ATTRIBUTE>
<ATTRIBUTE name='PhysicalAddress'>00:00:00:00:00</ATTRIBUTE>
</STRUCTURE>
<STRUCTURE name='computer'>
<ATTRIBUTE name='AssetTag'>XX2</ATTRIBUTE>
<ATTRIBUTE name='PhysicalAddress'>00:00:00:00:01</ATTRIBUTE>
</STRUCTURE>
</COLLECTION>
</STRUCTURE>
</Document>
```

The description file is as follows:

```
<PublishingSchema>
<STRUCTURE name="computers">
<ATTRIBUTE name="test" type="String" />
<COLLECTION name="computer">
<STRUCTURE name="computer" >
<ATTRIBUTE name="AssetTag" type="String" />
<ATTRIBUTE name="PhysicalAddress" type="String" />
</STRUCTURE>
</COLLECTION>
</STRUCTURE>
<STRUCTURE name='Employee'>
<ATTRIBUTE name='Name' type='String'/>
<ATTRIBUTE name='BarCode' type='String'/>
<ATTRIBUTE name='Field1' type='String'/>
<COLLECTION name='Group'>
<STRUCTURE name="Group">
<ATTRIBUTE name='BarCode' type='String'/>
<ATTRIBUTE name='Name' type='String'/>
```



```
</STRUCTURE>  
</COLLECTION>  
</STRUCTURE>  
</PublishingSchema>
```

For example:

```
<operation target='ODBC'>  
<STRUCTURE name='AmPortfolio'>  
<ATTRIBUTE name='AssetTag' type='String' />  
<ATTRIBUTE name='FullName' type='String' />  
</STRUCTURE>  
</operation>
```

A directive may be specified in order to control the appearance of the documents recovered.

For example

Recovering records using a **WHERE** clause.

```
<operation>  
<STRUCTURE name='AmPortfolio'>  
<ATTRIBUTE name='AssetTag' type='String' />  
<ATTRIBUTE name='FullName' type='String' />  
</STRUCTURE>  
<layer>  
<STRUCTURE name='AmPortfolio'>  
<where>AssetTag LIKE 'UTL%'</where>  
</STRUCTURE>  
</layer>  
</operation>
```

This example returns to the client socket an XML document as a collection of elements:

```
<COLLECTION>  
<STRUCTURE name='AmPortfolio'>  
<ATTRIBUTE name='AssetTag'>UTL000338</ATTRIBUTE>  
<ATTRIBUTE name='FullName'>/GENASSET-338/UTL000338</ATTRIBUTE>  
</STRUCTURE>  
<STRUCTURE name='AmPortfolio'>  
<ATTRIBUTE name='AssetTag'>UTL000442</ATTRIBUTE>  
<ATTRIBUTE name='FullName'>/GENASSET-442/UTL000442</ATTRIBUTE>  
</STRUCTURE>  
. . .  
</COLLECTION>
```

Note: For all these examples, the root tag of the 'operation' queries corresponds to the operation to use on the target connector in order to perform the query. This is appropriate if the target connector is a Java connector and corresponds to the Connector for interactions field in the connector configuration wizard. For the non-Java connectors, this tag must have the value 'null'.

```
<null>  
....  
</null>
```

For example: recovering records when the target connector is not a Java connector.

```
<null>
<STRUCTURE name='AmPortfolio'>
<ATTRIBUTE name='AssetTag' type='String' />
<ATTRIBUTE name='FullName' type='String' />
</STRUCTURE>
<layer>
<STRUCTURE name='AmPortfolio'>
<where>AssetTag LIKE 'UTL%'</where>
</STRUCTURE>
</layer>
</null>
```

XML Listening Connector Example- Tomcat Web Server

This section includes an deployment example and a servlet test. We assume that the client queries are HTTP GET type requests and that the appropriate XML document is included in a 'document' parameter of the query.

Deploying a servlet on the Tomcat server

The following illustration shows the structure of the Tomcat installation folder used in this example.

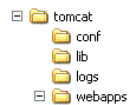


Figure 9-26: Tomcat installation folder

You must copy the Web application (.war) included in the datakit to the `webapps` folder. This file is used by the Tomcat server to deploy the servlet. The war file location is `%CIT_HOME\datakit\javasamples\servletsample\servletsample.war`.

Testing the XML listening connector

1. Start the Tomcat Web server.
2. Create a scenario in Connect-It using the XML listening connector. You can use the following parameters:
Port=1024
URL=http://localhost:8080/servletsample/queryDocument
Policy=Interaction
Resource=AssetManagement
3. Applying a scheduling to the scenario.
4. In an browser, enter the following URL: **http://localhost:8080/servletsample**. The following HTML page is displayed:

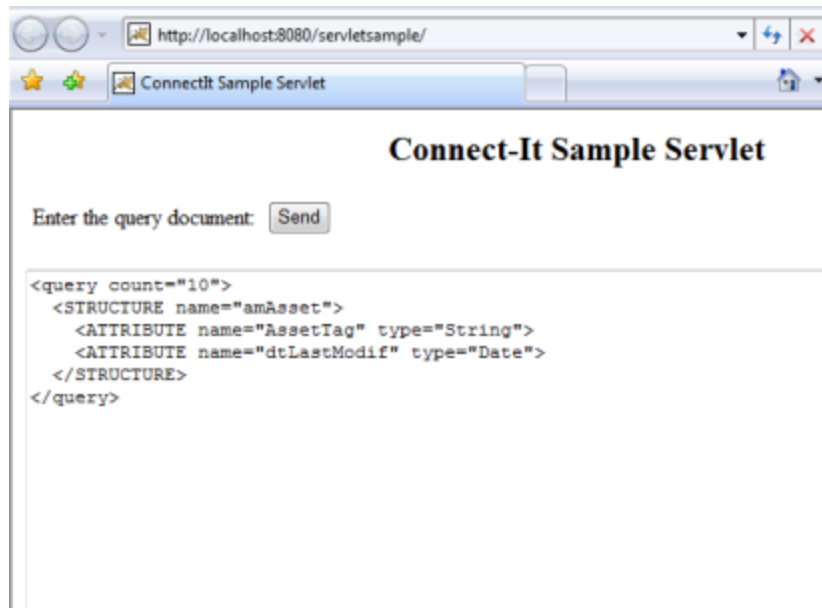


Figure 9-27: Production directives of the XML listening connector

5. Enter a document and submit it.
6. Close the scenario and stop the Tomcat server.

Production directives of the XML listening connector

These directives are present in queries processed in the form of XML documents by the connector.

Internal Tools

This section explains the tools available with Connect-It. For instructions on using the Data Generator, see the User Guide.

Status Report Connector

Connector type: **PRODUCTION**

The aim of this feature is to enhance the monitor module functionality that are unable to do the following:

- Define data types
- Design mapping to perform custom logic
- Choose a backend destination

Review legacy monitors

Monitor functionalities are provided by default. The legacy monitor module can save several types of data into different backend destinations, such as E-mail, ServiceCenter databses, Events logs and Asset Manager databases. For each kind of report generated by a monitor, database formats are internally defined and cannot be changed. Furthermore, there are no trace reports. In another words, if the session report fails, no further action could be defined to deal with this situation.

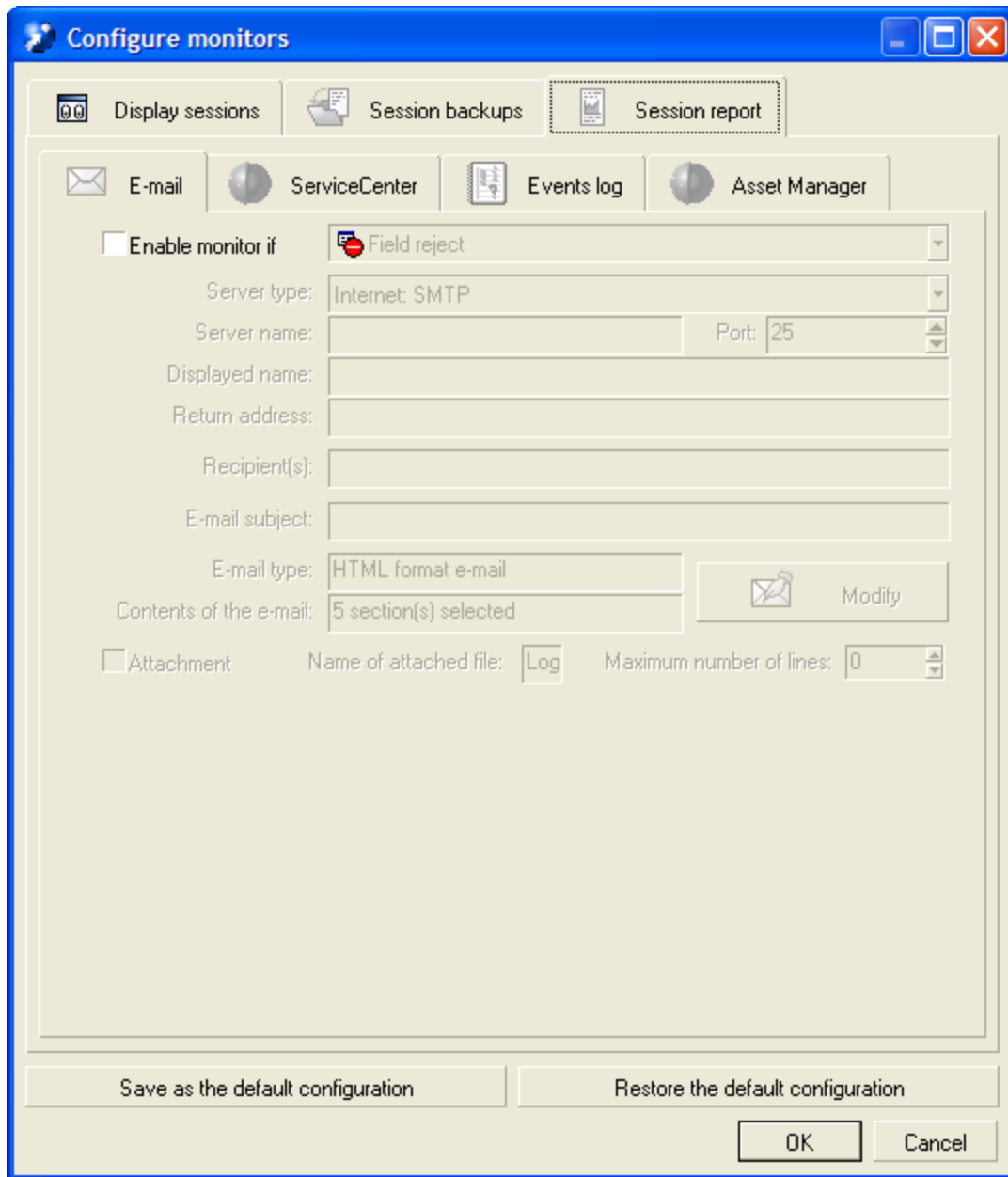


Figure 9-28: Configuring monitors for the Status Report connector

Motoring/Status report connector

A status report connector is another choice provided to save custom status reports into a specified backend destination like a database. You will have the ability to:

- Define data type end users
- Design mapping to perform custom logic
- Choose a backend destination
 - Set the status report trigger

The status report connector runs at the second session. You can add the status report

connector into the normal scenario. When running the scenario, the status report connector will be triggered after the normal session finishes. During session 2, all status items generated from the ServiceCenter connector, the text connector, and the mapping will be collected and grouped into a list in a report, which is saved in the database backend through mapping1 logic.

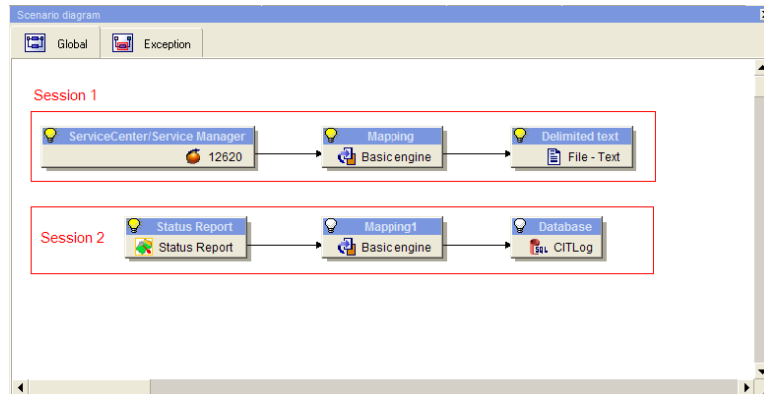


Figure 9-29: Status Report connector: running at the second session

Process Report

There is no process report for this connector.

Status report

A status report provides the status of each connector. The status provides information related to both the connectors and the functionality modules. A status report item is organized by three levels of information: the most common statuses, the properties of connectors which generates status information, and the real status items.

Configuring the Status Report Connector

Produced document types design

A produced document type is for collecting status information from connectors but not modifying them. The screenshot below show how the status report is represented. Define the produced documentation type for the Status Report. If you select **Logs**, you can select the log types to include: **Info** messages, **Warning** messages, and **Error** messages. If you select **No Filter**, all types of logs will be included.

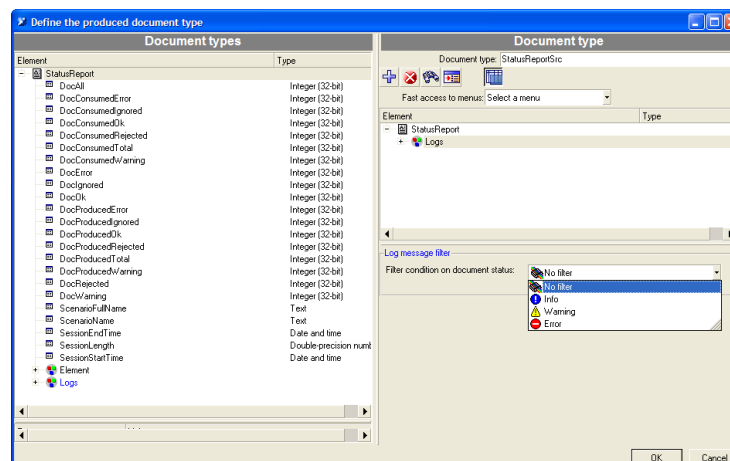


Figure 9-30: Status Report Connector produced document types

There are no consumed document types for the status report.

Mapping

You could use the status connector as an status generator, similar to a data generator. It collects report status items will be collected automatically and then links to destination connector.

Chapter 10

Out-of-Box Scenarios

Out-of-box scenarios are classified according to their source connector and can be divided into three categories:

- **Sample scenarios**
These scenarios only enable you to view how one or more connectors work. They have no other value and should not under any circumstances be used in a production environment.
- **Business scenarios to adapt**
These scenarios correspond to integration processes that actually occur in real situations. However, to be used in production environments, they must be adapted by the user. These adaptations notably concern the mappings between source connectors and destination connectors.
- **Standard business scenarios**
These scenarios correspond to integration processes that actually happen in real situations. They can be used directly in production environments.

HP Asset Manager Connector Scenarios

The table below shows the out-of-the-box scenarios that use the Asset Manager connector.

HP Asset Manager Connector Scenarios

Scenario	Description
ac\lac44\catalog.scn ac\lac50\catalog.scn <u>Scenario type: Business scenario to be adapted</u>	These scenarios enable you to import catalog-related data into an Asset Manager database.
ac\lac51\esscat\am51sm70\catalogitems.scn *	This scenario synchronizes catalog items from Asset Manager to ServiceCenter via SC Web Service.
ac\lac51\esscat\am51sm70\categories.scn *	This scenario synchronizes catalog categories from Asset Manager to ServiceCenter via SC Web Service.
ac\lac51\esscat\am51sm70\sso.scn *	This scenario synchronizes operators data from Asset Manager to ServiceCenter via SC Web Service.
ac\lac51\esscat\am51sm70\status.scn *	This scenario synchronizes internal request data from Asset Manager to ServiceCenter via SC Web Service.
ac\lac51\esscat\am51sm70\users.scn *	This scenario synchronizes employee data from Asset Manager to ServiceCenter via SC Web Service.

Scenario	Description
aclac51\esscat\am51sm71\catalogitems.scn aclac51\esscat\am51sm71\categories.scn aclac51\esscat\am51sm71\sso.scn aclac51\esscat\am51sm71\status.scn aclac51\esscat\am51sm71\users.scn	These scenarios synchronize data from Asset Manager to HP Service Manager. See the following Asset Manger document for more information: <HP Asset Manager installation folder>\doc\pdf\AM{xxx}-Procurement-EN.pdf
aclam52\esscat\am52sm71\catalogitems.scn aclam52\esscat\am52sm71\categories.scn aclam52\esscat\am52sm71\sso.scn aclam52\esscat\am52sm71\status.scn aclam52\esscat\am52sm71\users.scn	See the following Asset Manger document for more information: <HP Asset Manager installation folder>\doc\pdf\AM{xxx}-Procurement-EN.pdf
aclac51\sacm\sm71am51\amsm-ci-ppt-link.scn aclac51\sacm\sm71am51\amsm-ppt.scn aclac51\sacm\sm71am51\smam-ppt.scn aclac51\sacm\sm71am51\smam-wo.scn	See the Asset Manger and SACM documentation for more information.
aclam52\sacm\sm71am52\amsm-ci-ppt-link.scn aclam52\sacm\sm71am52\amsm-ppt.scn aclam52\sacm\sm71am52\smam-ppt.scn aclam52\sacm\sm71am52\smam-wo.scn	See the Asset Manger and SACM documentation for more information.
aclam52\sacm\sm92am52\amsm-ci-ppt-link.scn aclam52\sacm\sm92am52\amsm-ppt.scn aclam52\sacm\sm92am52\smam-ppt.scn aclam52\sacm\sm92am52\smam-wo.scn	See the Asset Manger and SACM documentation for more information.
aclac51\sacm\ucmdb8am51\ucmdbam-bs.scn aclam52\sacm\ucmdb9am52\ucmdbam-ba.scn aclam52\sacm\ucmdb9am52\ucmdbam-bs.scn	These scenarios enable you to replicate Business Services from UCMDB to Asset Manager.
aclam93\esscat\am93sm92\catalogitems.scn aclam93\esscat\am93sm92\categories.scn aclam93\esscat\am93sm92\sso.scn aclam93\esscat\am93sm92\status.scn aclam93\esscat\am93sm92\users.scn	See the following Asset Manger document for more information: <HP Asset Manager installation folder>\doc\pdf\AM{xxx}-Procurement-EN.pdf

* It is necessary to import UNL files from the datakit\sc\sc62 folder into the SC server before running these scenarios.

HP Business Service Management Dashboard / HP Universal CMDB Connector Scenarios

The table below shows the out-of-the-box scenarios that use the HP Business Service Management Dashboard / HP Universal CMDB connector.

Scenario type: Business scenario to be adapted

HP Business Service Management Dashboard / HP Universal CMDB Connector Scenarios

Scenario	Description
merq\mam242sc61\scmam-ci.scn	This scenario is used to transfer data from ServiceCenter to HP Application Mapping. The scenario exports Configuration Items (devices) from ServiceCenter and imports them into HP Application Mapping.
merq\mam242sc61\mamsc-ci-service.scn	This scenario is used to transfer data from ServiceCenter to HP Application Mapping. This scenario exports Business Services and relationships between Business Services from HP Application Mapping and imports them into ServiceCenter as devices.
merq\mam242sc61\mamsc-ci.scn	This scenario is used to transfer data from HP Application Mapping to ServiceCenter. This scenario exports Configuration Items (devices) and relationships between Configuration Items from HP Application Mapping and imports them into ServiceCenter. Reconciliation for Configuration Items is done using the MAC address.
merq\bac51sc61\scbac-ci.scn	This scenario is used to transfer data from ServiceCenter to HP BAC. The scenario exports Configuration Items (devices) from ServiceCenter and imports them into BAC.
merq\bac51sc61\scbac-changes.scn	This scenario is used to transfer data from ServiceCenter to HP BAC. This scenario exports changes and associated Configuration Items from ServiceCenter and imports them into BAC. This scenario exports the ServiceCenter URL that is associated with the change to allow the ServiceCenter Web page to be viewed in BAC.
merq\bac51sc61\bacsc-incident.scn	This scenario is used to transfer data from HP BAC to ServiceCenter. This scenario automatically opens a support ticket in ServiceCenter.

Scenario	Description
merq\bac51sc61\bacsc-ci.scn merq\bac51sc62\bacsc-ci.scn	These scenarios is used to transfer data from HP BAC to ServiceCenter. This scenario exports Configuration Items (devices) and relationships between Configuration Items from BAC and imports them into ServiceCenter.
merq\bac51sc62	Same as merq\bac51sc61\ scenarios.
merq\mam242sc62	Same as merq\mam242sc61 scenarios.
merq\ucmdb75sm71\smucmdb.scn merq\ucmdb75sm71\ucmdbsm.scn	Before running these scenarios, import datakit file to SM and UCMB.
merq\ucmdb8sm71\smucmdb.scn merq\ucmdb8sm71\ucmdbsm.scn	<ul style="list-style-type: none"> • datakit\sc\sm71\dbdict-chm-im.unl for SM • webservice-cm-im-chm.unl for SM
merq\ucmdb8sm92 merq\ucmdb8smws92	<ul style="list-style-type: none"> • datakit\merq\UCMDB_Configuration\IntegrationToSM.xml for UCMB, UCMB connector need configure this xml file.

HP Cloud Service Automation Connector Scenarios

The table below shows the out-of-the-box scenarios that use the HP Cloud Service Automation connector.

Scenario type: [Sample scenario](#)

HP Cloud Service Automation Connector Scenarios

Scenario	Description
samples\csa-admin.scn	This scenario enables you to retrieve CSA catalogs and subscriptions of one or multiple organizations.
samples\csa-consumerl.scn	This scenario enables you to retrieve CSA catalogs and subscriptions of a user.

HP Discovery and Dependency Mapping Inventory Connector Scenarios

The table below shows the out-of-the-box scenarios that use the HP Discovery and Dependency Mapping Inventory connector. These scenarios are documented separately.

Scenario type: [Business scenario to be adapted](#)

HP Discovery and Dependency Mapping Inventory Connector Scenarios

Scenario	Description
ed\ddmi75\ddmi75am51\ddmiac-hpovcmse.scn	These scenarios enable you to transfer data from DDMI and HP CM Service Events databases to an

Scenario	Description
<p>ed\ddmi75\ddmi75am52\ddmiam-hpovcmse.scn</p> <p>ed\ddmi76\ddmi76am52\ddmiam-caevents_sw.scn</p> <p>ed\ddmi76\ddmi76am93\ddmiam-caevents_sw.scn</p> <p>ed\ddmi77\ddmi77am52\ddmiam-caevents_sw.scn</p> <p>ed\ddmi77\ddmi77am93\ddmiam-caevents_sw.scn</p> <p>ed\ddmi93\ddmi93am52\ddmiam-caevents_sw.scn</p> <p>ed\ddmi93\ddmi93am93\ddmiam-caevents_sw.scn</p>	<p>Asset Manager database. See the Customizing the Asset Manager Database for additional information on the customizations that need to be made to your Asset Manager database.</p> <p>Note: <i>When loading the scenario, a warning message is displayed. This message requests that you define the data to use in the mactable for the scenario.</i></p>
<p>ed\ddmi75\ddmi75am51\ddmiac_swnorm.scn</p> <p>ed\ddmi75\ddmi75am52\ddmiam_swnorm.scn</p> <p>ed\ddmi76\ddmi76am52\ddmiam_swnorm.scn</p> <p>ed\ddmi76\ddmi76am93\ddmiam_swnorm.scn</p> <p>ed\ddmi77\ddmi77am52\ddmiam_swnorm.scn</p> <p>ed\ddmi77\ddmi77am93\ddmiam_swnorm.scn</p> <p>ed\ddmi93\ddmi93am52\ddmiam_swnorm.scn</p> <p>ed\ddmi93\ddmi93am93\ddmiam_swnorm.scn</p>	<p>These scenarios are used to access the software normalization module available in Asset Manager. They use amlInventModel records to assign a model to the software installations.</p>
<p>ed\ddmi75\ddmi75am51\ddmiac-reconc.scn</p> <p>ed\ddmi75\ddmi75am52\ddmiam-reconc.scn</p>	<p>These scenarios are used to generate reconciliation proposals for an Asset Manager application. The Asset Manager of Reconciliation module must be activated in this application. The reconciliation proposal used for Asset Manager is activated by a script for the IMemorySizeMb field which calls the ValidateReconcUpdate global function.</p>
<p>ed\ddmi75\ddmi75sm71\ddmism.scn*</p>	<p>These scenarios transfer inventory data to a</p>

Scenario	Description
<p>ed\ddmi76\ddmi76sm92\ddmism.scn ed\ddmi76\ddmi76smws92\ddmism.scn ed\ddmi93\ddmi93sm92\ddmism.scn</p> <p>* Before running the scenarios marked with an asterisk, import the UNL from the datakit\sc\sm71 folder into SM server.</p>	<p>ServiceCenter application. The following mappings are used in the scenario:</p> <ul style="list-style-type: none"> • ICM-Mainframe: This mapping processes information related to Mainframe Devices. • ICM-Telecom: This mapping processes information related to telecommunication equipment (telephones, PABXs, etc). • ICM-Computer: This mapping processes information related to computer work stations. • ICM-Network Component: This mapping processes information related to network components (switches, routers, etc). • ICM-Office Electronic: This mapping processes information related to electronic devices other than computers (fax machines, video projectors, photocopiers, etc). <p>In addition to the mappings used for physical devices, the following mappings process information related to software:</p> <ul style="list-style-type: none"> • PC Software Uninstall: This mapping must be launched before the PC Software Uninstall mapping. • PC Software (Install): This mapping re-writes new or existing information linked to each computer.
<p>ed\ddmi75\ddmi75am51\ddmiac-mobiledevices.scn ed\ddmi75\ddmi75am52\ddmiac-mobiledevices.scn</p>	<p>These scenarios enable you to transfer mobile device data from Discovery and Dependency Mapping Inventory databases to an Asset Manager database.</p>
<p>ed\ddmi76\ddmi76am52\ddmiam-aix.scn ed\ddmi77\ddmi77am52\ddmiam-aix.scn ed\ddmi93\ddmi93am52\ddmiam-aix.scn</p>	<p>Refer to details in the DDMI documentation. Verify that the DDMI connector uses the following extension file: \config\ed\ddmi76\config\ddmi76db-aix.cfg.</p>
<p>ed\ddmi76\ddmi76am52\ddmiam_hw.scn ed\ddmi76\ddmi76am93\ddmiam_hw.scn</p>	<p>These scenarios replicate all types of hardware data.</p>

Scenario	Description
<p>ed\ddmi77\ddmi77am52\ddmiam_hw.scn</p> <p>ed\ddmi77\ddmi77am93\ddmiam_hw.scn</p> <p>ed\ddmi93\ddmi93am52\ddmiam_hw.scn</p> <p>ed\ddmi93\ddmi93am93\ddmiam_hw.scn</p>	
<p>ed\ddmi76\ddmi76am52\ddmiam_sw.scn</p> <p>ed\ddmi76\ddmi76am93\ddmiam_sw.scn</p> <p>ed\ddmi77\ddmi77am52\ddmiam_sw.scn</p> <p>ed\ddmi77\ddmi77am93\ddmiam_hw.scn</p> <p>ed\ddmi93\ddmi93am52\ddmiam_sw.scn</p> <p>ed\ddmi93\ddmi93am93\ddmiam_sw.scn</p>	<p>These scenarios replicate software data. It does not retrieve any information from HP Client Automation (as ddmiam-caevents_sw.scn does).</p>
<p>ed\ddmi76\ddmi76am52-parallel\pre-import-once.scn</p> <p>ed\ddmi76\ddmi76am93-parallel\pre-import-once.scn</p> <p>ed\ddmi77\ddmi77am52-parallel\pre-import-once.scn</p> <p>ed\ddmi77\ddmi77am93-parallel\pre-import-once.scn</p> <p>ed\ddmi93\ddmi93am52-parallel\pre-import-once.scn</p> <p>ed\ddmi93\ddmi93am93-parallel\pre-import-once.scn</p> <p><i>Note: these must be run first before other *-parallel scenarios.</i></p>	<p>These scenarios import a list of pre-defined natures, hardware models, software models, and client-resource relationships into the Asset Manager database. The scenario should be executed only once as an initialization setup. It's not necessary to associate the scenario to a scheduler. See the configuration instructions below.</p>
<p>ed\ddmi76\ddmi76am52-parallel\ddmiam.scn</p> <p>ed\ddmi76\ddmi76am93-parallel\ddmiam.sc</p>	<p>These scenarios import hardware data and software data separately. See the configuration instructions below.</p>

Scenario	Description
<p>ed\ddmi77\ddmi77am52-parallel\ddmiam.sc</p> <p>ed\ddmi77\ddmi77am93-parallel\ddmiam.scn</p> <p>ed\ddmi93\ddmi93am52-parallel\ddmiam.scn</p> <p>ed\ddmi93\ddmi93am93-parallel\ddmiam.scn</p>	
<p>ed\ddmi76\ddmi76am52-parallel\ddmiam_norm.scn</p> <p>ed\ddmi76\ddmi76am93-parallel\ddmiam_norm.scn</p> <p>ed\ddmi77\ddmi77am52-parallel\ddmiam_norm.scn</p> <p>ed\ddmi77\ddmi77am93-parallel\ddmiam_norm.scn</p> <p>ed\ddmi93\ddmi93am52-parallel\ddmiam_norm.scn</p> <p>ed\ddmi93\ddmi93am93-parallel\ddmiam_norm.scn</p>	<p>These scenarios do the same as the ddmiam_swnorm.scn but uses the amInventModel records to assign a model to the software installations. See the configuration instructions below.</p>
<p>ed\ddmi76\ddmi76am52-parallel\ddmiam_mobile.scn</p> <p>ed\ddmi77\ddmi76am93-parallel\ddmiam_mobile.scn</p> <p>ed\ddmi77\ddmi77am52-parallel\ddmiam_mobile.scn</p> <p>ed\ddmi77\ddmi77am93-parallel\ddmiam_mobile.scn</p> <p>ed\ddmi93\ddmi93am52-parallel\ddmiam_mobile.scn</p> <p>ed\ddmi93\ddmi93am93-parallel\ddmiam_mobile.scn</p>	<p>These scenarios replicate mobile devices, software installed on them and their models from HP Discovery and Dependency Mapping Inventory to Asset Manager. See the configuration instructions below.</p>

-parallel Scenarios

A set of new Discovery and Dependency Mapping Inventory-Asset Manager scenarios are now available. These scenarios contain "-parallel" in their names. These scenarios allow parallel import of hardware and software inventory data from HP Discovery and Dependency Mapping Inventory database to Asset Manager database, which enhances the performance of the import. The original

scenarios (containing folder without parallel in the name) are also provided. You can choose between the two sets of scenarios, depending on the volume of their inventory data to be imported and the requirements for import performance.

Configuration

- pre-import-once.scn

When configuring the XML connector, you need to specify the path to the following files:

The XML file required on the Select files or folders page:

```
<Connect-It installation folder>\datakit\ed\special_model.xml
```

The XML schema required on the Choose a DTD/XSD page:

```
<Connect-It installation folder>\datakit\ed\model.xsd
```

- ddmiam.scn or ddmiam_norm.scn
 - The DDMI connectors: You should point the extension file required on the **Define document types** page of the connector configuration wizard to the `config` sub-folder of the scenario folder, not the default folder.
 - The Asset Manager connectors: To enable parallel import, you need to select the **parallelize document consumption** option on the **Parallelize consumption** page of the connector configuration wizard.
Note: Do not enable parallel import for those connectors with `Model` in their name. The import of models must not be done in parallel. Do not enable parallel import for the connector named `amPreload`. This connector is used to for query purpose only.
 - Do not change the default order of the produced document types when you customize the scheduling of the scenario
- ddmiam_mobile.scn
This scenario does not support parallel import. Do not select the parallelize document consumption option when configuring the Asset Manager connector.

Mappings

The following mappings are used in the scenarios above:

- Scanned Computers: This mapping is used to map inventory data to:
 - Software applications (licenses and usage)
 - Computer components (network cards, CPUs, disk drives - logical and physical, monitors, extension cards)
- Network Devices: This mapping handles network components (switches, routers, etc).
- Non-Scanned Computers: This mapping handles computers for which the scan returned no data (either because the network inventory has not been launched or because no agent is installed on the computer).
- Network Device Structures: This mapping handles network devices and is used to obtain detailed information about internal components such as backplanes, cards plugged into backplanes, etc. The hierarchy of the components is captured and maintained.
- Connections: This mapping produces interconnection details for network devices (network topology).

Scanned Computers Mapping

The part related to software usage has been changed for this mapping. In the previous version of DDMI software usage was calculated per computer. In the current version, software usage is calculated per computer and user. Two users were created in order to maintain compatibility with previous versions.

ALL USERS: All users of a given computer. If no named user is available, software usage data is assigned to this user. If a named user is associated with a given computer, ALL USER data must not be used.

SYSTEM USER: Retrieves usage data concerning tasks for a given computer that can not be assigned to a user.

Network Devices Mapping

This mapping handles data involving network devices.

In the previous version of DDMI, the `ddmiam.scn` scenario was divided into several different mappings in order to retrieve pertinent information. This was particularly useful when conflicting information was retrieved for a given computer. The Network Devices mapping provides a good illustration of how the information is retrieved. The Scanned Computers mapping uses the **hwNetworkData.hwNetworkCards** collection.

Non-Scanned Computers Mapping

This mapping is similar to the Network Device mapping and allows information related to a computer (scanned or not scanned) to be imported without importing information related to the network devices.

Network Device Structures Mapping

This mapping prepares the internal hierarchy of network devices information for the Asset Manager database. The **AddOn** collection and the link to the parent tables are used in order to reproduce the hierarchy.

Connections Mapping

This mapping executes last after all information related to network devices has been processed and saved to the Asset Manager database. This mapping is used to provide connection information between the different devices. This mapping is used to create a Business service named Network equipment in Asset Manager. Network elements are categorized as follows:

- A computer always has a parent whose type is Network Device.
- Parent-child relationships do not exist between network devices.
- End nodes are child to the Network Device resource.

Configuration Management 5.1 / Client Automation 7.x Scenarios

The table below shows the out-of-the-box scenarios that use the Configuration Management and Client Automation connector.

Configuration Management 5.1 / Client Automation Scenarios 7.x

Scenario	Description
<p>hpovcm\im75am52\imac.scn</p> <p>hpovcm\im7xam93\imac.scn</p>	<p>This scenario enables you to transfer data from an OpenView Configuration Management Inventory Manager database to an Asset Manager database. The following data is transferred:</p> <ul style="list-style-type: none"> • computers • logical hard drives • physical hard drives • network cards • internal devices <p>Reconciliation is now performed on the BarCode element instead of the FullName element. The sysComputer value for the Nature element was replaced with the CPU value.</p>
<p>hpovcm\im75am52\imac-swnorm.scn</p> <p>hpovcm\im7xam93\imac-swnorm.scn</p>	<p>This scenario enables you to transfer data from an OpenView Configuration Management Inventory Manager database to an Asset Manager database. This scenario is used for software normalization.</p> <p>Use a reconciliation key with a combination of the model and installation path when importing software installations to Asset Manager.</p>
<p>hpovcm\im75am52\imac-usage.scn</p> <p>hpovcm\im7xam93\imac-usage.scn</p>	<p>This scenario uses a reconciliation key with a combination of the model and installation path when importing software installations to Asset Manager. Before running this scenario, edit the SQL script file in the following directory: \hpovcm\im75am52\sqlscript.</p> <p>Modify the Client Automation database name and schema name, and then execute the script file in the corresponding database. This scenario prompts you to use \config\hpovcm\config\usage.cfg, which is located in the Connect-It installation folder as an extension file instead of the default value in "Define document types" for the Usage Manager connector.</p>
<p>hpovcm\im75am52\imac-se.scn</p> <p>hpovcm\im7xam93\imac-se.scn</p>	<p>This scenario enables you to transfer data from OpenView Configuration Management Inventory Manager and OpenView Configuration Management Service Events databases to an Asset Manager database. See Customizing an Asset Manager Database.</p> <p>Use a reconciliation key with a combination of the model and installation path when importing software installations to Asset Manager.</p>

Scenario	Description
<p>hpovcm\im51sm71\imsm.scn hpovcm\im75sm71\imsm.scn hpovcm\im7xsm92\imsm.scn hpovcm\im7xsmws92\imsm.scn</p>	<p>Before executing this scenario, you must import the pcsoft-uninst.unl and pcssoftware.unl files into ServiceCenter. These files are provided with the installation of Connect-It (datakit\sc folder of your installation folder). This scenario enables you to transfer data from an OpenView Configuration Management Inventory Manager database to an ServiceCenter database.</p> <p>When loading the scenario, a warning message is displayed. This message requests that you define the data to use in the mactable for the scenario.</p> <p>Before running this scenario, import the UNL from the datakit\sc\sm71 folder to the Service Manager server.</p>
<p>hpovcm\cm51ac51\ldap_active_directory_all.scn hpovcm\ca75am52\ldap_active_directory_all.scn hpovcm\ca7xam93\ldap_active_directory_all.scn</p>	<p>This scenario transfers data from Active Directory to Asset Manager portfolio item. The following data is transferred:</p> <ul style="list-style-type: none"> • user account • group • organization • domain • computer
<p>hpovcm\cm51ac51\ldap_directory_service_all.scn hpovcm\ca75am52\ldap_directory_service_all.scn hpovcm\ca7xam93\ldap_directory_service_all.scn</p>	<p>This scenario transfers person data from Directory Service to Asset Manager database.</p>
<p>hpovcm\cm51ac51\ws_groups_devices.scn hpovcm\ca75am52\ws_groups_devices.scn hpovcm\ca7xam93\ws_groups_devices.scn</p>	<p>This scenario transfers group data to Asset Manager database.</p>
<p>hpovcm\cm51ac51\ws_jobs.scn hpovcm\ca75am52\ws_jobs.scn hpovcm\ca7xam93\ws_jobs.scn</p>	<p>This scenario transfers and update work orders from Configure Manager/Client Automation to Asset Manager.</p>
<p>hpovcm\cm51ac51\ws_policies.scn</p>	<p>This scenario update policies from Configure Manager/Client Automation to Asset Manager.</p>

Scenario	Description
<p>hpovcm\ca75am52\ws_policies.scn</p> <p>hpovcm\ca7xam93\ws_policies.scn</p>	
<p>hpovcm\cm51ac51\ws_services.scn</p> <p>hpovcm\ca75am52\ws_services.scn</p> <p>hpovcm\ca7xam93\ws_services.scn</p>	<p>This scenario transfers and update software info from Configure Manager/Client Automation to Asset Manager.</p>
<p>hpovcm\cm51ac51\ws_status.scn</p> <p>hpovcm\ca75am52\ws_status.scn</p> <p>hpovcm\ca7xam93\ws_status.scn</p>	<p>This scenario transfers current and history tasks related data from Configure Manager/Client Automation to Asset Manager.</p>
<p>hpovcm\cm51sm71*.scn</p> <p>hpovcm\ca7xsm71*.scn</p> <p>hpovcm\ca7xsm92*.scn</p> <p>hpovcm\ca7xsmws92*.scn</p>	<p>Before running this scenario, import the UNL from the datakit\sc\sm71 folder to the Service Manager server.</p> <ul style="list-style-type: none"> • cmsm_connectivity_incident.scn • cmsm_device.scn • cmsm_group.scn • cmsm_policy_compliance.scn • cmsm_service.scn • ldapsm_device.scn • smcm_change_policy.scn • smcm_software_incident.scn

Notes for imac-usage.scn

Scenarios enable you to transfer data from an HP Client Automation Inventory Manager database and from a Usage Manager database to an Asset Manager database. This scenario enables you to transfer data involving computers and users to an Asset Manager database.

For a HP Client Automation Usage Manager table, each ComputerNames record represents a computer and a user. This means that if there are several users for the same computer, a Radia inventory will produce as many records as there are users for that computer. In Asset Manager, information involving application use is recorded in the Software installations and utilizations table and can be viewed via the Utilization tab. There may be more than one user for a given application (each user is recorded in the departments and employees table). This results in there being several

lines in the Software installations and utilizations table which represent use of the application by each of the different users. The filter that is used to reconcile behavioral differences between both applications is defined in the `amSoftInstall` structure:

```
seType=1
```

This filter is used to define an application as being used (the Installation type is Utilization).

Performance

In order to improve data processing, the following behaviors have been defined:

- If information concerning computer usage is not present in a HP Client Automation Usage Manager database and the computer does not exist in the Asset Manager database, it is ignored.
- Applications for which there is no information in Radia Usage Management are not inserted into Asset Manager.
- Advanced reconciliation is not implemented for the **AddOn** collection involving software. If advanced reconciliation is enabled, software found in an HP Client Automation Inventory Manager database but not found in a HP Client Automation User Manager database is set to type 6 "absent".
- The **rcaFilePropertyNames** filter, which only takes into account the "ProductVersion", "ProductName" and "CompanyName", is applied for the HP Client Automation Usage Manager connector's **ComputersName** produced document type.
- Indexes have been created for the HP Client Automation Usage Manager database:
 - Index for the **WindowsComputerUser_id** column in the **rcaWindowsFileUsage** table. For example, for an MS SQL Server database:

```
CREATE
INDEX [idxCIT_rcaWindowsFileUsage_WindowsComputerUser_id]
ON [dbo].[ rcaWindowsFileUsage] ([WindowsComputerUser_id]) ON
[PRIMARY]
```
 - Index for the **FileSignature_id** column in the **rcaFileSignatureProperties** table. For example, for an MS SQL Server database:

```
CREATE
INDEX [idxCIT_rcaFileSignatureProperties_FileSignatureId] ON
[dbo].[rcaFileSignatureProperties] ([FileSignature_id])
ON [PRIMARY]
```

Customizing the Asset Manager Database

The Asset Manager database must be customized in order for the out-of-the-box scenarios to operate correctly.

Caution: *We recommend that you make a backup copy of your Asset Manager database before customizing it.*

Changes to be made are as follows:

- Add a field to the **amSoftInstall** table to allow uninstallation dates to be inserted:
 - SQL name: `dtUninstalledByRadia`
 - Label: Uninstalled by Radia on

- Description: Uninstalled by Radia on
- Type: Date and time
- Keep history: Yes
- Add a field to the **amSoftInstall** table to allow installation dates to be inserted:
 - SQL name: dtInstalledByRadia
 - Label: Installed by Radia on
 - Description: Installed by Radia on
 - Type: Date and time
 - Keep history: Yes
- Add a field to the **amSoftInstall** table to indicate whether or not an application was installed by Radia:
 - SQL name:
 - deployedByRadia (for Asset Manager version 5.1x or earlier)
 - bdeployedByRadia (for Asset Manager version 5.20 or later)
 - Label: Is deployed by Radia
 - Description: Is deployed by Radia (1=yes, 0=no)
 - Type: Boolean
- Add a field to the **amSoftInstall** table to indicate the names of services that are used:
 - SQL name: RadiaService
 - Label: Radia services
 - Description: Radia services used to deploy the application (separated by commas)
 - Type: Text (size 150)
 - Create an index for this field: option selected
- Add a field to the **amSoftInstall** table to indicate the component in charge of the operation:
 - SQL name: RadiaCmpt
 - Label: Radia component
 - Description: Radia component used
 - Type: Text (size 50)

After the scenario executes, the information can be found on the Applications tab of a computer or on the software installations screen.

Scenario behavior

The out-of-the-box scenario is used to integrate inventory data completed with HP OpenView Service Events data into an Asset Manager database. To prevent unnecessary records from being inserted, the following rule is used:

- Any information related to a computer that is present in an inventory tool database (for example, Inventory Management) is mapped using the **Update or insert** in an Asset Manager database mode.

- Any information returned by the OpenView Configuration Management Service Events connector is mapped using the Update only in an Asset Manager database mode.
- If information related to a computer is present in only one of the two source databases (inventory tool or OpenView Configuration Management Service Events) and no record related to the computer is present in the Asset Manager database, then no record is inserted into the Asset Manager database.
- If information related to a computer is present in only one of the two source databases (inventory tool or OpenView Configuration Management Service Events) and a record related to the computer is present in the Asset Manager database, then the record is updated using the new information from the source database.

Data reconciliation

A common model must be defined in Asset Manager in order to reconcile inventory data with data from a OpenView Configuration Management Service Events connector. The common model is used to correctly identify and insert information related to a computer returned by an inventory or a OpenView Configuration Management Service Events connector into an Asset Manager database.

The following naming convention has been defined to enable data reconciliation:

- for an DDMI / AC or HP OVCM IM / AC integration, the software installation model is: "Application Name" + " " + "Version".
- for HP OVCM tools, the **Short Description** field of the HP OVCM deployment application (**app_name** field in the **AppEvent** table) must use the following naming convention: "Application Name" + " " + "Version" determined by the inventory tool. In the integration scenario, the **app_name** field is mapped to the **Model Name** field.

If the naming convention is not applied when creating deployments using the HP OVCM tool, reconciliation will not be performed.

Reconciliation situations

Three situations involving reconciliation exist:

- A deployment using HP OVCM deployment tools corresponds to a single software installation in Asset Manager. The deployment's Short Description is used to reconcile using the Asset Manager model name.
- A deployment using HP OVCM deployment tools corresponds to multiple software installations in Asset Manager.

This situation may arise when the inventory tool that is used inventories each of the applications in a software suite. For example, the inventory tool counts each application that is included in the "Microsoft Office 2003" suite: "Excel 2003", "Word 2003", "PowerPoint 2003", etc., instead of referencing the "Microsoft Office 2003" suite itself. For the HP OVCM deployment tool, deployment of these applications corresponds to a single deployment called "Microsoft Office 2003".

In this case, the RadiaService<Inventory tool>Application mactable (where <Inventory tool> corresponds to the inventory tool that is used, such as OpenView Configuration Management Inventory Manager) must be populated with the key that corresponds to the application deployed by the HP OVCM tools, and the values from the components list. For example, for the

"Microsoft Office 2003" key, you define the following values:

```
"Microsoft Office 2003" | "Excel 2003" | "Word 2003" | "PowerPoint
2003 "
```

This mactable is found in the `radia_product.mpt` file. Once the mactable has been defined, Connect-It reconciles the HP OVCM "Microsoft Office 2003" deployment with all software installations saved in Asset Manager. Next, Connect-It propagates the data from the HP OVCM database to each of the software installations in the suite.

- Multiple deployments using HP OVCM deployment tools correspond to a single software installation in Asset Manager. This situation occurs when the inventory tool only detects a software suite and not the applications included in that suite. For example, the inventory tool counts "Microsoft Office 2003" and not the installed applications ("Excel 2003", "Word 2003", "PowerPoint 2003", etc.). In the HP OVCM deployment tools, the number of used deployments can equal the number of applications.

In this case, the `RadiaService<Inventory tool>Application` mactable (where `<Inventory tool>` corresponds to the inventory tool that is used, such as OpenView Configuration Management Inventory Manager) must be populated with the key that corresponds to the application deployed by the HP OVCM tools, and the value of the software suite.

For example, for the value "Microsoft Office 2003", you define the following keys:

```
"PowerPoint 2003 | "Microsoft Office 2003"
"Word 2003" | "Microsoft Office 2003"
"Excel 2003" | "Microsoft Office 2003"
```

Next, Connect-It reconciles the "PowerPoint 2003", "Word 2003", and other deployments with the "Microsoft Office 2003" software installations that are found in Asset Manager. Next, Connect-It propagates the data from the HP OVCM database to the corresponding software installations.

The out-of-the-box, predefined mactable can be used when doing an integration with an existing Radia infrastructure (the Radia database is already populated with data). In this case, the predefined naming convention does not correspond to the existing one. In this case, update the mactable in `radia_product.mpt` using the keys that correspond to the Short Description field of the deployment and the corresponding values for an Asset Manager database.

HP DecisionCenter Connector Scenarios

The table below shows the out-of-the-box scenarios that use the HP DecisionCenter connector.

[Scenario type: Business scenario to be adapted](#)

HP DecisionCenter Connector Scenarios

Scenario	Description
<code>rds\rds52ac4\rdsac.scn</code>	This scenario enables you to prepare data from an Asset Manager database for use by BiPortal.

HP Network Node Manager i-series Web Service Scenarios

The table below shows the out-of-the-box scenarios that use the HP Network Node Manager i-series Web Service connector.

Scenario type: [Sample scenario](#)

HP Network Node Manager i-series Web Service Scenarios

Scenario	Description
nnmi\ nnmi81am52\ nnmiam.scn	<p>This scenario enables you to transfer NNMi data to Asset Manager, and uses three NNMi web service objects: Node, Interface, IPAddress. In the wsdl files below, change the localhost name to your NNMi web servers hostname.</p> <ul style="list-style-type: none"> • http://localhost/NodeBeanService/NodeBean?wsdl • http://localhost/InterfaceBeanService/InterfaceBean?wsdl • http://localhost/IPAddressBeanService/IPAddressBean?wsdl

The following table shows the mapping between NNMi and Asset Manager.

Mapping between NNMi and Asset Manager

Destination	Mapping / Source	Comment
amComputerDst	getNodeSrc1	Create Computer
getInterfacesDst	getNodeSrc1	Get Interface by NodeID
getIPAddressesDst	getNodeSrc1	Get IP by NodeID
amComputerDst1	getIPAddressesSrc	Update Computer IP and Network Card IP
amComputerDst2	getIPInterfacesSrc	Create Network Cards

HP Operations Manager Scenarios

The table below shows the out-of-the-box scenarios that use the HP Network Node Manager i-series Web Service connector.

HP Operations Manager Scenarios

Scenario	Description
ovou82ac50\ovouac.scn	<p>This scenario lets you retrieve data related to the servers (nodes) that are stored in HP Operations and insert them in HP Asset Manager software.</p>
ovow75ac50\ovowac.scn hpovo\ovow80ac50\ovowac.scn hpovo\ovow80ac51\ovowac.scn	<p>This scenario enables you to:</p> <ul style="list-style-type: none"> • Export the list of nodes via WMI • Insert the list of nodes in Asset Manager

Scenario	Description
	<p>This scenario lets you retrieve data related to the servers (nodes) that are stored in HP Operations and insert them in Asset Manager.</p> <p>This scenario operates in the following manner: A Visual Basic script retrieves the list of servers (nodes) via WMI. Once the list of servers has been retrieved, it is saved as a tab-delimited text file. Only one parameter is accepted by the script: The name of the file containing the data to export.</p>

The node fields that are retrieved by the Delimited Text connector via WMI are the following:

WMI field	Column for the delimited text file	Comment
Caption	Caption	Node description
PrimaryNodeName	PrimaryNodeName	Hostname (may be suffixed with the domain name)
OSType	OSType	OSType returns a number in WMI, the corresponding text value is determined using a script
OSVersion	OSVersion	OS version/build
CommunicationPath	CommunicationPath	Generally, the hostname + domain or an IP address
Name	Name	Identifier in OVO Windows
SystemType	SystemType	SystemType returns a number in WMI, the corresponding text value is determined using a script

Analyzing the file generated by the script

The file generated by the script is a text delimited file type. This file is analyzed by the Connect-It scenario in order to insert data into an application. A .dsc file is provided in order to analyze the file. The document exposed by the Delimited Text connector is called a node.

WMI field	Column for the delimited text file	Comment
Caption	Caption	Node description
PrimaryNodeName	PrimaryNodeName	Hostname (may be suffixed with the domain name)
OSType	OSType	OSType returns a number in WMI, the

WMI field	Column for the delimited text file	Comment
		corresponding text value is determined using a script corresponding text value is determined using a script corresponding text value is determined using a script
OSVersion	OSVersion	OS version/build
CommunicationPath	CommunicationPath	Generally, the hostname + domain or an IP address
Name	Name	Identifier in OVO Windows
SystemType	SystemType	SystemType returns a number in WMI, the corresponding text value is determined using a script

Scenario	Description
ovou82sc62/ovosc.scn	This scenario lets you retrieve data related to the servers (nodes) that are stored in HP Operations and insert them in ServiceCenter.
ovow75sc62/ovowsc.scn	This scenario enables you to: <ul style="list-style-type: none"> • Export the list of nodes via WMI • Insert the list of nodes in ServiceCenter <p>Important: The scenario must be located on the OVO Windows server in order to operate correctly.</p>
ovou8xsc6x-nodebank/sc-nb.scn ovou8xsc6x-nodebank/sc-nb-clean.scn ovou8xsc6x-outage/sc-outage.scn ovou8xsm7x-nodebank/sm-nb-clean.scn ovou8xsm7x-nodebank/sm-nb.scn ovou8xsm7x-outage/sm-outage.scn	This scenarios are documented separately. Please refer to the SC_OVO-U_NodeBank_Outage_Integration_Guide.pdf guide.

HP Project and Portfolio Management Center Scenarios

These scenarios are documented separately. Please read the following document for details: <HP Asset Manager installation folder>\doc\pdf\Portfolio-EN.pdf.

HP Service Desk (Outbound) Connector Scenarios

These scenarios are documented separately. Please consult the SDAC_Integration_Userguide.

- sd45ac44\acsd\acsd_sync.scn
- sd45ac44\sdac\sdac_init.scn
- sd45ac44\sdac\sdac_sync.scn
- sd45ac50\acsd\acsd_sync.scn
- sd45ac50\sdac\sdac_init.scn
- sd45ac50\sdac\sdac_sync.scn
- sd51ac50\acsd\acsd_sync.scn
- sd51ac50\sdac\sdac_init.scn
- sd51ac50\sdac\sdac_sync.scn
- sd51ac51\acsd\acsd_sync.scn
- sd51ac51\sdac\sdac_init.scn
- sd51ac51\sdac\sdac_sync.scn

HP ServiceCenter and Service Manager Connector Scenarios

The table below shows the out-of-the-box scenarios that use the HP ServiceCenter and Service Manager connector. In addition, this connector can also use scenarios from the following list:

- [HP Discovery and Dependency Mapping Inventory scenarios](#)
- [Configuration Management 5.1 / Client Automation 7.x Scenarios](#)
- [LANDesk scenarios](#)
- [IBM WebSphere MQ scenarios](#)
- [HP DecisionCenter scenarios](#)
- [Tivoli scenarios](#)
- [TS Census scenarios](#)

Scenario type: Business scenario to be adapted

HP ServiceCenter and Service Manager Connector Scenarios

Scenario	Description
scac\sc60ac44\scac.scn scac\sc60ac44\acsc.scn scac\sc61ac50\acsc.scn scac\sc62ac50\acsc.scn	Before executing this scenario, you must import the file acsc.unl, which creates the required events on importing the data in ServiceCenter. The two scenarios (scac.scn and acsc.scn) replicate the inventory data and the associated data (contacts, locations, models, etc.) between a ServiceCenter 6.0 database and an Asset Manager 4.4 database.

Scenario	Description
	<p>We recommend against replicating data in both directions because of differences in the database schemas. We recommend managing (authorization to update, insert and/or delete) each functional category in one or the other applications and to enable only the corresponding mappings for each scenario. For example, updating Employees and Locations in Asset Manager, and updating inventory data in ServiceCenter, implies disabling the replication of Employees/Locations in the scac.scn scenario and disabling the replication of the inventory data in the acsc.scn scenario.</p>
<p>scac\sc61ac44\acsc-ci.sc scac\sc62ac51\acsc-ci.scn</p>	<p>This scenario enables the integration of Asset Manager Configuration Items into ServiceCenter.</p>
<p>scac\sc61ac44\acsc-incident.scn scac\sc62ac51\acsc-incident.scn</p>	<p>This scenario is used to create an incident in ServiceCenter when the reconciliation proposal is done in Asset Manager.</p>
<p>scac\sc61ac44\scac-ci.scn scac\sc62ac51\scac-ci.scn</p>	<p>This scenario enables the integration of ServiceCenter Configuration Items into Asset Manager.</p>
<p>scac\sc61ac44\scac-wo.scn scac\sc62ac51\scac-wo.scn</p>	<p>This scenario is used to create a work order in Asset Manager that corresponds to a change request in ServiceCenter.</p>
<p>scauto\scacfg\scacfg.scn</p>	<p>Mapped source: Output Event Type Mapped destination: InputEventTypes</p> <p>Mapped source: Input Event Type Mapped destination: OutputEventTypes</p> <p>Mapped source: EventTypes Mapped destination: SCAutoConfiguration</p>
<p>scauto\sca-sc\sca-sc.scn</p>	<p>This scenario enables you to create a configuration file (scautoconfiguration.xml) for the SCAuto listening connector. This file contains the definition of a ServiceCenter database's elements.</p>

Scenario	Description
sc\sc60mail\scincident-mail.scn sc\sc61mail\scincident-mail.scn sc\sc62mail\scincident-mail.scn sc\sm71mail\smincident-mail.scn* scac\sm92mail\scincident-mail.scn scac\smws92mail\smincident-mail.scn	This scenario sends an e-mail when an incident ticket is created. The data in the probsummary table is read and an e-mail is automatically sent to the assignee, using the e-mail address contained in the operator record. The e-mail contains the incident description and the file or files linked to the incident ticket are sent as attachments. This scenario is scheduled to be executed every hour from 8 AM through 6 PM, and inserts or updates the records.
scac\sm71ac51\amsm-ci.scn*	This scenario enables the integration of Asset Manager Configuration Items into Service Manager.
scac\sm71ac51\amsm-incident.scn*	This scenario is used to create an incident in Service Manager when the reconciliation proposal is done in Asset Manager.
scac\sm71ac51\smam-ci.scn*	This scenario enables the integration of Service Manager Configuration Items into Asset Manager.
scac\sm71ac51\smam-wo.scn*	This scenario is used to create a work order in Asset Manager that corresponds to a change request in Service Manager, before running these scenarios.

* It is necessary to import UNL files from the datakit\sc\sc62 folder into the SC server before running these scenarios.

HP ServiceCenter and Service Manager Web Service Scenarios

HP SCAuto Listening Connector Scenarios

The table below shows the out-of-the-box scenarios that use the HP SCAuto Listening connector.

Scenario type: Business scenario to be adapted

HP SCAuto Listening Connector Scenarios

Scenario	Description
scauto\scacfg\scacfg.scn	The following are the mapped source and destination document types: <ul style="list-style-type: none"> • Source: Output Event Type Destination: InputEventTypes • Source: Input Event Type Destination: OutputEventTypes

Scenario	Description
	<ul style="list-style-type: none"> Source: EventTypes Destination: SCAutoConfiguration
scauto\sca-sc\sca-sc.scn	For this scenario, you must use the <code>BasicScaCfg.xml</code> configuration file during the configuration of the scenario. This file is located in the same folder as the scenario.

HP Service Anywhere Connector Scenarios

The tables below show the out-of-the-box scenarios that use the HP Service Anywhere connector.

Email to Service Anywhere scenarios

Scenario type: [Business scenario to be adapted](#)

Scenario	Description
saw\mail\saw\mail\saw.scn	This scenario creates incidents via incoming emails and writes results in a text file. It uses <code>IncidentManagement.wsdl</code> in Service Anywhere.
saw\mail\saw\result.dsc	This is a description file used by the text connector in this scenario.

Service Anywhere to Email scenarios

Scenario type: [Sample scenario](#)

Scenario	Description
samples\saw\sawmail\sawmail.scn	This scenario retrieves incident tickets from Service Anywhere and sends emails to the owners .
samples\saw\sawmail\date.dsc	This is a description file used by the text connector in this scenario.
samples\saw\sawmail\date.txt	This file contains date filter value. The <code>sawmail.scn</code> scenario retrieves incident tickets which last modified date is later than the value in this file.

LDAP to Service Anywhere scenarios

Scenario type: [Sample scenario](#)

File name	Description
samples\saw\ldapsaw\FunctionalGroup.scn	This scenario retrieves FunctionGroup, and relationship of approver-FunctionGroup and member-FunctionGroup, from LDAP and creates/updates into Service Anywhere.

File name	Description
samples\saw\ldapsaw\Location.scn	This scenario retrieves location, and relationship of location-organization, from LDAP and creates/updates into Service Anywhere.
samples\saw\ldapsaw\Organization.scn	This scenario retrieves organization, and relationship of organization-location, from LDAP and creates/updates into Service Anywhere.
samples\saw\ldapsaw\Person.scn	This scenario creates/updates the relationships of member-organization, from LDAP and creates/updates into Service Anywhere

These scenarios use `DataAdministration.wsdl` in Service Anywhere.

Universal CMDB to Service Anywhere scenarios

[Scenario type: Business scenario to be adapted](#)

Refer to `SAW_UCMDB_CI_Integration_Guide.PDF` in the <Connect-It installation folder>\doc folder.

HP ServiceCenter uCMDB Scenarios

These scenarios are documented separately. Please read the HP ServiceCenter 6.x to uCMDB 6.x Integration Overview, HP ServiceCenter 6.x to uCMDB 6.x Integration Deployment Guide, and ServiceCenter 6.x to uCMDB 6.x Integration Enhancement Guide.

Altiris Scenarios

The table below shows the out-of-the-box scenarios that use the Altiris connector.

Altiris Scenarios

Scenario	Description
altiris\altiris61ac44\altirisac.scn	This scenario enables you to transfer data from an Altiris database to an Asset Manager database. The following data is transferred: <ul style="list-style-type: none"> • computers • installed software • monitors • installed cards • logical hard drives
altiris\altiris61ac50\altirisac.scn	
altiris\altiris61ac51\altirisac.scn	
altiris\altiris7am93\altirisam.scn	

Scenario	Description
	<ul style="list-style-type: none"> physical hard drives network printers
altiris\altiris61ac44\altirisac_ swnorm.scn altiris\altiris61ac50\altirisac_ swnorm.scn altiris\altiris61ac51\altirisac_ swnorm.scn altiris\altiris7am93\altirisam_ swnorm.scn	This scenario enables you to transfer data from an Altiris database to an Asset Manager database. This scenario is used for software normalization.

Aperture Vista Scenarios

This section briefly describes the out-of-the-box scenarios that use the Aperture Vista connector. The scenarios are documented separately. Please read the following document for details: <HP Asset Manager installation folder>\doc\pdf\IntegrationWithDCIM-EN.pdf.

CA Unicenter DSM Scenarios

The table below shows the out-of-the-box scenarios that use the CA Unicenter DSM connector.

Scenario type: Business scenario to be adapted

CA Unicenter DSM Scenarios

Scenario	Description
amo\dsm11ac44\dsmac.scn amo\dsm11ac50\dsmac.scn amo\dsm11ac51\dsmac.scn	This scenario enables you to transfer data from a Unicenter DSM 11 database to an Asset Manager application. The following data is transferred: <ul style="list-style-type: none"> Computers Software Associated devices

LANDesk Scenarios

The table below shows the out-of-the-box scenarios that use the LANDesk connectors.

Scenario type: Business scenario to be adapted

LANDesk Scenarios

Scenario	Description
ldsk\ld8ac44\ldskac.scn ldsk\ld8ac50\ldskac.scn	This scenario enables you to transfer data from a LANDesk database to an Asset Manager

Scenario	Description
ldesk\ld8ac51\ldskac.scn ldsk\ld9am93\ldskam.scn	database. <ul style="list-style-type: none"> • Computers • Devices • Software
ldsk\ld8sc6\ldsksc.scn ldsk\ld8sc61\ldsksc.scn ldsk\ld8sc62\ldsksc.scn ldesk\ld8sm71\ldsksm.scn ldsk\ld9sm9\ldsksm.scn	Before executing this scenario, you must import the pcsoft-uninst.unl and pcsoftware.unl files into ServiceCenter. These files are provided with the Connect-It installation (datakit\sc folder of your installation folder). This scenario enables you to update the information relating to the computers (and their software) in a ServiceCenter/ Service Manger database using data obtained with LANDesk.
ldskws\ld8ac50\LDStartTasks.scn ldskws\ld8ac51\LDStartTasks.scn	This scenario updates the following tables depending on the LANDesk information that is received: <ul style="list-style-type: none"> • amESDPackage • amESDDelivMethod
ldskws\ld8ac50\LDUpdateRepository.scn ldskws\ld8ac51\LDUpdateRepository.scn	This scenario creates scheduled tasks in LANDesk Software Distribution from information that is in the Asset Manager amESDTask scheduled tasks table.
ldskws\ld8ac50\LDUpdateTask.scn ldskws\ld8ac51\LDUpdateTask.scn	This scenario updates scheduled tasks in LANDesk Software Distribution from information that is in the Asset Manager amESDTask scheduled tasks table.

Microsoft SCCM Scenarios

The table below shows the out-of-the-box scenarios that use the Microsoft SMS 2003 and SCCM connectors. Note that the scenarios only support SCCM 2007 for now.

Scenario type: Business scenario to be adapted

Microsoft SCCM Scenarios

Scenario	Description
sms\sms2003ac44\smsac.scn sms\sms2003ac50\smsac.scn sms\sms2003ac51\smsac.scn	This scenario enables you to transfer information from an SMS database (IT portfolio) to the amComputer table of an Asset Manager 4.4 database.
sms\sms2003ac51\smsac-swnorm.scn	This scenario transfers software normalize data from a SMS2003 database to an Asset Manager

Scenario	Description
	5.10 database(amComputer).
sms\sms2003sc6\smssc.scn sms\sms2003sc61\smssc.scn ms\sms2003sc62\smssc.scn	Before executing this scenario, you must import the pcsoft-uninst.unl and pcsoftware.unl files into ServiceCenter. These files are provided with the Connect-It installation (datakit\sc folder of your installation folder). This scenario enables you to update the information relating to the computers (and their software) in a ServiceCenter 6.0 database using data obtained with SMS2003.
sms\sccm2007ac44\sccmac.scn sms\sccm2007ac50\sccmac.scn sms\sccm2007ac51\sccmac.scn sms\sccm2007sc60\sccmac.scn sms\sccm2007sc61\sccmac.scn sms\sccm2007sc62\sccmac.scn	This scenario provides for the transfer of IT portfolio data from an SCCM database to an Asset Manager 4.4 database table of computers (amComputer).
sms\sccm2007ac51\sccmac-swnorm.scn sms\sccm2007ac44\sccmac-swnorm.scn sms\sccm2007ac50\sccmac-swnorm.scn	This scenario transfers software normalize data from a SCCM2007 database to an Asset Manager 5.10 database(amComputer).
sms\sccm2007am93\sccmachw. scn sms\sccm2007am93\sccmacsw. scn sms\sccm2007am93\sccmacswnorm.scn	This scenario transfers software normalize data from a SCCM 2007 database to an Asset Manager 9.31 database (amComputer).
sms\sccm2007sm7\sccmsm.scn sms\sccm2007sm71\sccmsm71.scn* sms\sccm2007sm92\sccmsm.scn* * Before running these scenarios, import the UNL from datakit\sc\sm71 to the Service Manager server.	Port 12760 is used for connector output. Ports 12690 and 13080 are used by default for the SM connector.

TS Census Scenarios

The table below shows the out-of-the-box scenarios that use the TS Census connectors.

Scenario type: Business scenario to be adapted

TS Census Scenarios

Scenario	Description
tsc\tsc3ac44\tscac.scn	These scenarios enables you to transfer data from a

Scenario	Description
tsc\tsc3ac50\tscac.scn	TSCensus database to an Asset Manager database (for the relevant version specified in the scenario name).
tsc\tsc3sc6\tscsc.scn tsc\tsc3sc61\tscsc.scn tsc\tsc3sc62\tscsc.scn	Before executing this scenario, you must import the pcsoft-uninst.unl and pcsoftware.unl files into ServiceCenter. These files are provided with the Connect-It installation (datakit\sc folder of your installation folder). This scenario enables you to update the information relating to the computers (and their software) in a ServiceCenter 6.0 database using data obtained with TS Census.

Tivoli Scenarios

The table below shows the out-of-the-box scenarios that use the Tivoli connectors.

Scenario type: Business scenario to be adapted

Tivoli Scenarios

Scenario	Description
tivoli\tim\tim4ac44\timac.scn tivoli\tim\tim4ac50\timac.scn	These scenarios enable you to transfer data from a Tivoli Inventory Management 4.0 database to an Asset Manager database. The following data is transferred: <ul style="list-style-type: none"> • Computers • Software
tivoli\tcm\tcm42sc60\tcmsc.scn tivoli\tcm\tcm42sc61\tcmsc.scn tivoli\tcm\tcm42sc62\tcmsc.scn	Before executing these scenarios, you must import the pcsoft-uninst.unl and pcsoftware.unl files into ServiceCenter. These files are provided with the Connect-It installation (datakit\sc folder of your installation folder). This scenario enables you to transfer data from a Tivoli Inventory Management 4.2 database to an ServiceCenter 6.0 application. The following data is transferred: <ul style="list-style-type: none"> • Computers • Software
tivoli\tcm\tcm42ac44\tcmac.scn tivoli\tcm\tcm42ac50\tcmac.scn	These scenarios enable you to transfer data from a Tivoli Configuration Manager 4.2 database to an Asset Manager database. The following data is transferred: <ul style="list-style-type: none"> • Computers • Software
tivoli\tcm\tcm42ac44\tcmac-swnorm.scn	These scenarios are dedicated to users of the Software Licenses module for Asset Manager and allows to reconcile software models.

Scenario	Description
tivoli\tcm\tcm42ac50\tcmac-swnorm.scn	
tivoli\tcm_sd\tcmsd42ac44 tivoli\tcm_sd\tcmsd421ac44 tivoli\tcm_sd\tcmsd421ac50	The scenarios contained in this folder are documented separately.

Action Request System Connector Scenarios

The table below shows the out-of-the-box scenarios that use the Action Request System connector.

Action Request System Connector Scenarios

Scenario	Description				
ars\ars7am52\sharedat.scn Scenario type: Sample scenario	This scenario enables you to transfer data from Action Request System to an Asset Manager 5.2 database. The data items transferred by this scenario is Sample:Enrollments . <table border="1" data-bbox="636 919 1576 1066"> <thead> <tr> <th>Source</th> <th>Destination</th> </tr> </thead> <tbody> <tr> <td>Sample:Enrollments(Sample:EnrollmentsSrc)</td> <td>amEmpIDept(amEmpIDeptDst)</td> </tr> </tbody> </table>	Source	Destination	Sample:Enrollments(Sample:EnrollmentsSrc)	amEmpIDept(amEmpIDeptDst)
Source	Destination				
Sample:Enrollments(Sample:EnrollmentsSrc)	amEmpIDept(amEmpIDeptDst)				

HP Network Node Manager Scenarios

The table below shows the out-of-the-box scenarios that use the HP Network Node Manager connector.

[Scenario type: Sample scenario](#)

HP Network Node Manager Scenarios

Scenario	Description
nnm\nnm75ac50\nnmac.scn nnm\nnm75ac51\nnmac.scn	See the notes below .
nnm\nnm75sc62\nnmsc.scn nnm\nnm75sm71\nnmsm.scn* * Before running these scenarios, import the UNL from the datakit\scsm71 to the Service Manager server.	This scenario lets you transfer node type data from a NNM application to a ServiceCenter application.

Notes for nnm\nnm75ac50\nnmac.scn and nnm\nnm75ac51\nnmac.scn

Prerequisites

- The integration scenario must run on the same server as the NNM program
- Python 2.5 must be installed (and python.exe must be declared in the environment variables)

How the scenario works

This scenario uses the ovtodump command line tool. The l (long) and r (recursive) options are both used with ovtodump in order to retrieve a detailed list of NNM objects and to specify that the tool must cover all node type NNM branches. The script is called via a Python script in both Windows and Unix environments. The Python script writes files in xml format. The xml files are then processed by Connect-It.

The script's logic is as follows:

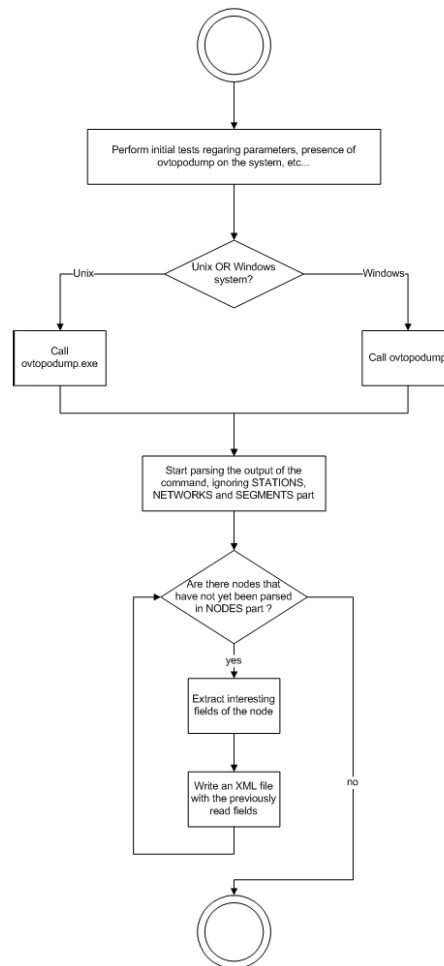


Figure 10-1: Python script logic

The scenario is divided into two parts:

- The Python script is called via the command line connector
- The mapping between the xml files and Asset Manager. The xml files are processed by the XML connector.

Mapping Node -> amComputer

Destination	Mapping / Source	Comment
Name	[label]	
ComputerDesc	[description]	
lpxSpxAddress	<pre>If [IPXAddress] = "00000000 0:0000000000000" Then PifIgnoreNodeMapping Else RetVal = [IPXAddress] End If</pre>	
lpxSpxServer	[IPXServerName]	
TcplpHostName	[hostname]	
TcplpAddress	<pre>If [interfaces.interface(0).IPAddr] <> "" Then RetVal = [interfaces.inter face(0).IPAddr] Else PifIgnoreNodeMapping End If</pre>	
PhysicalAddress	<pre>If [interfaces.interface(0).physicalAddress] <> "" Then RetVal = RightPart([interf aces.interface(0).physical Address], "x", 1) Else PifIgnoreNodeMapping End If</pre>	
Portfolio		
Folder [UUID]		
Folder Portfolio.Asset		
ExternalAssetID [ID]		
Name	RetVal = PifStrVal("MODEL_ UNKNOWN_ COMPUTER_MODEL")	
Portfolio.Asset.Model.Nature		
Code	"CPU"	
NetworkCards	<pre>If [interfaces.interface.p [interfaces.interface.physicalAddress] = "<none>" Then Ignore the current interface</pre>	[interfaces.interface]

Destination	Mapping / Source	Comment
	End If	
Description	[interfaces.interface.description]	If the description is empty, the record is ignored.
PhysAddress	RetVal = RightPart([interf aces.interface.physicalAdd ress], "x", 1)	
TcplpAddress	[interfaces.interface.IPAddr]	
SubnetMask	[interfaces.interface.IPMask]	

Note: Fields *TcplpHostName* and *PhysAddress* have a reconciliation key.

IBM Websphere MQ Connector Scenarios

The table below shows the out-of-the-box scenarios that use the IBM Websphere MQ connector.

Scenario type: [Standard business scenario](#)

IBM Websphere MQ Connector Scenarios

Scenario	Description
mqseries\ac44mq\ mqtoprgn.scn	<p>This scenario enables you to create, from IBM WebSphere MQ messages, records in the following tables of an Asset Manager application:</p> <ul style="list-style-type: none"> • Cost centers table (amCostCenter) • Companies table (amCompany) • Purchase orders table (amPOrder) • Receiving slips table (amDeliv) <p>The following table lists the mapped source and destination document types.</p> <p>Source: CostCenterSrc Destination: amCostCenterDst</p> <p>Source: VendorSrc Destination: amCompanyDst</p> <p>Source: ExtPOAckSrc Destination: amPOrderDst</p> <p>Source: ExtReceiptAckSrc Destination: amDelivDst</p>
mqseries\ac44mq\ prgntomq.scn	This scenario enables you to:

Scenario	Description
	<ul style="list-style-type: none"> • Create IBM WebSphere MQ messages from records in the following tables of an Asset Manager application: • Purchase orders table (amPOrder) • Receiving slips table (amDeliv) • Update these same tables from the PEREGRINE.OUT queue and the PEREGRINE.IN queue <p>Source: amPOrderSrc Destination: PO</p> <p>Source: amDeliv Destination: Receipt</p>
<p>mqseries\sc6mq\mqsc.scn mqseries\sc61mq\mqsc.scn mqseries\sm71mq\mqsm.scn* mq\sm71mq\mqsm.scn*</p> <p>* Before running these scenarios, import the UNL from the datakit\sc\sm71\ folder to the Service Manager server.</p>	<p>This scenario enables you to:</p> <ul style="list-style-type: none"> • Create records in ServiceCenter's contact file from the IBM WebSphere MQ messages in the PEREGRINE.IN queue. • Create IBM WebSphere MQ messages in the PEREGRINE.OUT.PERSON queue from records in ServiceCenter's contact file. <p>The following table lists the mapped source and destination document types.</p> <p>External contacts: Contacts from Exterior</p> <p>ServiceCenter contacts: Contacts from ServiceCenter</p>
<p>mq\acmq\acmq.scn mq\acmq\mqac.scn</p>	<p>These scenarios have the same function as mqseries\lac44\prgntomq.scn. The differences between these two scenarios are:</p> <ul style="list-style-type: none"> • This scenario uses the JMS for IBM WebSphere MQ connector instead of the IBM WebSphere MQ connector • This scenario replaces amDeliv with amReceipt in Asset Manager as receiving slips • This scenario uses [JMSType] to filter different message schema in the same message queue • This scenario uses mq.xsd instead of mqam.dtd as message schema file.
<p>mq\mqworkflow\CreateAndStartWorkflowInstance.scn mq\mqworkflow\ReceiveAndReplyWorkflowRequest.scn</p>	<p>These scenarios enable you to create and start workflows between IBM WebSphere MQ and Asset Manager. It is updated from mqworkflow\ac\CreateAndStartWorkflowInstance.scn</p>

Scenario	Description
	<p>.</p> <p>The differences between these two scenarios are:</p> <ul style="list-style-type: none"> • This scenario uses the JMS for IBM WebSphere MQ connector instead of the IBM WebSphere MQ connector • This scenario uses mqw.xsd instead of mqw.dtd

Lotus Notes Connector Scenarios

The table below shows the out-of-the-box scenarios that use the Lotus Notes connector.

[Scenario type: Sample scenario](#)

Lotus Notes Connector Scenarios

Scenario	Description				
ldap\ldapnote\names.scn	<p>This scenario enables you to transfer data from an LDAP directory to a Lotus Notes database. The following table lists the mapped source and destination document types.</p> <table border="1"> <thead> <tr> <th>Source</th> <th>Destination</th> </tr> </thead> <tbody> <tr> <td>organizationalPersonSrc1</td> <td>PersonDst</td> </tr> </tbody> </table>	Source	Destination	organizationalPersonSrc1	PersonDst
Source	Destination				
organizationalPersonSrc1	PersonDst				

NT Security Connector Scenarios

The table below shows the out-of-the-box scenarios that use the NT Security connector.

[Scenario type: Standard business scenario](#)

NT Security Connector Scenarios

Scenario	Description
ntsec\ntac44\addcpu.scn ntsec\ntac50\addcpu.scn ntsec\ntac51\addcpu.scn	<p>This scenario enables you to import NT information on computers in NT domains specified in Asset Manager 4.4 database.</p>
ntsec\ntac44\adduser.scn ntsec\ntac50\adduser.scn ntsec\ntac51\adduser.scn	<p>This scenario enables you to import NT information obtained from departments and employees to NT domains specified in an Asset Manager 4.4 database.</p>

SAP BAPI and Web Service scenarios

The table below shows the out-of-the-box scenarios that use the SAP BAPI and Web Service connector.

[Scenario type: Business scenario to be adapted](#)

SAP BAPI and Web Service scenarios

Scenario	Description
<p>sap\sapac44\purchaseprocess\BAPI_CREATE_REQUEST.scn</p> <p>ap\sapac50\purchaseprocess\BAPI_CREATE_REQUEST.scn</p> <p>sap\sapac51\purchaseprocess\WS_CREATE_REQUEST.scn</p> <p>sap\sapac51\purchaseprocess\BAPI_CREATE_REQUEST.scn</p>	<p>This scenario enables you to transfer data from an SAP database to an Asset Manager database. The data items transferred by this scenario are the following:</p> <ul style="list-style-type: none"> • Requests • Suppliers • Employees <p>The following lists the mapped source and destination document types:</p> <p>Source: amPOrder Destination: NewRequestDst</p>
<p>sap\sapac44\purchaseprocess\BAPI_CREATE_PO.scn</p> <p>sap\sapac50\purchaseprocess\BAPI_CREATE_PO.scn</p> <p>sap\sapac51\purchaseprocess\WS_CREATE_PO.scn</p> <p>sap\sapac51\purchaseprocess\BAPI_CREATE_PO.scn</p>	<p>This scenario enables you to transfer data from an SAP database to an Asset Manager database and vice-versa. The data items transferred by this scenario are the following:</p> <ul style="list-style-type: none"> • Purchase orders • Purchase order lines <p>The following lists the mapped source and destination document types:</p> <p>Source: amPOrderSrc Destination: NewPurchaseOrderDst</p> <p>Source: ProcessReportNewPurchaseOrderSrc Destination: amPOrderDst</p>
<p>sap\sapac44\purchaseprocess\BAPI_RECEIVE.scn</p> <p>sap\sapac50\purchaseprocess\BAPI_RECEIVE.scn</p> <p>sap\sapac51\purchaseprocess\WS_RECEIVE.scn</p> <p>sap\sapac51\purchaseprocess\BAPI_RECEIVE.scn</p>	<p>Manager database and vice-versa. The data items transferred by this scenario are the following:</p> <ul style="list-style-type: none"> • Receiving slips • Receiving lines <p>The following lists the mapped source and destination document types:</p> <p>Source: amReceiptSrc2 Destination: BAPI_GOODSMVT_CREATEDst2</p> <p>Source: ProcessReportBAPI_GOODSMVT_CREATES-rc1 Destination: amReceiptDst2</p>

SAP IDOC Scenarios

The table below shows the out-of-the-box scenarios that use the SAP IDOC connector.

Scenario type: [Business scenario to be adapted](#)

SAP IDOC Scenarios

Scenario	Description
sap\sapac44\masterdata sap\sapac50\masterdata sap\sapac51\masterdata	The following scenarios have the same functions as the scenarios for Asset Manager version 5.1. <ul style="list-style-type: none"> • IGetCompany.scn • IGetCostcenter.scn • IGetPeople.scn
sap\sapac44\purchaseprocess sap\sapac50\purchaseprocess sap\sapac51\purchaseprocess	The following scenarios have the same functions as the scenarios for Asset Manager version 5.1. <ul style="list-style-type: none"> • IGetOrderNbr.scn • IGetInvoice.scn

Email Connectors Scenarios

The table below shows the out-of-the-box scenarios that use the email connectors.

Email Connectors Scenarios

Scenario	Description
mail\mailac51\finconfi.scn	This scenario enables you to send a message corresponding to a purchase approval. The status of the request is changed in the Purchase Requests table in the Asset Manager application. <ul style="list-style-type: none"> • Source: InMailMessageSrc • Destination: amRequestDst
mail\mailac51\finreque	This scenario enables you to send a purchase request (corresponding to a record in the Requests table) to an e-mail address (that of the procurement manager). <ul style="list-style-type: none"> • Source: amRequestSrc • Destination: OutMailMessageDst
mail\mailac51\newemplo	This scenario enables you to create a record in the Departments and Employee table in the Asset Manager application from an e-mail message. <ul style="list-style-type: none"> • Source: InMailMessageSrc • Destination: amEmplDeptDst

HTTP Service Connector Scenarios

The table below shows the out-of-the-box scenarios that use the HTTP Service connector. For samples of using these scenarios, see [Appendix A](#).

[Scenario type: Sample scenario](#)

HTTP Service Connector Scenarios

Scenario	Description
<code>samples\httpservice\samples-db-server.scn</code>	This is a server scenario providing CRUD (creating, reading, updating and deleting) service mappings.
<code>samples\httpservice\samples-db-client-node.scn</code>	This scenario enables you to transfer data from or to an Asset Manager database (amComputer). <i>Note: This scenario must be used with the scenario <code>samples-db-server.scn</code>.</i>
<code>samples\httpservice\samples-db-client-soft.scn</code>	This scenario enables you to transfer data from or to an Asset Manager database (amSoftware). <i>Note: This scenario must be used with the scenario <code>samples-db-server.scn</code>.</i>
<code>samples\httpservice\samples-db-client-even.scn</code>	This scenarios enable you to trigger data transferring from an Asset Manager database (amAsset) to a text file. <i>Note: This scenario must be used with the scenario <code>samples-db-server.scn</code>.</i>
<code>samples\httpservice\scheduling-db-server.scn</code>	This is a server scenario providing mapping for reading data from an Asset Manager database.
<code>samples\httpservice\scheduling-db-client-delta.scn</code>	This scenario is scheduled to read documents that have not been processed since the last session from an Asset Manager database. <i>Note: This scenario must be used with the scenario <code>scheduling-db-server.scn</code>.</i>
<code>samples\httpservice\webservice-server.scn</code>	This is a server scenario for information query from the SOAP connector.
<code>samples\httpservice\webservice-client.scn</code>	This scenario enables you to query information from the SOAP connector through a HTTP request. <i>Note: This scenario must be used with the scenario <code>webservice-server.scn</code>.</i>

LDAP Connector Scenarios

The table below shows the out-of-the-box scenarios that use the LDAP connector.

[Scenario type: Sample scenario](#)

LDAP Connector Scenarios

Scenario	Description
<code>ldap\ldapac51\complete.scn</code>	This scenario enables you to transfer data from an LDAP directory to the Departments and Employees table in an Asset Manager application.

Scenario	Description
	<ul style="list-style-type: none"> • Source: inetOrgPersonSrc • Destination: amEmplDeptDst
ldap\ldapac51\simple.scn	<p>This scenario enables you to transfer data from an LDAP directory to the Departments and Employees table in an Asset Manager application.</p> <ul style="list-style-type: none"> • Source: organizationalPersonSrc • Destination: amEmplDeptDst
ldap\ldapnote\names.scn	<p>This scenario enables you to transfer data from an LDAP directory to a Lotus Notes database.</p> <ul style="list-style-type: none"> • Source: organizationalPersonSrc1 • Destination: PersonDst
ldapam93\sync_dn.scn	<p>This scenario enables you to synchronize identity information from an LDAP directory to the Departments and Employees table in an Asset Manager application.</p>

RESTful Client Connector Scenarios

The table below shows the out-of-the-box scenarios that use the RESTful Client connector. For samples of using these scenarios, see [Configuring the RESTful Client connector](#).

Scenario type: [Sample scenario](#)

RESTful Client Connector Scenarios

Scenario	Description
samples\rest-weather\learn-xsd_initialize.scn	This scenario generates an XML schema (XSD) file.
samples\rest-weather\airport-status.scn	This scenario enables you to request weather information from the FAA Airport Status REST service interface, and display the response in an XML file.

SOAP Scenarios

The table below shows the out-of-the-box scenarios that use the SOAP connectors.

Scenario type: [Sample scenario](#)

SOAP Scenarios

Scenario	Description
ws\wsac51\currency.scn	<p>This scenario enables you to update the exchange rate already defined in the amCurRate table of your Asset Manager application.</p> <p>The WSDL address of the Web service is:</p>

Scenario	Description
	<code>http://www.xmethods.net/sd/2001/CurrencyExchangeService.wsdl</code> .

Chapter 11

AQL Queries

Presentation

This section presents AQL and lists the places in the software where you can use queries.

AQL

AQL ("Advanced Query Language") is the query language used by Asset Manager to access the data in the Asset Manager database. It is comparable to SQL. AQL is automatically translated into SQL by the database engine when it's being used for queries.

Note: *We recommend that you have some knowledge of SQL and a good understanding of databases before using AQL.*

AQL is better suited than SQL for writing queries involving the Asset Manager database. This is due to the following reasons:

- **Independence from database**

The various database engines supported by Asset Manager all use differing versions of SQL, which are incompatible with each other. AQL is independent of the database engine used. Consequently, if you write AQL queries and migrate from one database engine to another, the queries will still work with the new database engine. For example, AQL uses the same set of functions, regardless of the database engine used. Thus, the Substring function in AQL is equivalent to Substr under Oracle for WorkGroupsSQL and Substring under Microsoft SQL Server SQL.

- **Generation of optimized SQL code**

AQL generates SQL code optimized according to the database engine used. This is especially visible when using indexes. For example, to search the full name of models by forcing the use of the indexes **Model ID(Model_ID)** and **Full name (FullName)** you will write the following AQL query:

```
SELECT FIRST_ROWS lModelId, FullName FROM amModel
```

The SQL code that is generated will be different depending on the target database engine and optimized according to this engine. The equivalent Oracle SQL code will be:

```
SELECT /*+ FIRST_ROWS INDEX_ASC(M1 Model_lModelId) */ M1.lModelId,  
M1.Full Name FROM amModel M1
```

The Microsoft SQL Server or Sybase SQL Server code will be:

```
SELECT M1.lModelId, M1.FullName FROM amModel M1 ORDER BY M1.lModelId
```

The IBM DB2 code will be:

```
SELECT lModelId, FullName FROM amModel OPTIMIZE FOR 100 ROWS
```

- **Simplified access to the Asset Manager database**

AQL simplifies the management of links and joins. This greatly facilitates access to the database when writing queries as compared to using SQL directly. In addition, AQL simplifies access to features, allowing you to use them as direct fields in their related tables. AQL also facilitates the utilization of calculated fields.

Specifics of AQL compared to SQL

AQL does not support DDL ("Data Definition Language") orders. AQL comprises extensions that simplify the handling of joins, features and calculated fields.

You should never write to the Asset Manager database directly using SQL statements.

Queries in Asset Manager

Queries enable you to combine criteria concerning information in a table or in linked tables. You can use queries to:

- Create filters on record lists. In this case, the queries are, in general, simple and based on the "Where" clause.
- Define views.
- Define export conditions in the export module.
- Create Crystal Reports.
- Create wizards.
- When you use Asset Manager APIs.
- If Asset Manager is used as a DDE server.

The queries are written in AQL ("Advanced Query Language"): It is an internal language used by Asset Manager to access the data in the Asset Manager database. Asset Manager has a query editor that helps write queries:

- Either indirectly by using the graphic interface.
- Or by writing the query directly in AQL.

Caution: *In order to be illustrative of the capabilities of AQL, the examples given later use all of the AQL syntax. The SELECT, WHERE, and FROM clauses, in particular, are explicitly shown. Certain functions, such as query filters (where the user only defines the WHERE clause in the AQL query) or the expression builder greatly simplify the creation of queries for the user (certain clauses are not visible). These examples cannot be used for these functions.*

Recommendations for Writing AQL Queries

We recommend reading this section before starting to write queries in AQL. This section deals with:

- Notation specific to AQL.
- Particularities specific to AQL and the Asset Manager database that condition the best way to write queries.

Caution: *SQL names of fields, links and tables of the Asset Manager database are used to write AQL queries. Refer to the Database.txt file describing the database structure for an exhaustive list of these names. This file is located in the following folder: [Asset Manager installation folder]/doc/infos*

Presentation of AQL joins.

A join consolidates multiple data tables in a single query.

AQL joins

The Asset Manager data model defines not only fields and tables, but it also defines the links

between tables. This enables you to automate the generation of join clauses in AQL. AQL links are expressed as:

```
Link[.Link[.Field]]
```

By simplifying the joins in such a manner, AQL also simplifies the creation of the majority of queries used for the database.

Example

The following query, written in AQL, returns for each model:

- Its **ID (lModelId)**.
- Its **Full name (FullName)**.
- The **Name** of the table linked to the brands (**amBrand**).

```
SELECT lModelId, FullName, Brand.Name FROM amModel
```

Here is the same query written in SQL Oracle or Microsoft SQL Server:

```
SELECT M1.lModelId, M1.FullName, B2.Name FROM amModel M1, amBrand B2
WHERE M1.lBrandId=B2.lBrandId
```

The two joins between the **Models** table and the **Brands** table (**amModel** and **amBrand**, respectively) are automatically generated in AQL. Using Asset Manager's interface query editor, you just need to click the hierarchic list on the fields of the selected table - or the fields of the linked tables - to generate the corresponding AQL code.

Note: *On all systems except Oracle and DB2, the number of outer joins is limited to 1. Under Microsoft SQL Server 7 and MSSQL 2000, you can modify the `amdb.ini` file to get round possible query-execution problems. Use the following instruction to modify this file in the detail of your connection:*

```
useSQL92Join=1
```

Reason for and usefulness of primary key 0 records

- **Primary key "0" records**

The data model of the Asset Manager database has certain particularities:

- The primary and foreign keys of each table are numeric (32-bit integer).
- A foreign key that does not point to a record is set to "0" (and not "NULL").
- Each table has an empty record whose primary key is set to "0".

- **Usefulness**

With these primary key "0" records, the results of a query using a non outer join between two given tables, A and B, can include records from table A which are not linked to any real record in table B (link not populated). These are records in table A, which are linked to the primary key "0" record in table B. Example: The following query, written in AQL, returns for each portfolio item's asset tag, the name of its user and its supervisor:

```
SELECT AssetTag, User.Name, Supervisor.Name FROM amPortfolio
```

A portfolio item that is not assigned to a user and/or supervisor appears in the results of the query. At database level, such an asset is linked to the primary key "0" record in the Departments and employees table.

- **Reason for these specificities**

This section explains why primary key "0" records are used, whereas a query using an external

SQL join can select records in table A that are not linked to a record in table B.

Primary key "0" records enable you to compensate for the fact that certain RDBMs do not handle multiple outer-joins: Using primary key "0" records, SQL queries generated from AQL queries do not use outer joins. Example: The AQL query below searches for each portfolio item, its asset tag and name of location of its user. The results will include assets that do not have a user and assets whose users do not have a location.

```
SELECT AssetTag, user.location.name FROM amPortfolio
```

If the generated SQL code used the outer joins of the DBMS, the SQL code generated for Sybase SQL Server would be:

```
SELECT a.AssetTag, l.name FROM amPortfolio a, amEmplDept e,
amLocation l WHERE a.lUserId *= e.lEmplDeptId AND e.lLocaId *=
l.lLocaId
```

This code is not supported by Sybase SQL Server since it uses several outer joins one after another. However, because there is a primary-key ("0") record in the Departments and Employees table and in the Locations table, you do not need to call on external SQL joins. Asset Manager therefore generates an SQL query using normal (non external) joins:

```
SELECT l.name FROM amPortfolio a, amEmplDept e, amLocation l WHERE
a.lUser Id = e.lEmplDeptId AND e.lLocaId = l.lLocaId
```

This query gives the expected results, since the **User** and **Location** links still point to a record in the table of departments and employees or locations (they point to the primary key "0" record if the link is not populated).

- **Consequences**

It is important to take these records into account in the queries that you write, especially when using aggregate functions. Example:

```
SELECT count(AssetTag) FROM amPortfolio
```

If you execute the above query, which counts the number of assets in the table of assets, the primary key "0" record is taken into account in the results. You therefore need to decrease the results by 1 in order to obtain the real number of assets in the database. u It is rarely necessary to have to generate outer joins at DBMS level. Note:

Note: *If you really want to manage outer joins at DBMS level, use the "=" and "*" AQL operators.*

Usage of NULL

Asset Manager uses the NULL value of the DBMS in only two instances:

- For an empty "text" type field.
- For a non populated "date" or "date+time" type field.

AQL allows you to use several equivalent syntaxes, as shown below. It converts them to the equivalent valid SQL code of the database engine. For empty "Text" type fields, you can use any of the following syntaxes, since the NULL value will be stored in the database:

WHERE <text field> = NULL

WHERE <text field> IS NULL

WHERE <text field> = "

For non-populated "date" or "date+time" type fields, you can use any of the following syntaxes, since the NULL value will be stored in the database:

WHERE <date or date+time field> = NULL

WHERE <date or date+time field> = IS NULL

WHERE <date or date+time field> = []

Note: When a "Numeric" type field is not populated (its value is NULL), Connect-It sets its value to "0". Similarly, the absence of a link is comes out as "Link = 0" or "foreign key = 0". Example: "Location=0" or "Locald=0".

Self

"Self" is an expression that is equivalent to the description string on the table to which it is applied. Using "Self" enables you to simplify the queries while taking into account the customizations made to the Asset Manager database.

Example: If the description string of the table of departments and employees is:

```
[Name], [FirstName], ([Phone])
```

The AQL query:

```
SELECT self FROM amEmplDept
```

Is equivalent to:

```
SELECT (((((Name + ',') + FirstName) + '(') + Phone) + ')') FROM  
amEmplDept
```

CurrentUser

"CurrentUser" enables you to write queries that depend on the person connected to the database. "CurrentUser" can be used as an expression, for example in a query, or as a link. You have to enter this expression manually as it is not offered by the query editor.

Used as "expression"

Example: We are looking for all the portfolio items used by the employee connected to the database.

```
SELECT lPortfolioItemId FROM amPortfolio WHERE User = CurrentUser
```

Used as "link"

"CurrentUser" may be considered as a link starting in all tables and pointing to the record in the table of departments and employees corresponding to the current user.

- With the form "CurrentUser", this function points to the record corresponding to the current user.
- With the form "CurrentUser.Field", this function returns the value of the field for the current user.

For example, when an action is triggered by the connected user, it is possible to contextually trigger another messaging type action, which automatically sends a warning message to the connected user. You just need to populate the detail of the action as follows:

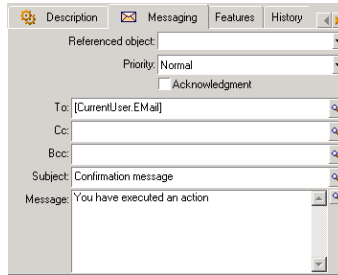


Figure 11-1: Configuring a messaging trigger

System itemized lists

If an AQL query uses a system itemized list, you must use the values that are stored in the database and not those which are displayed on screen. Example:
The following query selects those contracts whose Type field (SQL name: seType) is set to Master lease:

```
SELECT Self FROM amContract WHERE seType = 1
```

The Type field (SQL name: seType) is a system itemized list. The values stored in the database are:

- 0 for an **Other** type contract
- 1 for a **Master lease** type contract
- 2 for a **Lease schedule** type contract
- 3 for an **Insurance** type contract
- 4 for a **Maintenance** type contract

Note: The values of system itemized list can be found via Asset Manager Database Administrator, or the database.txt file, which describes the structure of the database. This file is located in the following folder: [Asset Manager installation folder]/doc/infos.

Hierarchic tables

All hierarchic tables contain:

- **"FullName" fields**
For each record in a hierarchic table, the "FullName" field stores the value of a field of the record, preceded by a tree structure constituted by the values of the fields of parent records until root level. Values are separated by the "/" character without spaces. This character also appears at the start and at the end of the tree-structure.

Examples:

- For the table of assets, the "FullName" field stores the Asset Tag of the asset preceded by the Asset Tag of its parent asset, that in turn preceded by the Asset Tag of its parent asset, and so on.

```
FullName = '/PC118/DD054/CR012/'
```

- In the table of locations, the "FullName" field stores the name of the location preceded by the names of parent locations.

```
FullName = '/Milwaukee/Water St. Site/Building A/5th floor/'
```

- **"sLvl" fields**

For each record in a hierarchic table, the "sLvl" indicates its level in the tree-structure. Root level is level 0.

The following query selects the "Sales Head Office" record and its sub-components:

```
SELECT Self FROM amEmplDept WHERE (FullName LIKE '/Sales Head Office/Sales/%') AND (sLvl >= 1)
```

The following query selects the "Sales Head Office" record but not its sub-components:

```
SELECT Self FROM amEmplDept WHERE (FullName LIKE '/Sales Head Office/Sales/%') AND (sLvl = 1)
```

The following query selects the sub-components of the "Sales Head Office" record but not the "Sales Head Office" record itself:

```
SELECT Self FROM amEmplDept WHERE (FullName LIKE '/Sales Head Office/Sales/%') AND (sLvl > 1)
```

Simplified AQL notation

This section lists the notation that can be used to simplify the writing of AQL queries:

- **Foreign keys**

In clauses other than the SELECT and ORDER BY clauses, the SQL name of a link that is not followed by a period is equivalent to the SQL name of the associated foreign key. Example: the clause:

```
WHERE location = 0
```

Is equivalent to:

```
WHERE lLocaId = 0
```

Where "location" is the SQL name of the "Location" link from the table of departments and employees to the table of locations, and "lLocaId" is the SQL name of the associated foreign key in the table of assets.

- **Description strings**

In SELECT and ORDER BY clauses, the SQL name of a link that is not followed by a period is equivalent to the join <SQL name of link >.self, which is in turn equivalent to <SQL name of link>.<Description string>.

Example: If the description string of the table of departments and employees is:

```
[Name], [FirstName] ([Phone])
```

The AQL query:

```
SELECT user FROM amPortfolio
```

Is equivalent to:

```
SELECT user.self FROM amPortfolio
```

That is itself equivalent to:

```
SELECT (((((User.Name + ',') + User.FirstName) + '(') + User.Phone) + ')') FROM amPortfolio
```

- **Features**

AQL gives you direct access to the features of a table, as if they were direct fields in the table.

To search feature values for a given table, you just need to write the SQL name of the feature (**fv_ prefix**). Example: The following query searches the values of the feature with SQL name **fv_WorkUnit** for the **Departments and Employees** table (**amEmpIDept**):

```
SELECT fv_WorkUnit FROM amEmpIDept
```

- **Calculated fields**

AQL facilitates the use of calculated fields associated with a table. You just need to write the SQL name of the calculated field (**cf_ prefix**).

AQL Sorts and Index

AQL allows two strategies for queries that use sorts (ORDER BY clause):

- A mode where Asset Manager forces the use of the indexes indicated in the query, when such indexes exist, and displays the results throughout the search.
- A mode where Asset Manager does not force the use of the indexes indicated in the queries. In this case, the database engine determines how the data is sorted.

Note: *The choice between these two different methods is not available under SQL Anywhere. The database engine automatically selects the most suitable method.*

Example

In the case of the following query:

```
SELECT lModelId, Brand FROM amModel ORDER BY Brand
```

- Access without **Forcing the indexes**: the database engine scans the table in full without using the "Brand" index indicated in the query. It searches all data items satisfying the query, sorts them according to the "Brand" and sends them to the user. The results will only be displayed after a certain period of time.
- In the other case: The database engine uses the "Brand" index, and displays the results as and when they are found. The first data items are thus shown rapidly, but the overall processing time is longer.

How to force the indexes

The way in which you force the utilization of the indexes depends on the way in which you create the query.

- **Via the Configure list menu**

You can configure the type of data access for each Asset Manager list, whether it be a main list or a tab list. To do this:

1. Go to the list you want to configure.
2. Right-click.
3. Select **Configure list** in the shortcut menu that appears.
4. In the Columns and sort tab, check the **Force indexes** box in order to use the indexes indicated in the query and display the results as and when they are generated. Uncheck this box in order to select the other type of access.

- **Using AQL**

If you write a query directly in AQL, you can force the use of indexes via the "FIRST_ROWS"

clause. Example:

```
SELECT FIRST_ROWS AssetTag FROM amAsset ORDER BY AssetTag
```

Note: *If the sort focuses on the system itemized lists (for example, on the **Features** table in the **seDataType** field), there is a possibility that the sort is not valid when an index is forced.*

Sort order

The sort order depends on:

- The database engine.
- The use of indexes.

Under Oracle for WorkGroups

- With indexes forced
- NULL records do not appear.

The sort is performed according to the value of the ASCII codes thus differentiating between upper and lower case characters (binary sort).

Without indexes forced

- NULL records appear.
- Oracle for WorkGroups is not case sensitive. Example Sort

For example:

Starting list **A B C D a b NULL NULL**

List with indexes forced **A B C D a b**

List without indexes forced **NULL NULL A a B b C D**

Microsoft SQL Server or Sybase SQL Server

The sort order depends on a parameter set when the database is created. These engines can be configured in order to be case sensitive or to take accented characters into account, etc.

Under Sybase SQL Anywhere

Under Sybase SQL Anywhere, the indexes cannot be forced via an AQL query. The database engine determines the optimal method used to access the data and to sort it.

Precautions

With complex queries it often difficult to determine whether it is more "advantageous" to force the indexes or not. In practice, we recommend performing tests before making a final decision. In particular, we recommend testing with and without the indexes forced in the case of a list that is filtered, be it explicitly (via a simple filter, query) or implicitly (via access restrictions).

The AQL Query Editor

A query editor is available in Asset Manager. It can be used to finalize, perfect and view queries. It is intended particularly for database administrators and advanced users.

How it works

The query editor makes it possible to design queries: n Either indirectly by using the graphic interface. n Or by writing the query directly in AQL. Whether you use the graphic method or prefer to

write directly in AQL (both approaches are frequently combined), you get to see a real-time transcription of your query in SQL. However, you cannot write your queries directly in SQL.



Figure 11-2: Query editor - composition modes

Using the query editor, a power user or an administrator can create, modify or delete AQL queries. These queries can be used in the appropriate context by their author or other users.

Accessing the query editor

You can access the query editor in different ways:

- From the **Tools > Queries** menu item. Using this menu, you can create queries for your own use, which can also be used freely by other users. The queries can be executed:
 - Either directly via the window displayed by the **Tools > Queries** menu item.
 - Or by when using a "query filter" when displaying the main table of the query.
- From numerous Asset Manager functions that rely on the queries: access restrictions, query filters, list configurations, purchase-request validations, tax formulas, etc.
- From external programs: Export component of the Asset Manager application, etc. The version of the query editor that is shown is simplified to a certain degree according to the context.

Example: Let's suppose that you have a query such as the following:

```
SELECT [FIRST_ROWS] <field>[, <field>...] FROM <table> [WHERE  
<clause>] [ORDER BY <clause>]
```

In the simplified versions of the query editor (simple filters, query filters, etc.), you only have to define the WHERE clause of the query. The other components of the query (starting table, fields, etc.) are implicit. For example, in the case of a query filter, the table is that on which the filter is applied, the fields and the sort conditions are the columns and the sort conditions which are defined via the **Configure list** shortcut menu item. The same is true for the query editor accessed via the **Tools > Queries** menu item.

Thus, the following query given in full:

```
SELECT self FROM amModel WHERE Brand.Name='Compaq'
```

is written as follows when used in a query filter (only the WHERE clause is explicitly given) used on the table of models:

```
Brand.Name='Compaq'
```

On the other hand, the **Configure list** command enables you to access a more comprehensive version of the query editor:

- The **Columns and sort** tab defines the fields to be displayed in columns and the sort conditions (these sort conditions correspond to the ORDER BY clause).
- The **Force indexes** box replaces the FIRST_ROWS clause in the SQL code.

- The **Filter (WHERE clause)** tab defines the "WHERE" clause.
- The table is implicit.

Creating a query using the query editor

To create a query using the query editor, select the **Tools > Queries** menu item. The window that is displayed has two tabs, **Filter (WHERE clause)** and **Preview**:

- The **Filter (WHERE clause)** tab is a graphical interface that determines the conditions of your query. It defines the elements of the WHERE clause.
- The **Preview** tab displays the transcription of your query in SQL code and enables you to test it.


Step 1: Populate the fields at the top of the query detail

You must specify the starting table of your query. If you want the query you are creating to be able to be accessible to other users, uncheck the **Not shared** option (SQL name: bPrivate).

Note: *The administrator has access to all queries stored in the database, even those queries that are **Not shared**. Once you have filled in the basic information on the query, click **Create** to be able to access the detail tabs of the query.*


Step 2: Define the filter conditions in the Filter (WHERE clause) tab

The Asset Manager query editor enables you to use criteria that involves fields, uses calculation expressions and integrates constants and operators. You can define one or more filter conditions. To define a filter condition:

1. From a start table, select a field, constant or expression (**Field 1**), that you will compare with a field, constant or expression (**Field 2**).
2. Confirm the filter condition by transferring it to the lower part of the screen using the  button.
3. Confirm the query by clicking **Modify** in the query detail.

To define several filter conditions linked by the AND and OR logical operators:


1. Create a first filter condition as indicated above.
2. Define the other filter conditions and confirm them using the **AND** or **OR** buttons.
3. Confirm the query by clicking **Modify** in the query detail.

Note: *In order to make a modification to the selected conditions, click the  button to delete the contents of the window, or else modify the AQL code directly.*

Note: *In place of the graphic tool, you can enter your query directly in AQL in the zone at the bottom of the **Filter (WHERE clause)** tab.*

Step 3: Preview the execution of the query

To test the query and see its transcription in SQL:

1. Go to the **Preview** tab in the query detail.
2. Click  : Asset Manager displays a view of the query results in the form of a list of records. The number of records returned by the query is listed in the lower right-hand corner of the window.


Note: *The SQL code contained in the **Preview** tab cannot be modified directly.*

Fields used in a query


When defining filter conditions in queries, you can call on:

- A field in the table concerned by the query.
- A linked field.
- Features associated with the table.

Writing an expression

Expressions  enable you to perform calculation operations in your query. For example, you can use the "Count" function to count the number of resultant records of a query. To write an expression, you can either:

- Enter it directly in the corresponding field.
- Or use the Asset Manager expression builder.

To use the expression builder, click the  button adjacent to the edit zone of the **Filter (WHERE clause)** tab of the query detail.

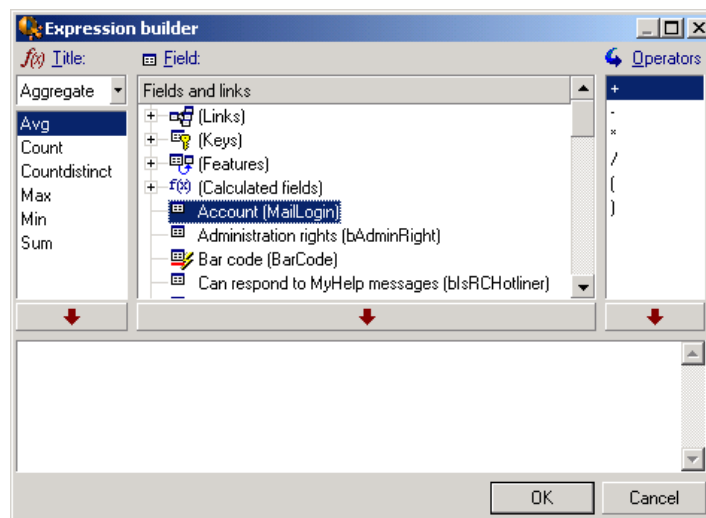



Figure 11-3: The AQL expression builder

The expression builder comprises three columns:

- The "Function" column lists existing AQL functions. Clicking applies a filter on the list of AQL functions according to their type: "Aggregate", "String", "Date", "Numeric", "Text".
- The "Field" column lists the different fields that can be used in a query.
- The "Operators" column lists the operators that can be used in the expression.

To insert a "Function", "Field" or "Operator" in the expression:


1. Select the function, field or operator.


2. Click . Once the expression is defined, click **OK** to transfer it over to the **Filter (WHERE clause)** tab in the query detail.

Constants

Constants are fixed values that you assign to selection criteria. For example, if you are searching all models with brand **3Com**, you assign the constant value **3Com** to the **Brand.Name** linked field in the table of models.

To select constant:

1. Click the  icon.
2. A selection window is displayed that shows the values present in the database for the field specified as search condition.

Note: Even in the case of "Itemized list" type fields, the window displayed by clicking the  icon only shows those values used in the database.

AQL Syntax

Using AQL requires familiarity with SQL. However, a detailed description of the syntax of SQL is beyond the scope of this manual. For further information, please consult the appropriate reference documentation.

Conventions

Conventions used to describe the syntax of AQL:

AQL Syntax conventions

Symbol	Description
[]	These brackets surround optional items. Do not type in these brackets.
< >	These brackets surround logical items. Do not type in these brackets.
	The vertical bar indicates that choices are mutually exclusive.
...	The ellipsis indicates that the preceding text may be repeated once or several times.
FROM	Terms in uppercase letters indicate literal expressions.

Syntax of queries

Simple queries

```
SELECT [DISTINCT]
[FIRST_ROWS] <selection list>
[FROM clause]
[WHERE clause]
[GROUP BY clause]
[HAVING clause]
[ORDER BY clause]
```

Sub queries

AQL supports the use of sub-queries in the place of fields.

Note: *In sub-queries, the SELECT statement only authorizes a single expression.*

```
(SELECT [DISTINCT] <expression>
[FROM clause]
[WHERE clause]
[GROUP BY clause]
[HAVING clause]
)
```

Note: *Sub-queries must be contained with parentheses.*

Example: `SELECT Self FROM amAsset WHERE mPrice >= (SELECT Max(mPrice)/2 FROM amAsset)`

UNION type queries

UNION enables you to group together the results of several queries:

```
SELECT <selection list>
[FROM clause]
[WHERE clause]
[GROUP BY clause]
[HAVING clause]
[ UNION | UNION ALL | INTERSECTS | MINUS] SELECT <selection list>
[FROM clause]
[WHERE clause]
[WHERE clause]
[GROUP BY clause]
[HAVING clause]...
[ORDER BY clause]
```

Elements of a query

Fields and links

The queries call into play fields and links of the Asset Manager database. You can indicate the name of a field:

- In reference to the starting table of the query. In this case, it is not necessary to specify the name of this table:

[Link. ...[Link.]]<field>

Examples from the **Portfolio items** table (**AmPortfolio**):

Model

User.Name

User.Location.Name

- As an absolute reference. In this case, you need to indicate the name of the table to which the field belongs:
 - Either by declaring the table in the **FROM** clause and using its name (or a possible alias):
`<table.[link...]field> <alias.[link...]field>`

- Or by not declaring the table in the **FROM** clause and instead using ":":
`<table.[link...]field> <table[_alias]:[link[_alias]...]field>`

These last two notations are particularly useful if you cannot use the **FROM** clause. For example, when you write a query in Asset Manager, you only have access to the **WHERE** clause. The query's starting table is implicit (table to which the filter applies, **Table** field (**TableName**) of the query's detail, etc.). However, you might need to use other tables in the query. In this case, the ":" character enables you to do this.

Constants

The following syntax is valid for the constants that may be involved in queries.

- **Numeric constants**

The period is used as decimal separator. Examples:

```
12
52.23
```

- **Text type constants**

They are contained within single quotes. Examples:

```
'Computer'
'Monitor'
```

- **Date or time type constants**

Date or time type constants are contained between # characters. Their format must respect the following rules:

- Years are expressed as 4 figures.
- Dates are expressed as Year-Month-Day.
- Time is expressed as Hours-Minutes-Seconds.
- The 24-hour clock is used (and not the 12-hour clock with A.M. or P.M.).
- The separator used in dates is the "/" or "-" character.
- The separator used for time is the ":" character.
- Months, days, hours, minutes, and seconds are expressed as 2 figures.
- When date and time are expressed together, the data always precedes the time, and they are separated by a space.

Examples:

```
#yyyy-mm-dd hh:mm:ss#
#yyyy-mm-dd#
#hh:mm:ss#
#2004-01-01 01:00:03#
```

- **Expressions**

Expressions are formed using:

- Constants
 - Fields
 - Functions
 - Sub-queries
- You can combine these elements with operators and parentheses in order to build complex

expressions. Comparison expressions take the form: <expression> <comparison operator> <expression>

Logical expressions take the form:

<comparison operator> <AND | OR> <comparison operator>

You can use parentheses to combine several logical expressions.

- **Operators**

- **Logical operators**

Logical operators are applied to link two expressions: **AND** and **OR**.

In order to optimize a query, it is sometime wise to avoid logical operators if a comparison operator can be used instead. The following example illustrates how to optimize a query filter used to select portfolio items whose **Assignment**field (SQL name: seAssignment) is set to **Awaiting delivery or Return for maintenance**. The values of these two elements of a system itemized list are "3" and "4" respectively. It is therefore possible to write:

```
(seAssignment=3) OR (seAssignment =4)
```

The last value of the system itemized list being "4", it is preferable to write the query as follows:

```
seAssignment >=3
```

- **Comparison operators**

Comparison operators are used to compare two expressions:

AQL comparison operators

Operator	Meaning
=	Equals
<>	Different than
!=	Different than
>	Greater than
<	Less than
>=	Greater or equal to
=<	Less than or equal to
=*	Right outer join. Because of the way in which AQL handles links, the use of this operator is limited.
*=	Left outer join. Because of the way in which AQL handles links, the use of this operator is limited.
LIKE NOT LIKE	Works like the = operator and allows you also to use wildcard characters. The following wildcard characters are available: "%" replaces any character string. "_" replaces any single character. Depending on the database engine used (SQL Anywhere, SQL Server and Sybase support it, Oracle for WorkGroups doesn't):

Operator	Meaning
	<p>[abc?] lets you define a list of possible characters (no space between the possible values.)</p> <p>[a-c] lets you define a range of possible values.</p> <p>DB2 does not support use of the LIKE X operator, if X includes a SQL column name. Only constants are supported for this operator. For example, the following query is not correct for DB2:</p> <p>SELECT COL1, COL2 FROM TABLE1 WHERE COL1 LIKE COL2</p>
IS NULL IS NOT NULL	<p>Tests whether the value of a field is "NULL" or not.</p> <p>Asset Manager only authorizes the "NULL" value for empty text fields and for Date or Date+Time fields that are not populated.</p>

Note: SQL Anywhere is not able to process "LIKE X" clauses when X contains more than 128 characters. If X is larger than 128 characters, applying the query provokes an ODBC error message. This problem can, for example, occur when displaying lists in tree view since this operation uses a "LIKE" clause on a "FullName" field.

- **Operators specific to sub-queries**

You can compare a value to the results of a sub-query using the following operators:

- = **ANY (sub-query)**
- = **ALL (sub-query)**
- = **SOME (sub-query)**

Example:

The following query provides the list of portfolio items whose brand is used at the Milwaukee site:

```
SELECT lModelId, Model.Brand FROM amPortfolio WHERE Model.Brand =
ANY (SELECT Model.Brand FROM amPortfolio WHERE Location.FullName
= '/Site Burbank' )
```

- **Selection list**

Selection lists define the items to be extracted or displayed. They specify the SELECT statements in queries. A selection list is made up of one or more expressions separated by commas: **<expression> [, <expression>...]**

Each expression can be linked to an alias. For example:

```
SELECT MrMrs, (Name + FirstName) Identity FROM amEmplDept
```

This is particularly useful at the level of export queries to attribute a name to the exported columns.

Note: Certain DBMSs limit the number of expressions in a given SELECT statement.

AQL Clauses

AQL clauses

Clause	Syntax	Examples
FROM	FROM <name of the table> [alias of the table] [, <name of	FROM amPortfolio FROM amPortfolio a, amLocation l

Clause	Syntax	Examples
	the table> [alias of the table>] ...]	<p>The following are equivalent:</p> <pre>SELECT AssetTag FROM amAsset SELECT a.AssetTag FROM amAsset a SELECT amAsset.AssetTag FROM AmAsset</pre>
<p>The first table indicated in the FROM clause of a query is the starting table of the query. If a query uses a field whose table is not specified, AQL considers that it belongs to the starting table of the query. The AQL FROM clause differs from the clause with the same name in SQL. For example, in the following sub-query, AQL searches the AssetTag field in the amAsset table:</p> <pre>SELECT AssetTag FROM amAsset</pre> <p>The number of tables authorized in a query depends on the DBMS used. Example:</p> <ul style="list-style-type: none"> • Oracle: You can use as many tables as you like. • Microsoft SQL Server or Sybase SQL Server: You are limited to 16 tables in a query. <p>Caution: When counting the number of tables in a query, do not forget to take into account those tables that are not explicitly mentioned, in particular if the query uses links. Also look out for the "fv_" notation (search of feature values) which generates an additional join at DBMS level. Similarly, the "cf_" notion (calculated fields) can generate additional joins.</p>		
<p>WHERE The AQL WHERE clause is equivalent to the clause in SQL of the same name. It specifies the search conditions, specifying the elements to extract from the database. These conditions can also express themselves in WHERE or HAVING clauses.</p>	<p>WHERE <Search conditions></p>	<pre><WHERE HAVING> [NOT] <expression> <comparison operator> <expression> <WHERE HAVING> [NOT] <logical expression> <WHERE HAVING> [NOT] <field> [NOT] LIKE 'xxxxx' <WHERE HAVING> [NOT] <logical expression> <AND OR> <logical expression > <WHERE HAVING> [NOT] <field> IS [NOT] NULL</pre> <p>In some other case, you may need to write more complex queries, such as:</p> <pre><WHERE HAVING> [NOT] EXISTS (<sub-query>) <WHERE HAVING> [NOT] <expression> [NOT] IN (<list of values> <sub-query>) <WHERE HAVING> [NOT] <expression> <comparison operator> <ANY ALL> (<sub-query>)</pre>

Clause	Syntax	Examples
<p>GROUP BY The AQL GROUP BY clause is equivalent to the clause in SQL of the same name.</p>	<p>GROUP BY <expression without aggregate> [, <expression without aggregate>]...</p>	<p>The following query gives the total number of brands found in the database. For each asset with an associated brand, Asset Manager returns an occurrence of the brand.</p> <pre>SELECT Count (Model.Brand.Name) FROM amAsset</pre> <p>By using the GROUP BY clause, we obtain a list of brands and the number of assets of each brand:</p> <pre>SELECT Model.Brand.Name, count(lAstId) FROM amAsset GROUP BY Model.Brand</pre>
<p>GROUP BY specifies subsets of the table. The subsets are defined in the GROUP BY clause by an expression, which can be the name of a field, for example. If aggregate functions are included in the selection list of the SELECT statement, GROUP BY searches the resulting value for each subset. These resultant values can be used in a HAVING clause. When a query makes use of the GROUP BY clause, each expression of the selection list must provide a single value for each subset.</p>		
<p>HAVING The AQL HAVING clause is equivalent to the SQL clause of the same name.</p>	<p>HAVING <Search conditions></p>	<p>Example of query where the WHERE clause is equivalent to the HAVING clause:</p> <p>The following query returns the list of brands whose name starts with a letter after the letter B and the number of asset of each of these brands:</p> <pre>SELECT Model.Brand.Name, count(lAstId) FROM amAsset GROUP BY Model.Brand.Name HAVING Model.Brand.Name > 'B'</pre> <p>It is also possible to express the same query using the WHERE clause:</p> <pre>SELECT Model.Brand.Name, count(lAstId) FROM amAsset WHERE Model.Brand.Name > 'B' GROUP BY Model.Brand.Name</pre> <p>Example of query using the HAVING clause: The HAVING clause enables you to use aggregate functions (such as Count); This is not the case with the WHERE clause. Thus, the following query searches all brands represented by more than one asset:</p> <pre>SELECT Model.Brand.Name, count(lAstId) FROM amAsset GROUP BY Model.Brand.N ame HAVING count (Model.Brand) > 1</pre>
<p>Differences with the WHERE clause It specifies search conditions like the WHERE clause. However, these two clauses differ as follows:</p> <ul style="list-style-type: none"> • The HAVING clause specifies the restrictions to be applied to aggregate functions in the selection list. The restrictions affect the number of resultant lines but do not affect the calculations linked to aggregate functions. • When the query uses an WHERE clause, the search conditions restrict the lines subject to aggregate calculation functions but do not affect the resultant lines. 		

Clause	Syntax	Examples
<p>ORDER BY The AQL ORDER BY clause is equivalent to the SQL clause of the same name. Items can be sorted:</p> <ul style="list-style-type: none"> • In ascending order: ASC. This is the default sort order. • In descending order: DESC. 	<p>ORDER BY <expression> [ASC DESC] [, <expression> [ASC DESC]...]</p>	

Clause	Syntax	Examples
<p>INSERT This clause enables you to insert one or more records into a database table.</p>	<p>INSERT INTO <Nam of the table> [alias of the table] (<Name of a field> [, <Name of a field>]...) VALUES (<expression> [, expression]...) AQL sub-query)</p>	<p>The INSERT clause can simplify the code of a Supplemental delivery information wizard:</p> <p>Wizard code not using the INSERT clause</p> <pre>hrAlarm = AmCreateRecord("amDateAlarm") lErr = AmSetFieldLongValue(hrAlarm, "bSecondLevel", 0) lErr = AmSetFieldLongValue(hrAlarm, "dtTrig1", AmGetFieldLongValue(hrAsset , 2)-lDaysBefore*86400) lErr = AmSetFieldLongValue(hrAlarm, "lAction1Id", lActionId) lErr = AmSetFieldLongValue(hrAlarm, "lMonitObjId", lAstId) lErr = AmSetFieldStrValue(hrAlarm, "MonitoredField", "dWarrEnd") lErr = AmSetFieldStrValue(hrAlarm, "MonitoredTable", "amAsset") lErr = AmSetFieldLongValue(hrAlarm, "sDaysBefore1", lDaysBefore) lErr = AmInsertRecord(hrAlarm)</pre> <p>Wizard code using the INSERT clause</p> <pre>lErr = AmDbExecAql("insert into amDateAlarm (bSecondLevel, dtTrig1, lActionId, lMonitObjId, MonitoredField, MonitoredTable, sDaysBefore1) values (0 , " & AmGetFieldLongValue (AmGetFieldLongValue (hrAsset, 2)-lDaysBefore*8640 0 & ", " & lAstId & ", 'dWarrEnd', 'amAsset', " & lDaysBefore & ")")</pre>
<p>UPDATE</p>	<p>UPDATE<Name of the table> [alias of the table] SET (<name of a field> [, <name of a field>...])</p>	<p>The UPDATE clause can help simplify the code of an action that triggers an order action.</p> <p>Action code not using the UPDATE clause</p> <pre>hr = AmGetRecordFromMainId("amPOrder", [lPOrdId]) lErr = AmSetFieldLongValue(hr, "seStatus", "\$ (IDS_POSTATUS_ORDERED) ") lErr = AmUpdateRecord(hr)</pre> <p>Action code using the UPDATE clause</p> <pre>lErr = AmDbExecAql("update amPOrder set seStatus = 21 where lPOrdId = " & [lPOrdId])</pre>

Clause	Syntax	Examples
<p>DUPLICATE This clause enables you to duplicate a record existing in a database table. This function is specific to Asset Manager. For more information, refer to the User interface guide, chapter Operations on records, section Duplicating a record.</p>	<p>DUPLICATE <Name of the table> [alias of the table] SET (<name of a field> [, <name of a field>...])</p>	
<p>DELETE This clause enables you to delete the fields of a record in a database table.</p>	<p>DELETE FROM... WHERE...</p>	

AQL Function References

You can also use native SQL functions of your DBMS. In this case, the resulting code is not portable.

Aggregate type AQL functions

Function	Description
Avg(<column>)	Returns the average value of a "number" type column. Returns "0" if the column does not have any records.
Count(<column>)	Counts the non-null values in a column.
Countdistinct(<column>)	Counts the distinct non-null values in a column.
Max(<column>)	Returns the maximum value in a "number", "string" or "date" type column. If the column does not have any records, returns "0" ("number" type column), "empty string" ("string" type column), or "empty date" ("date" type column).
Min(<column>)	Returns the minimum value in a "number", "string" or "date" type column. If the column does not have any records, returns "0" ("number" type column), "empty string" ("string" type column), or "empty date" ("date" type column).
Sum(<column>)	Returns the sum of the values of a "number" type column. Returns "0" if the column does not have any records.

These functions jointly use the **GROUP BY** and **HAVING** clauses.

String type AQL functions

Function	Description
Ascii(<String>)	Returns the ASCII value of the first character of the <string>.
Char(<n>)	Returns the character with ASCII code "n".
Left(<String>, <n>)	Returns the "n" first characters of the <string>.
Lower(<String>)	Returns the <string> in lowercase.
Ltrim(<String>)	Removes the spaces at the left of the <string>.
Right(<String>, <n>)	Returns the "n" last characters of the <string>
Rtrim(<String>)	Removes the spaces at the right of the <string>.
Substring(<String>, <n1>, <n2>)	Extracts the sub-string starting at character "n1" in the <string> and with length "n2" (the 1st character of the <string> is considered as character number 1).
Upper(<String>)	Returns the <string> in uppercase.

Date type AQL functions

Function	Description
Year(<date>)	Returns the number representing the year for a "date" or "date and time type field" (e.g: 2000).
Month(<date>)	Returns the number of the month for a "date" or "date and time type field" (1=January, ?, 12=December).
Day(<date>)	Returns the number of the day in the month for a "date" or "date and time" type field (1-31).
DayOfYear(<date>)	Returns the number of the day in the year for a "date" or "date and time" type field (1-366).
WeekDay(<date>)	Returns the number of the day in the week for a "date" or "date and time" type field. This number depends on how the server is configured. For example, the default configuration under Sybase or Microsoft SQL Server is (1=Sunday, 2=Monday, ?, 7=Saturday). The default configuration under Oracle is (1=Monday, ?, 7=Sunday).
Hour(<hour>)	Returns the number of the hour in the day for a "time" or "date and time" type field (0-23).

Function	Description
Minute(<hour>)	Returns the number of minutes for a "time" or "date and time" type field (0-59).
second(<hour>)	Returns the number of seconds for a "time" or "date and time" type field (0-59).
Getdate()	Returns the server's current system date.
AddDays(<date>, <number>)	Adds a given number of days to a "date" or "date and time" type field.
AddHours(<date>, <number>)	Adds a given number of hours to a "date" or "date and time" type field.
AddMinutes(<date>, <number>)	Adds a given number of minutes to a "date" or "date and time" type field.
AddSeconds(<date>, <number>)	Adds a given number of seconds to a "date" or "date and time" type field.
DaysDiff(<date1>, <date2>)	Number of days between the dates date1 and date2 ("floating point" number with decimals)
HoursDiff(<date1>, <date2>)	Number of hours between the dates date1 and date2 ("floating point" number with decimals)
MinutesDiff(<date1>, <date2>)	Number of minutes between the dates date1 and date2 ("floating point" number with decimals)
SecondsDiff(<date1>, <date2>)	Number of seconds between the dates date1 and date2 ("floating point" number with decimals)
DbToLocalDate(<date>)	Converts a date expressed in the time zone of the database server to a date expressed in the time zone defined at client machine level.
LocalToDbDate(<date>)	Converts a date expressed in the time zone of the client machine to a date expressed in the time zone of the database server.

Examples of Date-type functions

Example	Asset Manager Query Language
All records modified during the last week	AddDays(dtLastModif,7)>=Getdate()

Example	Asset Manager Query Language
All work orders notified in the last hour	HoursDiff(Getdate(), dtNotif) <= 1 Or AddHours(dtNotif, 1) >= Getdate()
All work orders closed in the last half-hour	MinutesDiff(Getdate(), dtActualFixed) <= 30 Or AddMinutes(dtActualFixed, 30) >= Getdate()

The following query lists the work orders effectively carried out and resolved during the same day. The time zone of the client machine is taken into account:

```
SELECT Self FROM amWorkorder WHERE
DayOfYear( DbToLocalDate( dtActualFixStar t ) ) =
DayOfYear( DbToLocalDate( dtActualFixed ) )
```

The following query lists all work orders that have effectively been started today:

```
SELECT Self FROM amWorkorder WHERE
DayOfYear( DbToLocalDate( dtActualFixStar t ) ) =
DayOfYear( DbToLocalDate( GetDate ( ) ) )
```

Numeric type AQL functions

Function	Description
Abs(<Number>)	Returns the absolute value of a "number".
Ceil(<Number>)	Returns the smallest integer greater or equal to a "number".
Floor(<Number>)	Returns largest integer less than or equal to a "number".
Mod(<a>,)	Returns the remainder of the integer division from "a" by "b" (a = qb + r, with q integer and 0 =< r < q).
Round(<a>, <n>)	Rounds "a" to "n" decimal places.
Trunc(<a>, <n>)	Truncates "a" to "n" decimals.

Examples of application:

```
Abs (2.516) = 2.
Ceil (2.516) = 3.
Floor (2.516) = 2.
Mod (6,4) = 2.
Round (31.16, 1) = 31.20.
Round (31.16, 0) = 31.00.
Round (31.16, -1) = 30.00.
Trunc (31.16, 1) = 31.1.
```

Test type AQL functions

Function	Description
IsNull(<a>,)	If "a" is "Null", replaces "a" by "b". The types of "a" and "b" must be compatible.

Appendix A Sample Scenarios of HTTP Service Connector

Connect-It provides sample scenarios to demonstrate the use of the HTTP Service connector, and the scenarios can be grouped by prefix of name.

Scenarios

samples-*.scn

The scenarios in this group demonstrate how to publish a certain document type of the Asset Manager connector as a service of specified format, for example, UDM. See the scenario names as below:

samples-db-server.scn
samples-db-client-even.scn
samples-db-client-node.scn
samples-db-client-soft.scn

scheduling-*.scn

The scenarios of this group demonstrates a service of reading delta data from the Asset Manager connector (**amComputer**). This group has the following scenarios:

scheduling-db-server.scn
scheduling-db-client-delta.scn

webservice-*.scn

The scenarios of this group demonstrate how to convert a Web Service to an HTTP REST service. The group consists of the following scenarios:

webservice-server.scn
webservice-client.scn

Folders

- **conf** - to save the `.xsd` and `.dsc` files that are required for running scenarios.
- **output** - to save output files.

Scenario description

sample-db-server.scn

This scenario provides three kinds of services: Node Service, Software Service, and Event Trigger Service. Run this scenario using the 'start all scheduler' function. The entry URL of the scenario is **`http://localhost:8080/cit-rest/service/`**

Node Service

The entry URL of the service is **`http://localhost:8080/cit-rest/service/node/`**

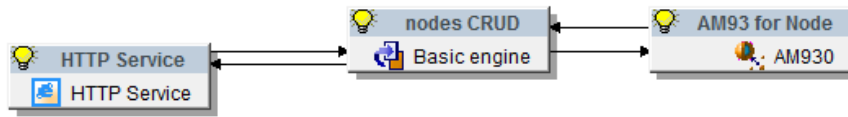


Figure 11-4: Node service of sample-db-server.scn

Source/Mapping	Destination
amComputer (amComputerSrc)	node (node)
amComputerSrc-nodeDst	amComputer (amComputerDst)
node (node-write)	amComputer (amComputerDst1)
Insert	node (node-response)
Delete	
ProcessReport (ProcessReportSrc)	
Report for CRUD	

Figure 11-5: Node service mapping

These mappings in the figure above implement data transfer between the Asset Manager connector and HTTP Service connector. The first mapping is to read data from the Asset Manager **amComputer** node. The second and third mappings are to receive data request from the HTTP Service connector and insert/delete the data in the **amComputer** node. The last mapping is the response of the second and third mapping request.

Consumed document types	Produced document types
<ul style="list-style-type: none"> node (node) asset (asset) node (node-response) running-software (software) running-software (running-software-response) 	<ul style="list-style-type: none"> node (node-write) running-software (software-write)

Figure 11-6: Detail of document types

There are three document types related to the node service:

- **node(node)** - for REST client (method = GET) to retrieve node data from the **amComputer** node
- **(node-write)** - for REST client (method = PUT/POST/DELETE) to send node data to **amComputer**
- **node(node-response)** - to return the result data to REST client (method = PUT/POST/DELETE)

When you define the document type, pay attention that **node(node-write)** must be linked to **node(node)** and **node(node-response)** must be linked to **node(node-write)**. See the screenshot as below:

Use a Status Report connector as the data source

Status report connector:

Use the current document type as a response of a produced document type

Produced document type:

Figure 11-7: Defining the produced and consumed document types

Software Service

Similar with Node Service. The entry URL of the service is **http://localhost:8080/cit-rest/service/software/**In this service, the HTTP structure includes the HTTP method and URL parameters for HTTP REST (method = PUT/POST/DELETE) . It's defined in second mapping in the **soft CRUD** mapping box.

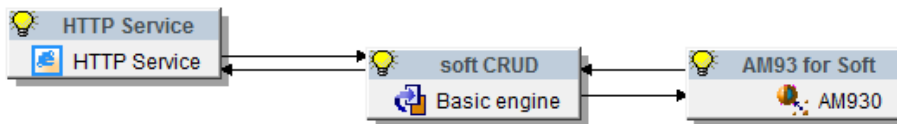


Figure 11-8: Software service of sample-db-server.scn

Event Trigger Service

This service returns a status report as response. The entry URL of the service is **http://localhost:8080/cit-rest/service/asset/**When the HTTP Service connector listener receives an HTTP request, it will trigger the mapping starting to run. When the mapping is finished, the HTTP Service connector will get the report result from the Status Report connector, and then return a Restful response (method = GET).

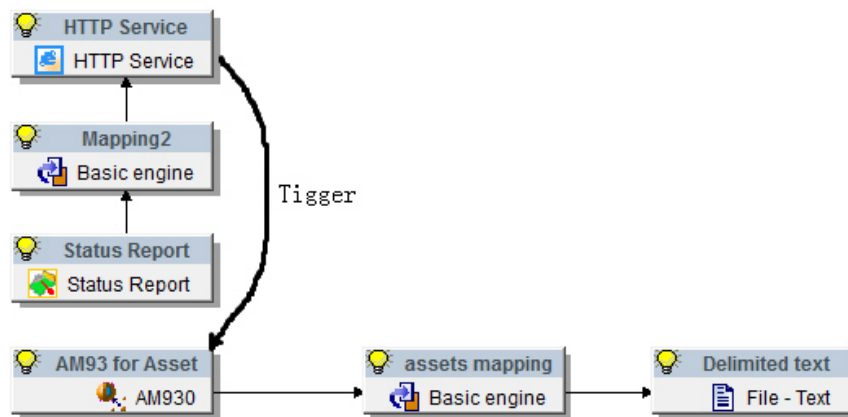


Figure 11-9: Event trigger service of sample-db-server.scn

samples-db-client-node.scn

This scenario demonstrates how to transfer data from or to an Asset Manager database (**amComputer**).

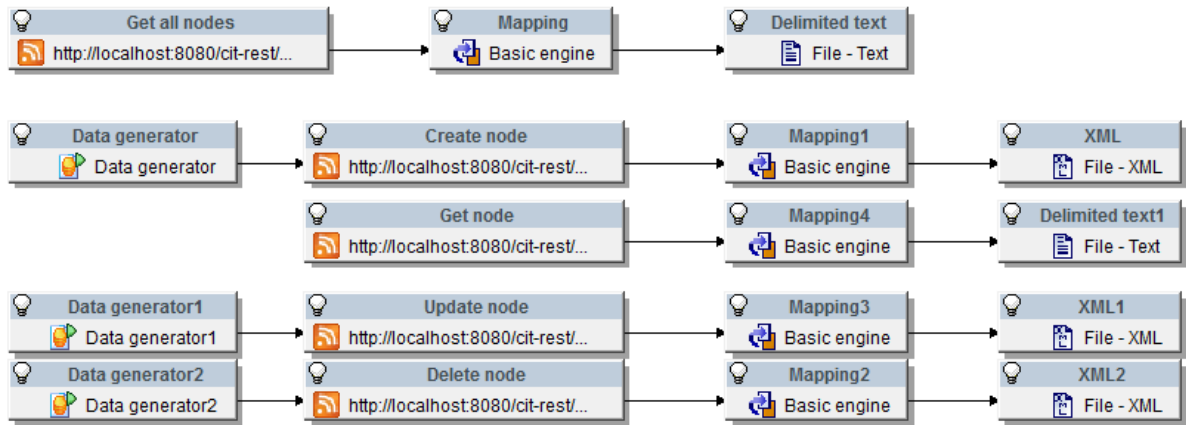


Figure 11-10: Node service mapping for client

Notes:

- The first mapping REST URL is **http://localhost:8080/cit-rest/service/node/?query=ComputerType='Laptop'&order-by=ComputerDesc** To use parameters, **query** and **order-by** are allowed.
- The third mapping uses **http://localhost:8080/cit-rest/service/node/?query=AssetTag='maso001'** To retrieve single record.
- The rest of mappings are for creating, updating and deleting records.

samples-db-client-soft.scn

This scenario demonstrates how to create, update, read or delete a software service.

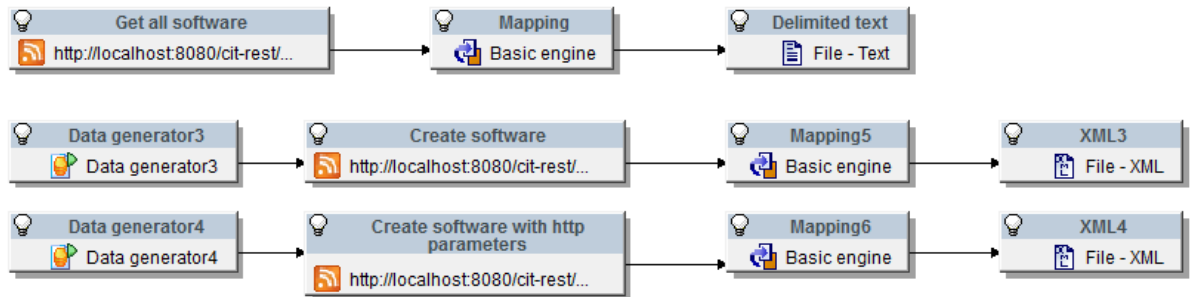


Figure 11-11: Software service mapping for client

The third mapping demonstrates how to use the HTTP parameter in URL. In the example shown as below, the **os** parameter will be sent to server mapping. You may refer to the mapping script in server side:

```

if [HTTP.Parameters(0).Name] = "os" then
retval = [HTTP.Parameters(0).Value]
else
retval = ""
end if
    
```

samples-db-client-even.scn

This scenario demonstrates how to trigger a standard mapping.

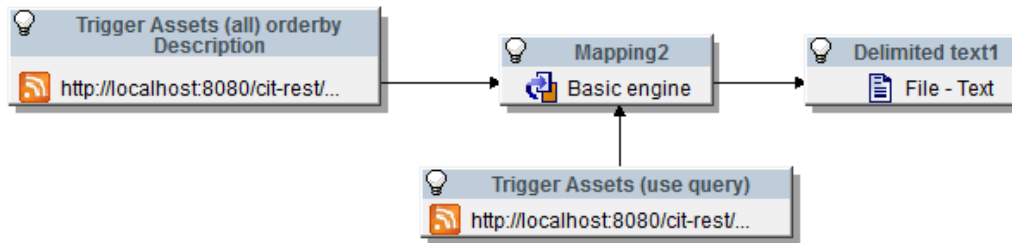


Figure 11-12: Event trigger service mapping for client

The first mapping <http://localhost:8080/cit-rest/service/asset/?order-by=Description> uses **order-by** parameter. The mapping 'Trigger Assets (user query)' uses <http://localhost:8080/cit-rest/service/asset/?query=AssetTag='DEMO-001'> to fetch a specified record.

scheduling-db-server.scn

This scenario This is a server scenario providing mapping for reading data from an Asset Manager database. Run this scenario using the 'start all scheduler' function. The entry URL of this scenario is <http://localhost:8080/cit-rest/service/>The 'Use a schedule pointer' function is enabled in the Asset Manager connector wizard, and it is for a client to retrieve delta data since the last session. You should disable the schedule pointer function when using the HTTP Service connector.

scheduling-db-client-delta.scn

This scenario is scheduled to read documents that have not been processed since the last session from an Asset Manager database. This scenario must be used with the scenario **scheduling-db-server.scn**.

webservice-server.scn

This is a server scenario for information query from the SOAP connector. Run this scenario using the 'start all scheduler' function, and then a translation web service is returned to HTTP REST service.

The HTTP Service connector can get HTTP parameters from a client request. For example, suppose 'word' is an HTTP parameter. Below is the script using this parameter value:

```

if [HTTP.Parameters(0).Name] = "word" then
retval = [HTTP.Parameters(0).Value]
else
retval = [Word]
end if
  
```

This script indicates that if there is an HTTP parameter named "word", translate this parameter's value. Otherwise translate the data sent from REST client (method = POST).

webservice-client.scn

The REST (Deprecated) Client URL of this scenarios is **http://localhost:8083/cit-rest/service/translator?word=test**

Even if the Data Generator put 'word=Connected' (HTTP REST method = POST), the server still translates it as 'word=test'.

Index

A	
Action Request System	136
additional connectors	12
Altiris	124
Aperture VISTA	126
AQL	288
AQL syntax	300
asset manager exposure	67
B	
base connectors	12
C	
Client Automation Inventory Manager	75
Client Automation Management Portal	76
command-line	172
compatibility	12
D	
DTD	29
M	
Microsoft System Center Configuration Manager (SCCM)	130
O	
obsolete connectors	12
optional connectors	12
R	
RESTful (Deprecated) client	222
X	
XSD	29