SAP NetWeaver® Identity Management Connector Development Kit

Certification

Version 7.2 Rev 1

Preface

The product

The SAP NetWeaver Identity Management Connector Development Kit enables independent software vendors (ISVs) or SAP partners to create an Identity Management connector for their application, and to integrate the application into the Identity Management landscape.

The Connector Development Kit contains information necessary for development of an Identity Management connector – criteria, guidelines, templates, test tool, certification guide, etc.

The reader

This manual is written for people who have implemented or plan to implement an Identity Management connector for their application, as well as people who are to perform the certification process.

Prerequisites

To get the most benefit from this manual, you should have the following knowledge:

- Thorough knowledge of the Identity Center.
- Thorough knowledge of the Virtual Directory Server.
- Basic knowledge of LDAP.

The following software is required:

- SAP NetWeaver Identity Management Identity Center version 7.1 SP2 or newer, or version 7.2 or newer installed and licensed. At least one dispatcher has been configured and is running.
- (Optional) SAP NetWeaver Identity Management Virtual Directory Server version 7.1 SP2 or newer, or version 7.2 or newer installed and licensed.
- (Optional) Vendor specific module correctly installed and configured, in cases where SAP NetWeaver Identity Management Virtual Directory Server is not used.
- SAP NetWeaver Identity Management User Interface must be installed and configured for this Identity Center and identity store (according to the document *SAP NetWeaver Identity Management Identity Center: Installing and configuring the Identity Management User Interface* available with the Identity Center installation set or on SAP Service Marketplace).
- A Java development environment. This can be downloaded from http://java.sun.com (version 1.4/1.5).
- Access to the target application.
- The Identity Management connector for target application, repository definition, job and tasks for the certification process, and the corresponding documentation created and delivered by the independent software vendors (ISVs) or SAP partners.

The manual

The purpose of this document is to outline the requirements and the process of the Identity Management connector certification.

Related documents

You can find useful information in the following documents:

- *SAP NetWeaver Identity Management Connector Development Kit Overview.*
- *SAP NetWeaver Identity Management Connector Development Kit Implementing the Virtual Directory Server Connector*.
- *SAP NetWeaver Identity Management Connector Development Kit Virtual Directory Server Connector Testing Tool.*

Table of contents

Introduction

SAP offers integration and certification support for their partners and independent software vendors (ISVs) that wish to certify their products.

SAP partners and ISVs are assisted by SAP during their product integration projects. They are offered consulting, certification testing and test system access services independently. The actual development is executed by the ISVs. It is recommended that ISVs build up some knowledge about the relevant SAP solutions and on integration topics in-house, attend SAP training classes or work with SAP Service Partners.

SAP tests the Identity Management connector built by ISVs or SAP partners for correct implementation of technologies in an SAP environment, i.e. only the architectural and functional aspects are considered in the certification process. The quality measures (security, performance, scalability, etc.) of the solution are neither measured nor considered in this certification.

SAP does not guarantee that a third-party solution is error-free after the certification is completed, since it is only testing the limited scope within a pre-defined integration scenario and not the solution as a whole. Also, SAP neither sells nor supports any of these solutions as part of the certification process.

The certification validates (according to the documentation) the installation of the connector and that the following functions of the Identity Management connector work:

- Read and search for entries
- Create a new entry
- Modify an entry
- Delete an entry

These tests cover the basic functionality to verify that the connector works properly.

Validation results are structured in the form on page 34 and presented in the test report.

Once all tests have been completed successfully and all the certification criteria (described in the following subsection) are met by the partner, SAP will grant the certificate.

Certification criteria

This section contains the certification criteria (the requirements) that need to be taken into consideration and fulfilled by the partners before the testing process for certification can be executed. The requirements for Identity Management connector are categorized as:

- Packaging and documentation requirements (deliverables)
- Development requirements

Be aware that the criteria are relevant to the development of the connector and they must be kept in mind before the final version of the connector package is created.

Deliverables

Part of the certification process is the submission (by ISVs or SAP partners) of the following documents and information:

- Necessary target application Java libraries.
- An initial data set needs to be provided by the partner (the number of entries has to be at least 300). It is a partner's responsibility also to provide the necessary documentation for the correct installation of the initial data set.
- Identity Center, including the Identity Management User Interface, deliverables:
  - Job and the connector tasks in the Identity Center necessary for the certification process (a repository definition, a job which reads entries from the target application and a search job, a job for creating of account privilege, tasks for adding, modifying and deleting a user, and a User Interface task for editing of a user), with installation and configuration documentation.
  - (Optional) Identity store schema extension: if the connector requires any identity store schema extensions, these need to be provided for import (an exported .mcc file). Store the exported schema file the folder *IC files* in the connector certification package.
- Virtual Directory Server deliverables (Optional, not to be delivered if vendor specific (connector) module is used instead of Virtual Directory Server):
  - Virtual Directory Server (VDS) configuration (as template): as explained in *SAP NetWeaver Identity Management Connector Development Kit Implementing the Virtual Directory Server Connector*.
  - (Optional) Data Source configuration (as template): as explained in *SAP NetWeaver Identity Management Connector Development Kit Implementing the Virtual Directory Server Connector*.
  - Connector class (compiled for JDK 1.4 or 1.5) or as source file (stored in the Virtual Directory Server configuration file). The class is stored in <VDS_dir>\configurations in a folder with the same name as your configuration file.

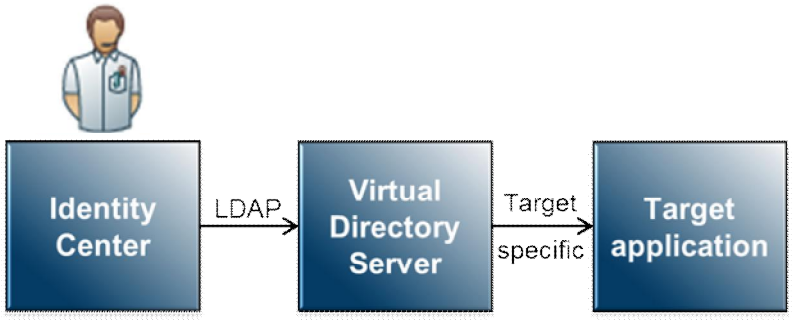
  Note:
  Whether the class should be compiled for JDK 1.4 or 1.5 will depend on which AS Java version the connector class is to be deployed on. Compiling the class for JDK 1.4, the Virtual Directory Server configuration that utilizes this class will be deployable on AS Java versions 7.0, EHP 1 for SAP NW CE 7.1, SAP NW CE 7.2 and SAP NW 7.3. Compiling for JDK 1.5 will result in connector not being deployable on AS Java 7.0 versions.
- Vendor specific (connector) module (Optional, only delivered if used instead of Virtual Directory Server).
- Documentation deliverables:
  - Functionality description sheet which documents the purpose of the connector, which features are supported, as well as entry types and attributes. The documentation must be submitted in English only (PDF).
  - Installation and configuration documentation for the connector tasks and jobs, and the repository definitions in the Identity Center. The documentation must be submitted in English only (PDF).

- Installation and configuration documentation for the Virtual Directory Server connector (configuration with the code and the API) need to be included in the documentation. The documentation must be submitted in English only (PDF). Optionally, if a vendor specific connector is used instead of Virtual Directory Server connector, then a detailed installation and configuration documentation for the vendor connector is necessary instead.

Development requirements

In addition to the information given by this document, the information given by document *SAP NetWeaver Identity Management Connector Development Kit Implementing the Virtual Directory Server Connector* (available on the SAP Developer Network (SDN)) affects the development of the Identity Management connector content. This is only relevant if the Virtual Directory Server is used when developing the Identity Management connector.

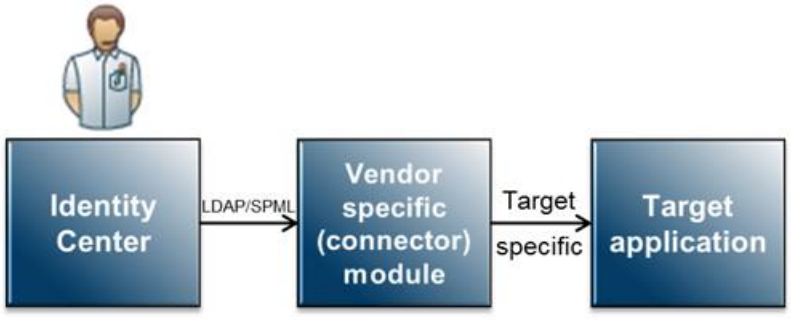
Validation environment



The figure above shows the validation testing environment setup of the recommended solution. All tests are performed using the SAP NetWeaver Identity Management Identity Center (including the Identity Management User Interface).

In the recommended solution, the Identity Center communicates with the Virtual Directory Server using the LDAP protocol, using jobs and tasks (as described in this document in section *Logical setup in the Identity Center* on page 4). The Virtual Directory Server in turn communicates with the target application, using the target specific protocol.

Alternatively, the communication can go to either the backend or to a vendor specific connector module using one of SAP NetWeaver Identity Management standard protocols like SPML or LDAP. In this case the Virtual Directory Server and its deliverables as previously described are optional (can be omitted), and it is necessary to specify vendor specific module deliverables.

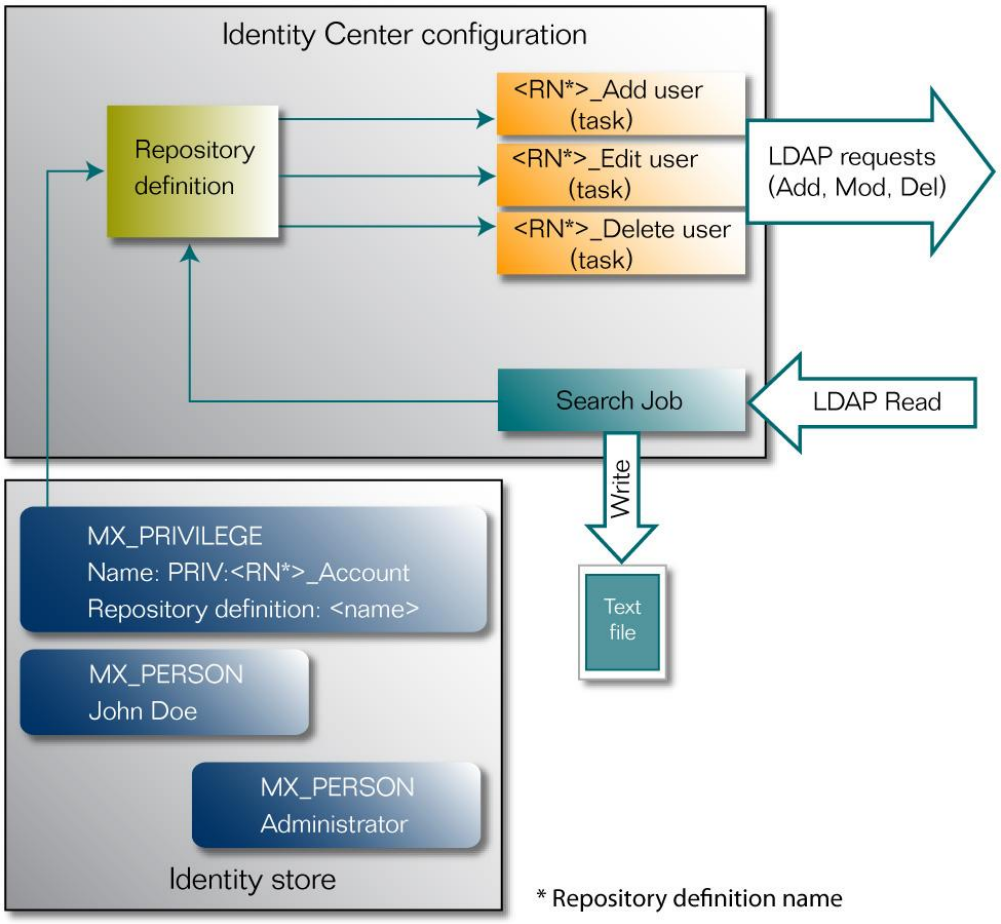


Inspecting the results of some of the tests involves inspecting data in the target application.

Throughout the document, the recommended version of Identity Management connector with the Virtual Directory Server connector is used as the example.

Note:
The processes described in the documentation are valid for both SAP NetWeaver Identity Management 7.1 and 7.2. Most of the screen shots are taken from the 7.2 version. There are separate descriptions in cases where the two versions differ from each other.

Logical setup in the Identity Center



The figure shows the configuration of the Identity Center:

- The repository definition defines the target application. Parameters of this repository definition are to be defined by the partner.
- There are three tasks *(<RN>_Add user*, *<RN>_Edit user* and *<RN>_Delete user*, where *<RN>* is a repository definition name) which are used to add, modify and delete the user in the target application respectively. These tasks are referenced from the repository definition and are triggered when privileges referencing this repository definition are added or removed from the user or the user with the privilege that is referencing this repository definition is modified.
- The job *Search Job* is used to read the entries from the target application and test the search functionality. The read data is stored in a file (.txt or .csv), which can be inspected and compared to the file *<Repository definition name> initial entries.txt* (or .csv) delivered by the partner.

- In the identity store, the following is needed:
  - Privilege object "PRIV:<Repository definition name>_Account": this privilege will reference the repository definition.
  - Person object "John Doe": this is the person which will be created and maintained in the repository definition.
  - Person object "Administrator": this user is used to execute the User Interface tasks (e.g. a User Interface task for user editing, like modifying attributes of a user, adding and removing the privileges etc.).

Relation between privilege and repository definition

A privilege references a repository definition by name, using the MX_REPOSITORYNAME attribute.

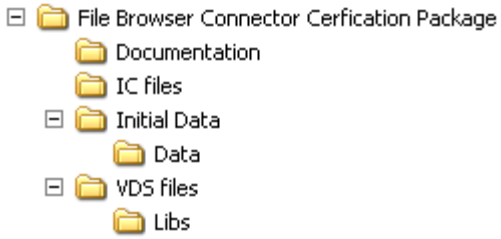
When the privilege is assigned to a user, a pending value object (MX_PENDING_VALUE) is created and the *<Repository definition name>_Add user* task is executed on this object, which will create the user in the target application. When operation completes successfully, the privilege is given to the user.

When a user which has the privilege is modified (e.g. telephone number attribute is altered), the *<Repository definition name>_Edit user* task is executed on the user. The task updates the user information in the target application.

When the privilege is removed from the user, a pending value object is created and the *<Repository definition name>_Delete user* task is executed on this object. The task removes the entry from the target application. When the task execution completes successfully, the privilege is removed.

The package

The package that the partner provides for the certification process must follow the recommended package structure and naming.



The package name must be *<Connector name> Certification Package* (e.g. *File Browser Connector Certification Package* where *File Browser Connector* is the name of the connector). The package must have the following structure with these four folders described in the subsections below:

- Documentation
- IC files
- Initial Data
- VDS files

Note:
The VDS files folder should be replaced by the folder for vendor module files in cases where the vendor specific module is used instead of Virtual Directory Server.

Documentation

The documentation must be submitted in English only and in the PDF format. The folder should contain documents covering the following topics:

- Functionality description sheet (*Functionality description sheet.pdf*): This document describes the purpose of the connector, the features that the connector supports and a description of the entry types and attributes that the connector manages.
- Identity Center specifications (*Identity Center specifications.pdf*): Installation description for the connector tasks in the Identity Center and the repository definitions. A complete explanation of the repository constants is needed, and how to link the modifiable attributes with the modification task. This process can be done in several ways and it is up to the partner to decide which way is the most adequate for every attribute.
- (Optional) VDS configuration specifications (*VDS configuration specifications.pdf*): Installation and configuration description for the VDS connector with the most relevant facts related to the setup of the Virtual Directory Server configuration (parameters to be set, configuration steps etc). If Virtual Directory Server is not used, there is no need to deliver the VDS configuration specification.
- (Optional) Connection configuration specifications for vendor specific (connector) module (*Connection configuration specifications.pdf*): If the vendor module is used instead of Virtual Directory Server, then the documentation describing the setup of the module as well as a detailed documentation on how to configure the connection in SAP NetWeaver Identity Management is necessary.

IC files

This folder contains the file *<Repository definition name> tasks and repository.mcc*. The file contains three tasks for adding, modifying and deleting a user in the backend system – three of four operations (the fourth being the search operation) that are required to get the certification for the connector, in addition to the related repository definition.

The folder also contains the file *<Repository definition name> jobs.mcc*, which may contain a job for the search operation in the backend system, as well as the job for creating of an account privilege. The file *<Repository definition name> jobs.mcc* may also contain any extra jobs that the partner considers necessary to accomplish some of the certification goals. All information about the jobs must be written in the Identity Center specifications file (*Identity Center specifications.pdf*). The section describing the extra jobs should be called "Extra jobs" and be the last section of the document. For each job the purpose of the job, the expected results and other important considerations related to the job (if any) will be described.

This folder may also contain a file *<Identity store name>_identity store schema.mcc* containing an export of the identity store schema, in cases where identity store schema extensions are necessary for the connector.

Both *<Repository definition name> tasks and repository.mcc*, *<Repository definition name> jobs.mcc* and *<Identity store name>_identity store schema.mcc* will be imported to the Identity Center as shown later in this document.

Initial Data

This folder contains an initial backend system data set, a file with a set of entries and the respective attributes. A sub-folder "Data" contains the actual initial data (entries) that the partner provides. The initial entries are also listed in the file *<Repository definition name> initial entries.txt.*

The number of entries that the file must contain has to be at least 300 and it must coincide with the result of the search job execution in the Identity Center. The result of the search job will be stored in a file (e.g. *<Repository definition name> search job result.txt*) which must be compared with the file *Initial entries.txt* in order to make sure that the search job works correctly, i.e. the search results need to be compared with the expected search result data (the files should be identical).

Note:
The file containing the initial entries and the file containing the results of the search job do not have to be .txt files. A CSV file can be used.

Note:
Comparing 300 entries (at least) manually can be a quite tedious job, but any tool for comparing the content of two files may be used for this purpose (e.g. the command line tool "fc" for those using Microsoft Windows as their operating system).

It is a partner's responsibility to provide the necessary documentation for the correct installation of the initial data set. The documentation file will have the name *<Repository definition name> initial data installation.pdf.*

VDS files

This folder contains the mandatory Virtual Directory Server configuration file which will follow the naming *<Connector name>.xml*. Besides the Virtual Directory Server configuration file, the folder may (optionally) contain a data source configuration file. This file should be called *<Data Source name>.xml*.

Also the connector class and the necessary target libraries are the content of this folder. The class can be either in source code (.java file) or a compiled class (.class file). It must compile for JDK 1.4 or 1.5. The libraries, stored in the sub-folder *Libs*, must be added to the Virtual Directory Server classpath in case it is necessary to compile the connector class before running the server.

This folder and the files can be omitted if the Virtual Directory Server is not used in the solution. It should be replaced by the necessary vendor module files if the vendor specific module is used instead.

Contents of the sample connector package

The sample connector (File Browser Connector), using the Virtual Directory Server, has the purpose of managing the file system through creation and deletion of files and directories, modification of the file content, and search operation which shows all the files and directories under a given directory.

The content of the package *File Browser Connector Certification Package* is:

- Documentation: Contains the functionality description sheet, the Identity Center and the Virtual Directory Server specifications.
- IC files: The folder contains two files: *FS tasks and repository.mcc* which contains the Identity Center tasks and the repository definition, and the file *FS jobs.mcc* that contains the search operation job. These two files must be imported to the Identity Center.
- Initial Data: Contains a set of data which the partner provides as the initial state in the system (stored in the sub-folder *Data*). It also contains a file describing the set of sample entries called *Initial entries.txt*. The entries must be copied as explained in the file *FS initial data installation.pdf* provided in this folder.
- VDS files: Contains the file *FileBrowserConnector.java* which holds the connector source code, the file *fileBrowser.jar* as the target application library (stored in the sub-folder *Libs*) and the Virtual Directory Server configuration file *File Browser Connector.xml*.

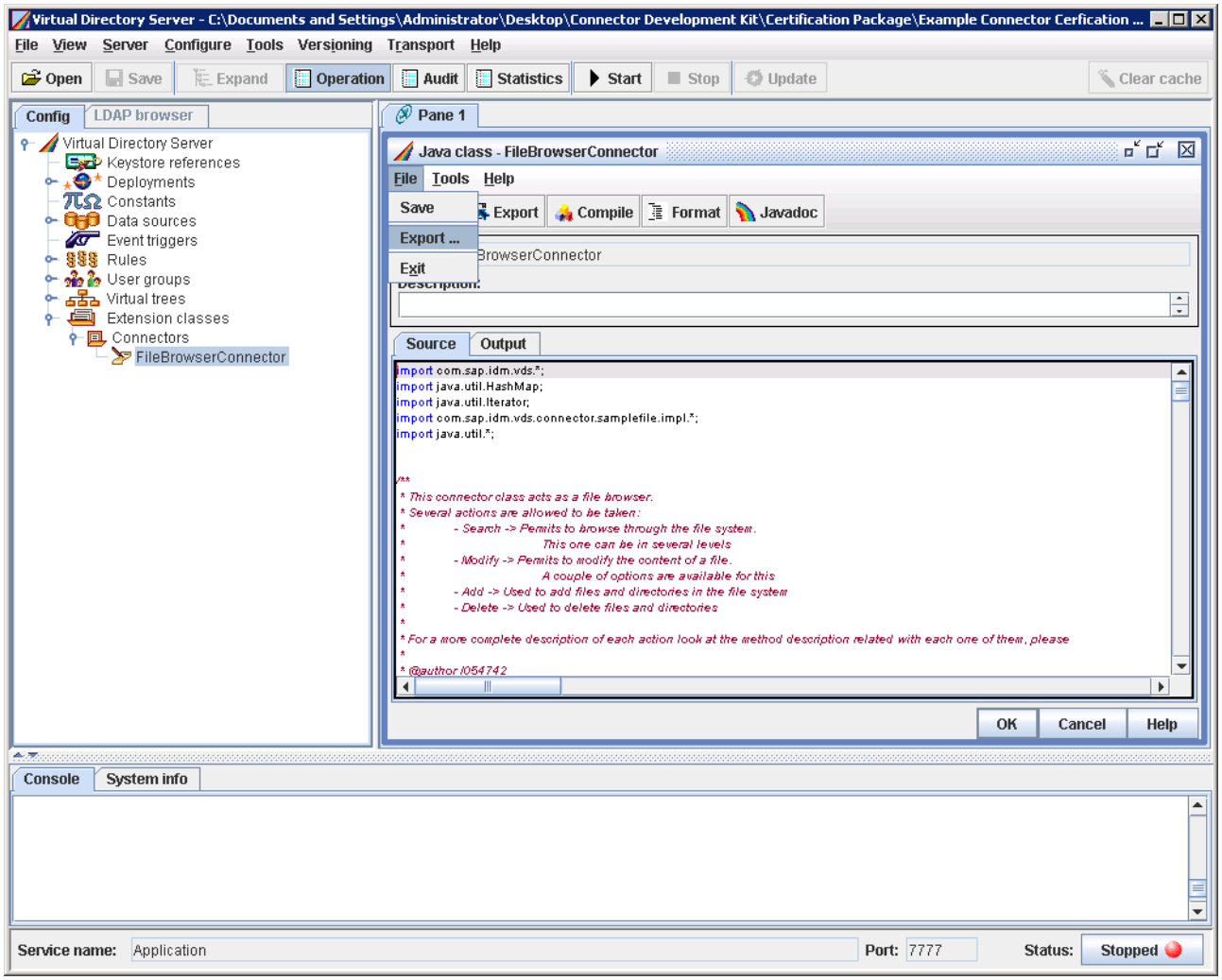
Section 1: Creating a certification package

In this section it is described how to create a certification package. The Virtual Directory Server based sample connector kit File Browser Connector is used as an example throughout the sections. The Virtual Directory Server parts of the certification package creation process should be replaced by the vendor module details, if it is used instead of the Virtual Directory Server.

Saving the connector class

In our example we will save the connector class in the Virtual Directory Server:

1. Navigate to (Extension classes\Connectors) and select the connector class, here "FileBrowserConnector", in the Virtual Directory Server console tree.
2. Open the connector class (by double-clicking it).
3. Once the class dialog box is open, choose "Export…" from "File" in the main menu:



4. Select the folder where you wish to save the source code .java file.

Another way of providing the connector class is to compile it with JDK 1.4 or 1.5 and provide the .class file. To do so, do the following:

1. Choose **Tools/Options…** from the main menu and select the "General" tab:

   Make sure that "Enable compilation" option is enabled in the "Java compiler" section.

   Select "1.4" or "1.5" in the "Target VM" field.

   Specify which compiler to use.

2. Choose "OK" to save the options and close the "Options" dialog box.
3. Choose 

to compile the Java class.

Now the .class file is ready. Include the .java or the .class file in the certification package (*VDS files* folder).

Saving the libraries

All non-standard java libraries needed by the connector class have to be included in the certification package (*VDS files/Libs* folder). For the sample connector the *fileBrowser.jar* library which contains all the logic for the file management is saved.

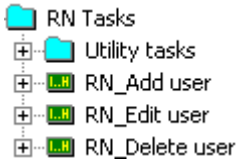
Saving the Virtual Directory Server configuration file

The server configuration is stored in an XML file. This file is usually stored in the *configuration*s folder in the Virtual Directory Server installation directory (normally *C:\usr\SAP\IdM\Virtual Directory Server\configurations*) as described in document *SAP NetWeaver Identity Management Connector Development Kit Implementing the Virtual Directory Server Connector*. Include this file in the certification package (*VDS files* folder).

Creating the Identity Center tasks and the repository definition

The next step is to create the Identity Center tasks and the repository definition that will contain all supporting data for the tasks and jobs.

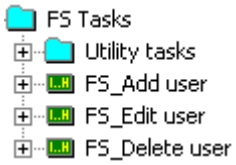
The set of tasks will follow a standard structure and naming. The structure can be defined in the following way:



where <RN> is the name of the repository definition. There are three tasks:

- RN_Add user: This task will provision a new entry to the target system - it will communicate to the Virtual Directory Server to create a new entry in the target system with certain information.
- RN_Edit user: This task is used to modify any attribute in any entry in the target system. The task will send the changes to the Virtual Directory Server.
- RN_Delete user: The task will de-provision entries from the target system.

For our sample connector the repository definition is called *FS* (from File System). The set of tasks for the sample connector will look like:



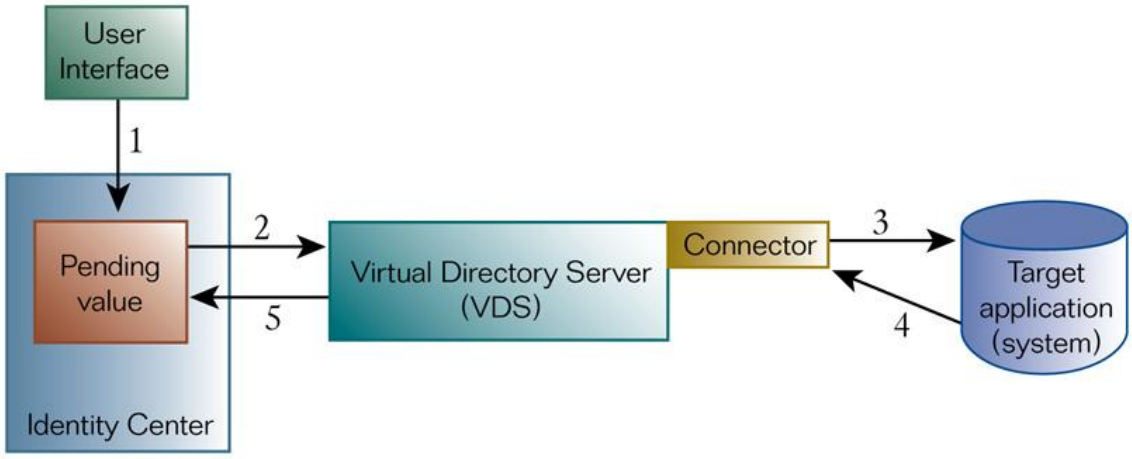
The task *FS_Add user* passes the information to the Virtual Directory Server (VDS) when creating a user is desired. The task *FS_Edit user* sends the information to VDS when it is desired to modify the content of a file, while the task *FS_Delete user* is used when deleting an entry.

A fourth, not mandatory, task exists in the folder *Utility tasks*. This folder will contain any extra tasks created to support the three mandatory tasks mentioned above. For the sample connector, the fourth task is *Start Privilege Task* which distinguishes between the provisioning and the de-provisioning operations and calls the correct task with the correct data. Provisioning and de-provisioning will occur when adding or removing the account privilege for our target system to/from any MX_PERSON entry in the Identity Center respectively. This results in the following workflow when adding an entry in our target system:

- The account privilege (previously created) is added to an MX_PERSON entry.
- When adding the account privilege a pending object is created. This pending object executes the task given by the attribute MX_ADD_MEMBER_TASK, which is a part of the account privilege attribute set.
- The task collects information to distinguish between provisioning (adding the privilege to an entry) and de-provisioning (removing the privilege from an entry), and hence it will be the same for the privilege attribute MX_DEL_MEMBER_TASK. The pending value waits for the result of the operation of the task.
- If the operation is successfully executed then the privilege will be assigned, otherwise it will not.

In summary, when adding the account privilege to an MX_PERSON entry in the Identity Center a pending value is created which will trigger the provisioning task (create an entry in the target system). If the operation completes successfully the attribute is assigned to the entry. The same process occurs when removing the account privilege from an MX_PERSON entry, but in this case the executed task is a de-provisioning task (deletes an entry in the target system).

A graphic presentation of this process could be:

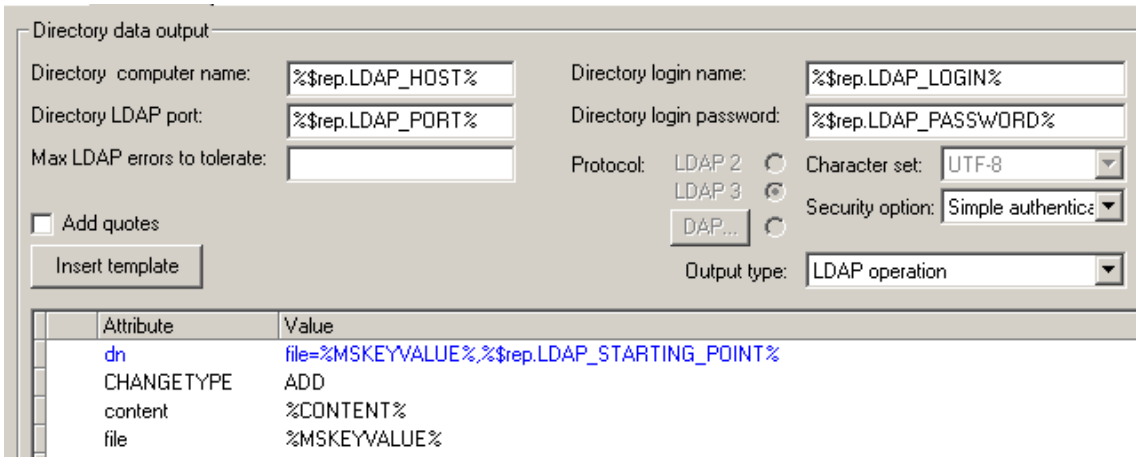


1: Account privilege adding request from the User Interface
2: Add operation request to VDS
3: Add operation request to target application
4: Success or error in target application
5: Success or error in requested operation.
If success the privilege is added. Otherwise it is not.

The three mandatory tasks for adding, modifying and deleting a user send a certain data to the Virtual Directory Server. The LDAP parameters that they share for the LDAP connection are stored in the repository definition and are:

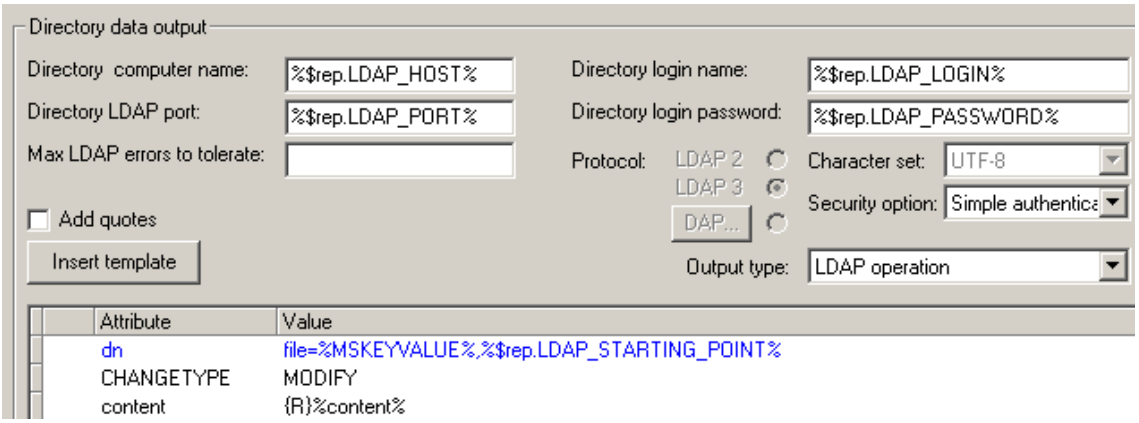
- LDAP_STARTING_POINT: Starting point where the operations are started from.
- LDAP_FILTER: Filter used for the searches.
- LDAP_HOST: Host of the Virtual Directory Server.
- LDAP_PORT: Port the Virtual Directory Server listens on.
- LDAP_LOGIN: Login for the Virtual Directory Server.
- LDAP_PASSWORD: Password corresponding to the login.

The referenced LDAP parameters may be observed in the "Destination" tab of for example the task *FS_Add user*:



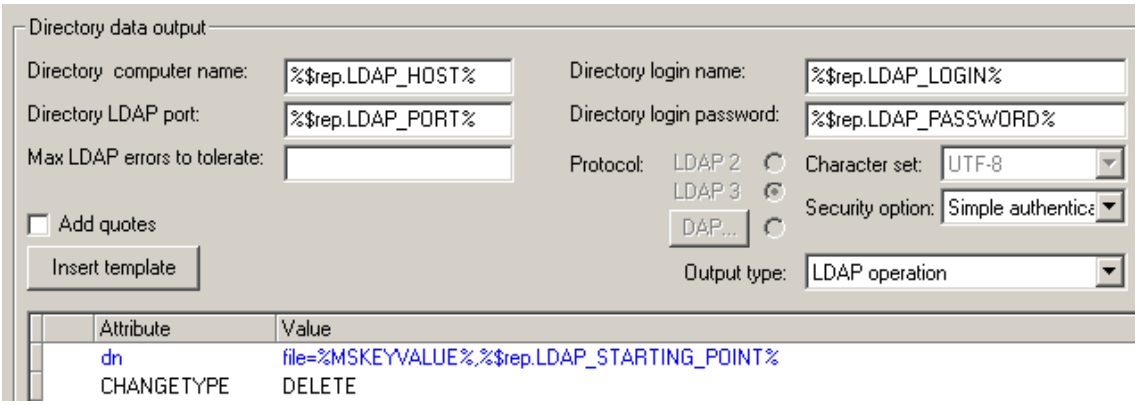
In addition, when a new file is added to the target system the file name will be the MSKEYVALUE of the corresponding entry in the Identity Center and the content will be given by the attribute CONTENT of the corresponding entry in the Identity Center. The distinguished name will be the concatenation of the RDN (file=MSKEYVALUE of the corresponding entry in the Identity Center) with the starting point given in the repository definition. The LDAP operation type is indicated by CHANGETYPE, which is ADD.

For the task *FS_Edit user* (select the "Destination" tab):



The main differences are the attributes CHANGETYPE (a MODIFY operation) and CONTENT (where the value of the attribute content will be replaced by the new value).

In the "Destination" tab of the task *FS_Delete user* the CHANGETYPE attribute has the value DELETE. The only information that the Virtual Directory Server needs is the distinguished name and that the requested operation is DELETE.



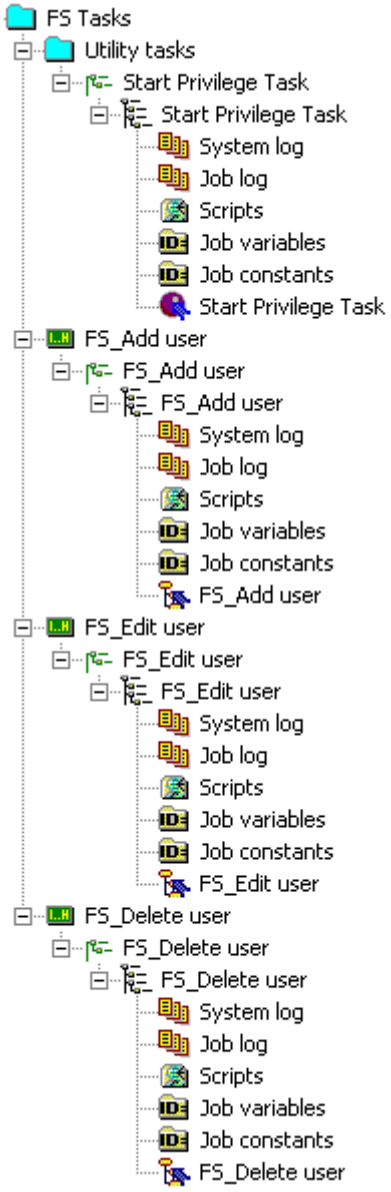
The repository definition for the sample connector (*FS*) contains the following constants:

ID= LDAP_FILTER	(objectclass=*)
ID= LDAP_HOST	localhost
ID= LDAP_LOGIN	admin
ID= LDAP_PASSWORD	**********
ID= LDAP_PORT	7777
ID= LDAP_STARTING_POINT	dir=certificationFBC,dir=testing,root=c
ID= MX_ADD_MEMBER_TASK	14/Start Privilege Task
ID= MX_DEL_MEMBER_TASK	14/Start Privilege Task
ID= MX_MODIFYTASK	10/FS_Edit user
ID= MX_PENDING_DEPROVISIONTASK	8/FS_Delete user
ID= MX_PENDING_PROVISIONTASK	12/FS_Add user

Three constants not commented yet are:

- MX_MODIFYTASK: The value of this constant is the identifier of the task *FS_Edit user*.
- MX_PENDING_DEPROVISIONTASK: The value of this constant is the identifier of the task *FS_Delete user*.
- MX_PENDING_PROVISIONTASK: The value of this constant is the identifier of the task *FS_Add user*.

In the figure below the expanded set of tasks can be viewed:



When creating the tasks, the type of tasks which the partner should use is ordered task group. Every task to be visible from the User Interface must be a public task.

Exporting the Identity Center tasks and the repository definition

Once the tasks are created the task set must be exported to the certification package. Do the following:

1. Select the folder "<RN> Tasks" in the Identity Center console tree, where <RN> is the name of the repository definition.
2. Choose "Export…" from the context menu.
3. Select the location where to save the file and name it "<Repository definition name> tasks and repository".
4. Choose "Save".
5. The "SAP NetWeaver Identity Center Syncutility" dialog box appears. Choose "Export", and then "Finish".

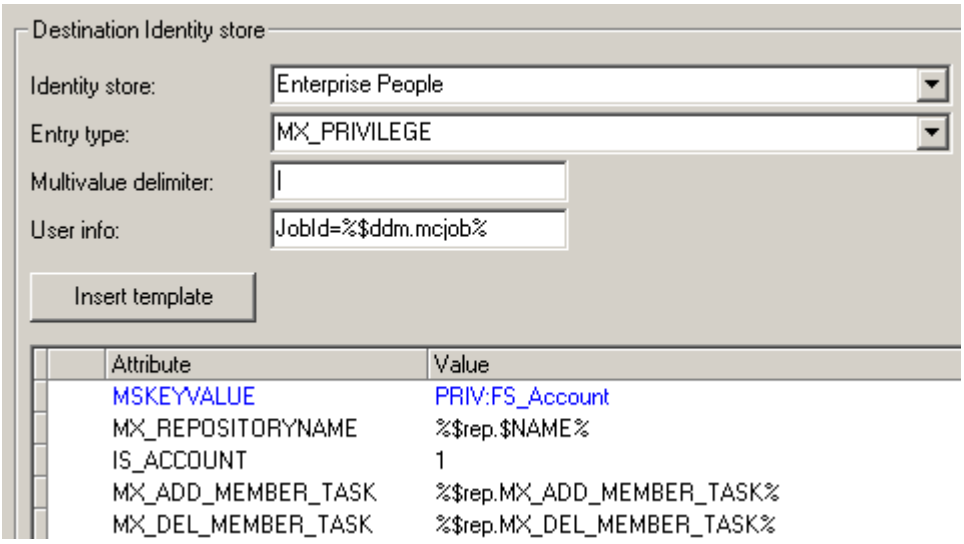
Creating the Identity Center jobs

Similar to the set of tasks, the set of jobs must follow a standard in both structure and naming. The set will be composed of two jobs:

- Create Account Privilege: This job will create the account privilege.
- Search Job: This job completes a search with one level option under the starting point given by a constant in the repository definition.

The job "Create Account Privilege"

The MSKEYVALUE for the account privilege follows the naming standard PRIV:<RN>_Account, where <RN> is the repository definition's name. In job's "Destination" tab you can observe the following set of attributes:



The values of the attributes MX_ADD_MEMBER_TASK and MX_DEL_MEMBER_TASK are retrieved from the repository definition's constants with the same name. The value of the attribute MX_REPOSITORYNAME is retrieved from the repository definition's constant NAME. The MSKEYVALUE has the value PRIV:<Repository definition name>_Account. The value of the attribute IS_ACCOUNT is "1".

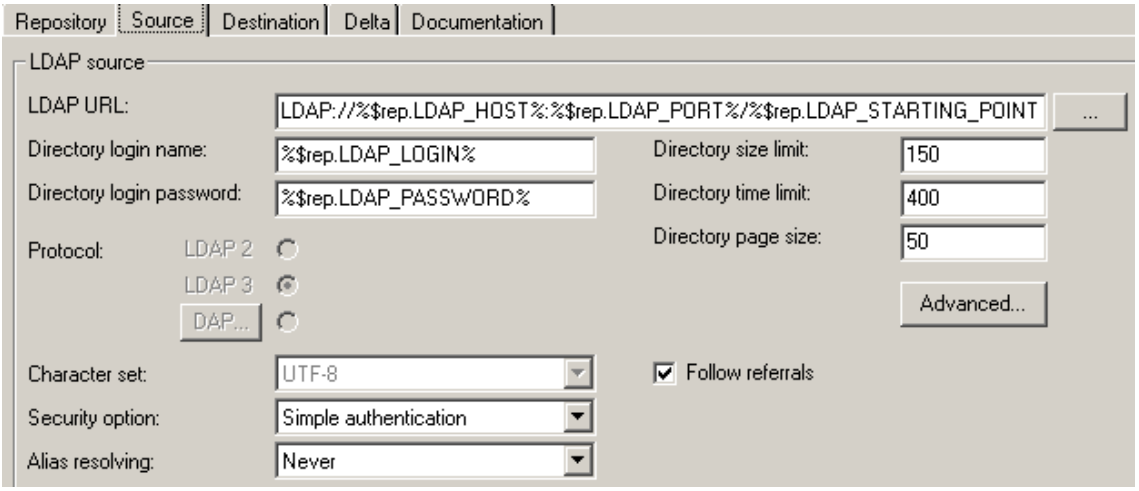
The attribute IS_ACCOUNT informs that a particular privilege is of an account type i.e. is creating an account for a user, as opposed to non-account privileges that may give users a certain rights in the backend system (but are dependent on the existence of an account).

The job "Search Job"

The job consists of two passes:

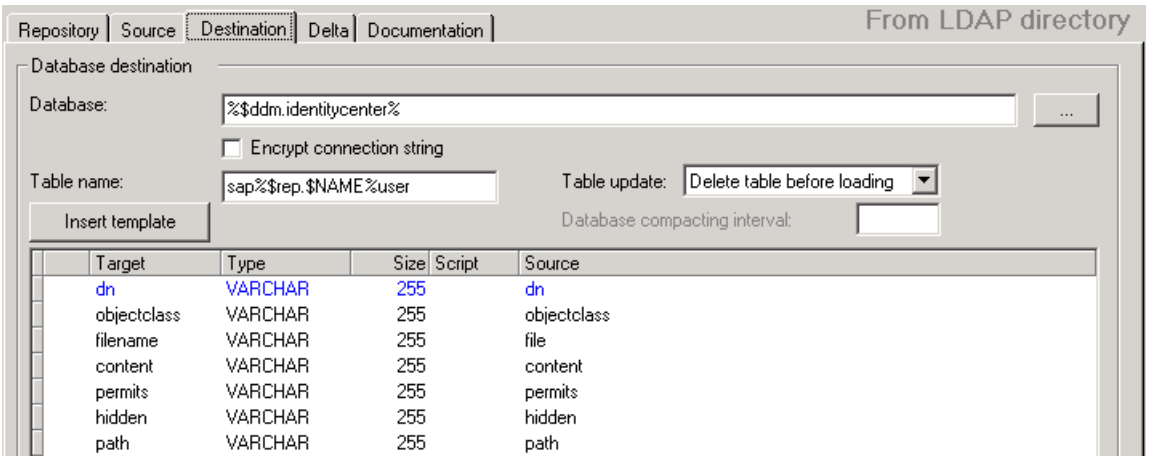
- Read Entries: This part of the job will read all the entries directly under the given starting point and will create an intermediate table where all entries will be stored.
- Write Result: The second step will read the entries from the intermediate table and write them to a file.

The *Read Entries* pass is a From LDAP directory pass that reads all entries by performing an LDAP search operation towards the Virtual Directory Server. A repository definition needs to be selected in the "Repository" tab of this pass. The tabs "Source" and "Destination" must be configured as well. The "Source" tab for the sample connector will look like this:



The three most important fields are *LDAP URL*, *Directory login name* and *Directory login password* which take their values from the *LDAP* repository definition's constants.

In the "Destination" tab it is described where the read data will be written. For the sample connector the "Destination" tab looks like this:

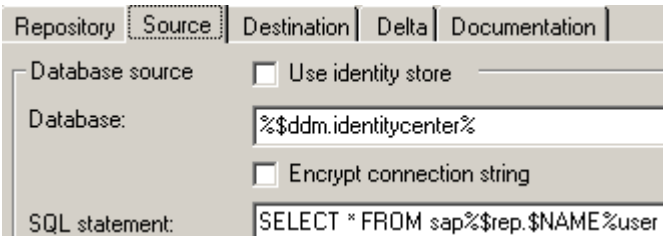


Here we define a set of attributes – what they will be named in the intermediate table, as well as their type and size.

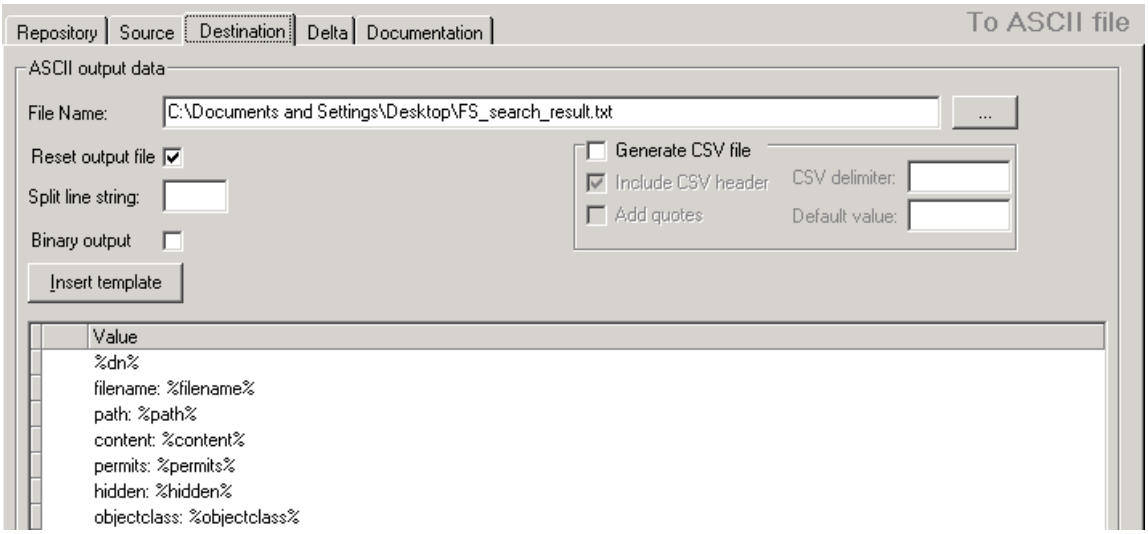
The "Database" field needs to be filled – the context menu may be used to insert the system parameter *%$ddm.identitycenter%*. Also the field "Table name" needs to be filled (use for example *sap%$rep.$NAME%user* as the value).

Once the entries are read and their data written in the intermediate table, then it is time to write this data to a file. This is done in the *Write Result* pass.

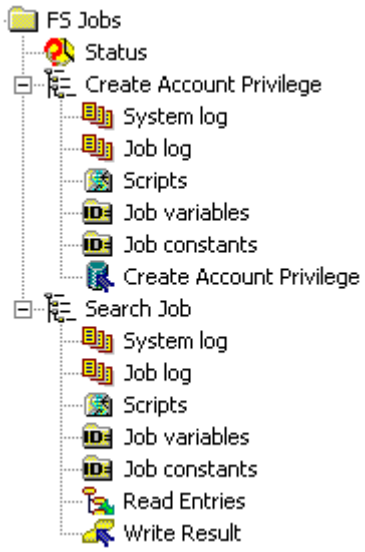
The *Write Result* pass is a To ASCII file pass. Repository definition is selected in the "Repository" tab of this pass. In the "Source" tab the table created when reading entries is defined as the data source:



In the "Destination" tab, a file to write the data to is defined and the option "Reset output file" selected:



The following figure shows the complete set of jobs for the sample connector:



Exporting the Identity Center jobs

Once the set of jobs is completed, it is time to export it to the certification package. Do the following:

1. Select the folder "<RN> Jobs" in the Identity Center console tree, where <RN> is the repository definition's name.
2. Choose "Export…" from the context menu.
3. Choose "<Repository definition name> jobs" as the name of the file and select the location where to save it.
4. Choose "Save".
5. The "SAP NetWeaver Identity Center Syncutility" dialog box appears. Choose "Export", and then "Finish".

Exporting the identity store schema (optional)

If additional entry types or attributes are required for the connector, make sure to export the identity store schema to a file and include it in the certification package (*IC files* folder). To export the schema, do the following:

1. Select "Identity store schema" of your identity store in the console tree and choose **Export schema…** from the context menu.
2. Define the name for the file (*<Identity store name>_identity store schema.mcc*) and where to store the file, then choose "Save".
3. Choose "Export", and then define which entry types and attributes to export.
4. Choose "Export" and then "Finish".

Creating the initial data for the sample connector

For the sample connector, the initial data set is composed of 1000 files. The files are called *initialfile_X* where X=[1…1000]. An initial data set provided by the partner has to have at least 300 entries. All files are stored in a subfolder *Data* of the folder called *Initial Data*. The files must be copied to the proper location as described in the corresponding documentation.

The file *Initial entries.txt* contains the entry descriptions for all the initial entries. It has the following format:

Entry_1_Dn
Attr_1_1_Name: Attr_1_1_Val
Attr_1_2_Name: Attr_1_2_Val
.
.
Attr_1_M_Name: Attr_1_M_Val
.
.
.
Entry_N_Dn
Attr_N_1_Name: Attr_N_1_Val
Attr_N_2_Name: Attr_N_2_Val
.
.
Attr_N_P_Name: Attr_N_P_Val

Section 2: Certification process steps

The certification process steps are:

- The package and documentation validation: validate that all mandatory (necessary) documentation and files are delivered by the partner and that the recommended structure and naming is followed.
- Installing and configuring the Identity Management Connector: install the Virtual Directory Server configuration (optionally vendor specific module) and configure the Identity Center.
- Functionality validation of the Identity Management Connector: test the search for, add, modify and delete user operations.

The package and documentation validation

Part of the certification process is the submission (by ISVs or SAP partners) of the files, documents and information as described in section *Deliverables* on page 2 and the certification package as shown in section *The package* on page 5.

Validate that these files and documents are delivered and that the recommended structure and naming of the files and folders in the package is followed, then enter the package and the documentation status (OK/Failed (to fulfill the requirements)) into the form (Item **1**) to be a part of the test report.

Installing and configuring the Identity Management Connector

Before installing the Identity Management connector, make sure that the Identity Center is installed and running and the Virtual Directory Server installed (optionally the vendor module). The Virtual Directory Server steps of the certification process should be replaced by the vendor module details, if it is used instead of the Virtual Directory Server.

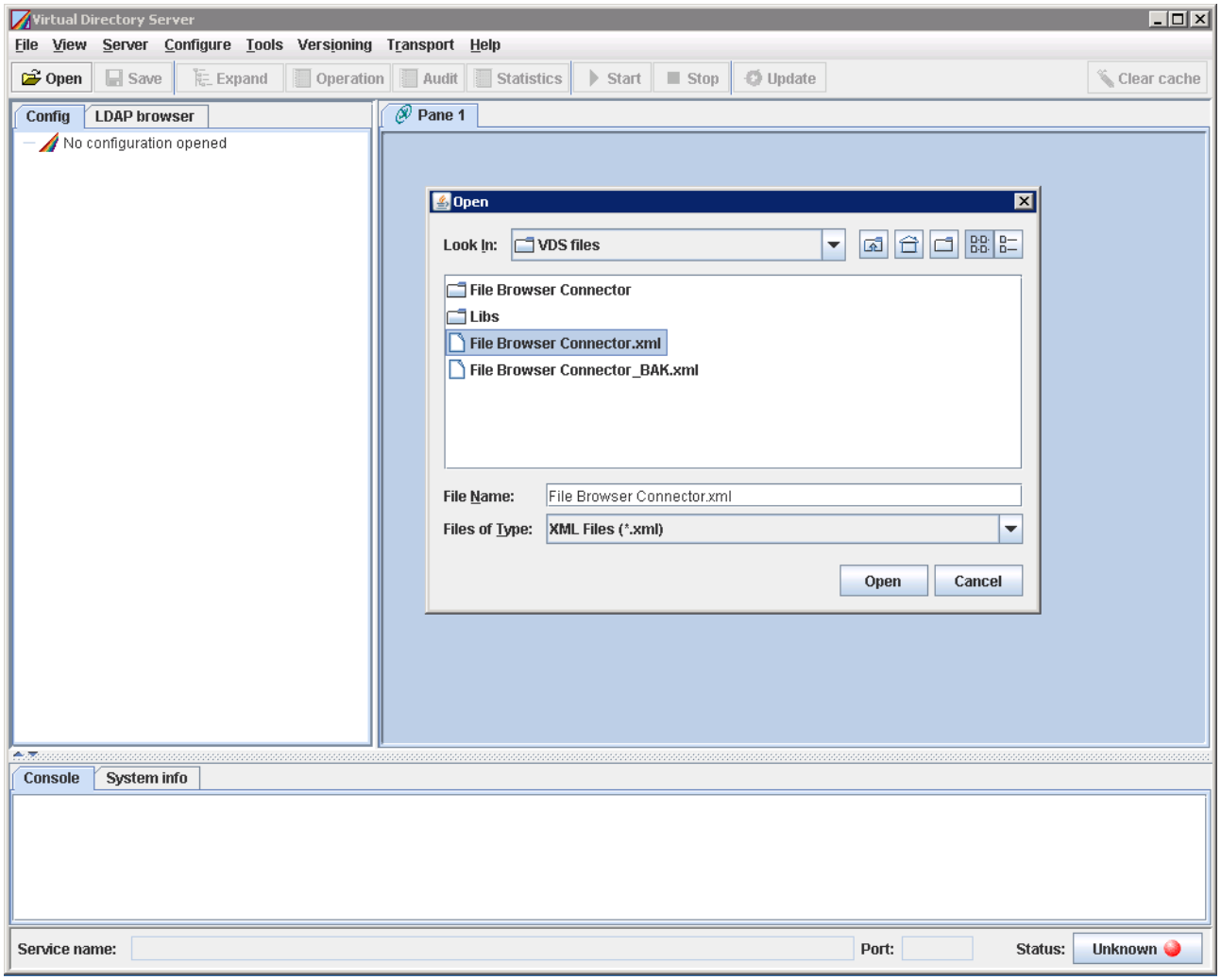
Install and configure the connector according to the installation and configuration documentation delivered by the partner.

Verify that the connector could be correctly installed (and configured) according to the documentation and enter the result (OK/Failed) into the form (Item **2**) to be a part of the test report.

Installing the Virtual Directory Server configuration

The first step is to open the Virtual Directory Server configuration:

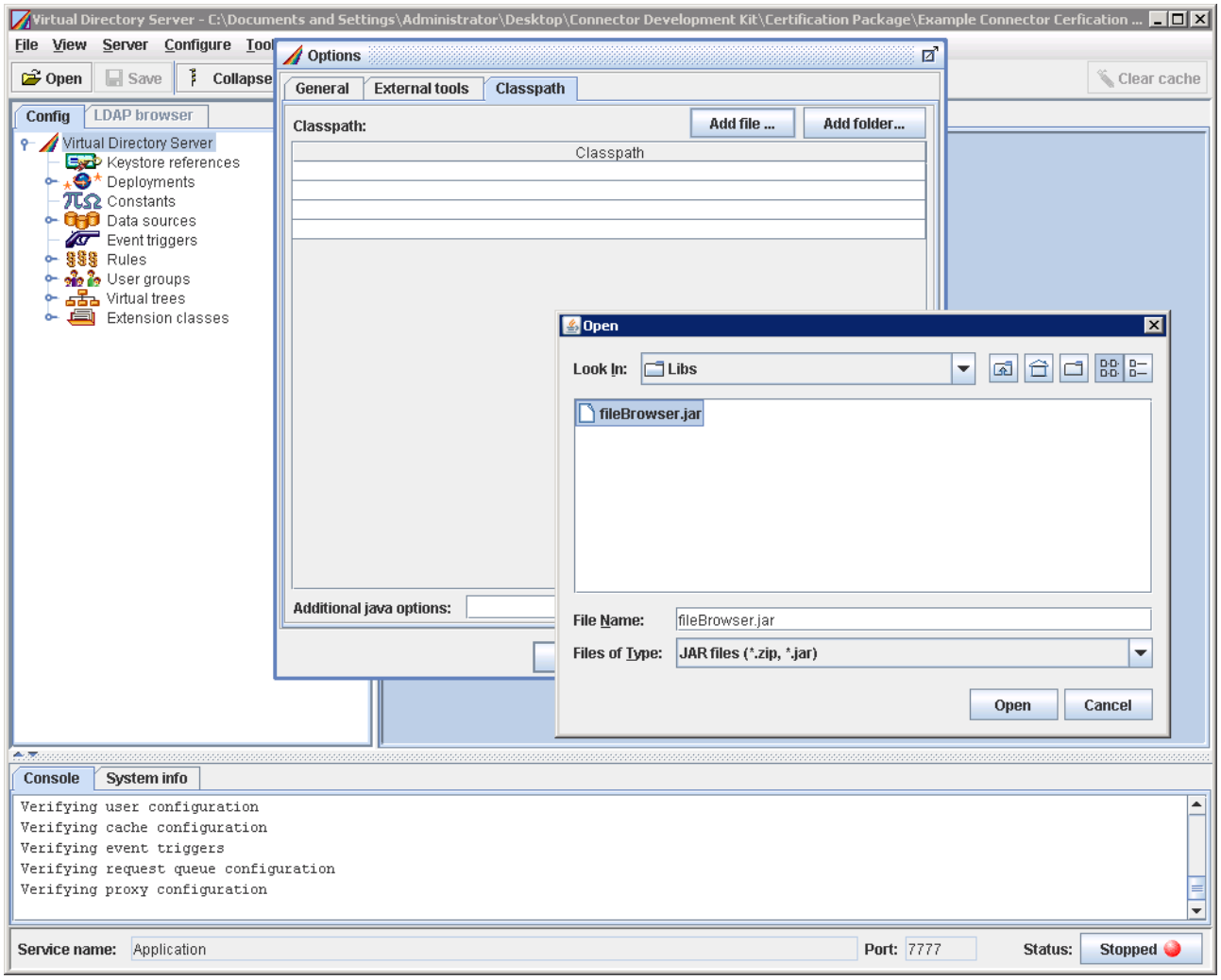
1. Open the Virtual Directory Server.
2. Select "Open file…" from "File" in the main menu.



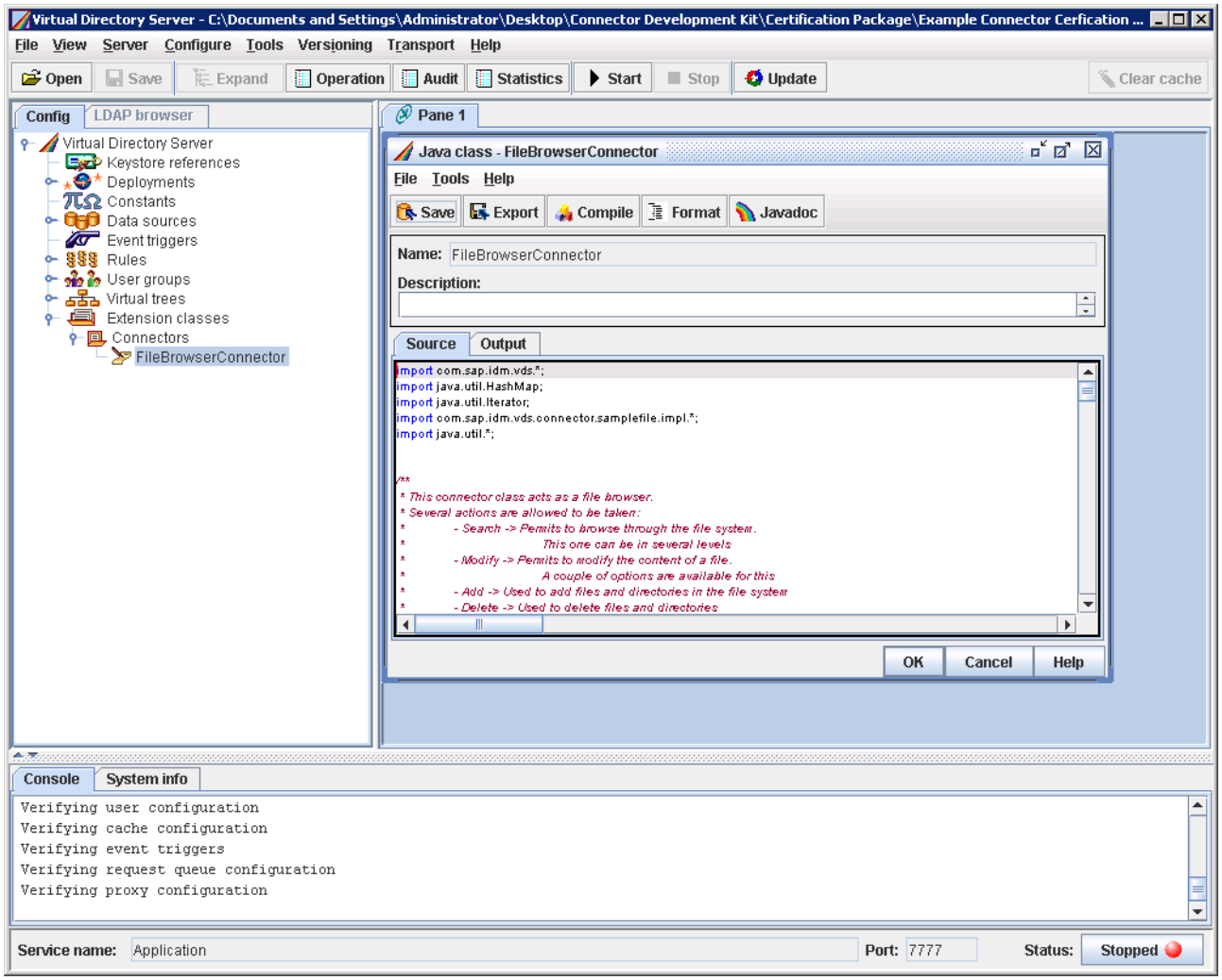
Select the file *<Connector name>.xml* (for the sample connector *File Browser Connector.xml* in the folder *VDS files* in the Connector Development Kit).

3. Choose "Open".

4. The next step consists of including the necessary target application libraries. Select "Tools" and then "Options…", and in the tab "Classpath" include all the libraries provided in the package (as e.g. *fileBrowser.jar* for the sample connector):



5. It might be necessary to compile java class(es) before the server can be run. For the sample connector, before running the server it is necessary to compile the java class *FileBrowserConnector*. To do so, extend the node "Extension classes" in the Virtual Directory Server console tree and then extend the node "Connectors". Double-click "FileBrowserConnector" and choose "Compile":



Note:
If some problems with the compilation occur, the reason might be that the java compiler is not properly configured. Select "Tools" and then "Options…", and then the tab "General" and check the compiler options.

6. Choose "Start" and the Virtual Directory Server will be running.

For some connectors, there might be some parameters that need to be set before running the server. In that case, all these must be gathered in the document covering the Virtual Directory Server configuration specifications, which should be stored in the folder *Documentation*.

Once the Virtual Directory Server is running properly the Identity Center should be configured in order to execute the use cases and finish the certification process. The Identity Center configuration consists of the following steps:

- Importing the identity store schema (optional)
- Importing the repository definition and the tasks
- Importing the jobs
- Defining modify task and modify task trigger attributes

- Creating an account privilege (in order to execute the provisioning and de-provisioning tasks)
- Other minor configurations that might be needed (in which case the Identity Center specifications must document these)

Importing the identity store schema (optional)

If additional entry types or attributes are required for the connector, make sure to import the identity store schema into the identity store. The file *<Identity store name>_identity store schema.mcc*, found in the *IC Files* folder of the certification package, must be imported. To import the file, do the following:

1. Select "Identity store schema" of your identity store in the console tree and choose **Import schema…** from the context menu.
2. Navigate to the file (*<Identity store name>_identity store schema.mcc*), then choose "Open".
3. Choose "Import", and make sure that the options "Merge" or "Overwrite", and "Use this for all matching entry types/attributes" are selected.
4. Choose "Next>" and then "Finish".

Importing the repository definition and the tasks into the Identity Center

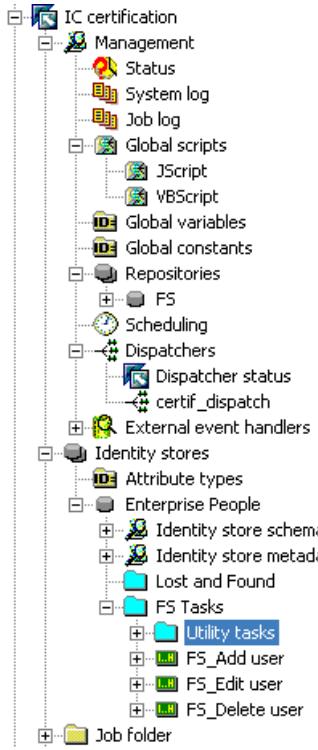
The file *<Repository definition name> tasks and repository.mcc* (which for the sample connector is the file *FS tasks and repository.mcc*) needs to be imported. Both the tasks and the repository definition are imported in the same operation.

Note:
Before importing make sure that the option "Enable imported jobs" is selected. Select the Identity Center node in the console tree and select the "Options" tab to select "Enable imported jobs". Then choose "Apply".

For the sample connector, an Identity Center called "IC certification" is created (with the dispatcher "certif_dispatch"). Then the file *FS tasks and repository.mcc* is imported. To do so, do the following:

1. Select the "Enterprise People" identity store in the console tree and select "Import…" from the context menu.
2. Navigate to and select the file *<Repository definition name> tasks and repository.mcc* (here *FS tasks and repository.mcc*). The "SAP NetWeaver Identity Center Syncutility" dialog box will open.
3. In the "General" tab, select the options "Link tasks into display- and event properties on entry types and attributes" and "Update repository constant(s)".
4. In the "Advanced" tab, select the previously created dispatcher.
5. Complete the import.

After importing the file *FS tasks and repository.mcc*, the Identity Center console tree could look like this:



The imported tasks follow a certain structure. A folder *<Repository definition name> Tasks* (*FS Tasks*) contains the tasks:

- for adding a user in the backend system, named *<Repository definition name>_Add user* (*FS_Add user*).
- for modifying a user, *<Repository definition name>_Edit user* (*FS_Edit user*).
- for deleting a user, *<Repository definition name>_Delete user* (*FS_Delete user*).

Other extra tasks must be placed into a folder called *Utility tasks*.

The repository definition *FS* holds the data needed by tasks and jobs.

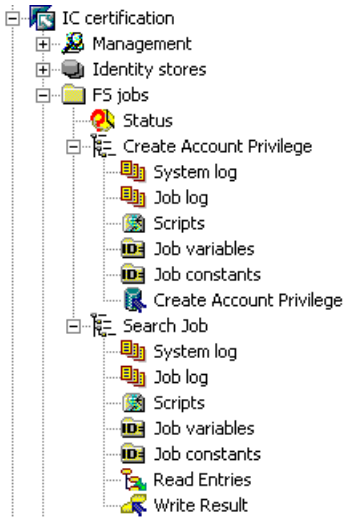
Importing the jobs into the Identity Center

The next step is to import the job folder that contains the two jobs necessary for the account privilege creation and the search operation. The first one will basically create an account privilege. Any object in the Identity Center that has the account privilege (for the target system) will have an entry in the target system. This is the privilege for provisioning/de-provisioning in the target system. The job for the search operation will search with one level option under the starting point given by the repository constant LDAP_STARTING_POINT. This job will write the results of the search in a file where the file location must be set by the user.

To import the job folder, do the following:

1. Select the Identity Center node in the console tree.
2. Select "Import…" from the context menu, and then navigate to and select the file *<Repository definition name> jobs.mcc* (here *FS jobs.mcc*).
3. Select the dispatcher for the jobs in the "Advanced" tab of the "SAP NetWeaver Identity Center Syncutility" dialog box.
4. Complete the import.

After import, the Identity Center console tree may look like this:

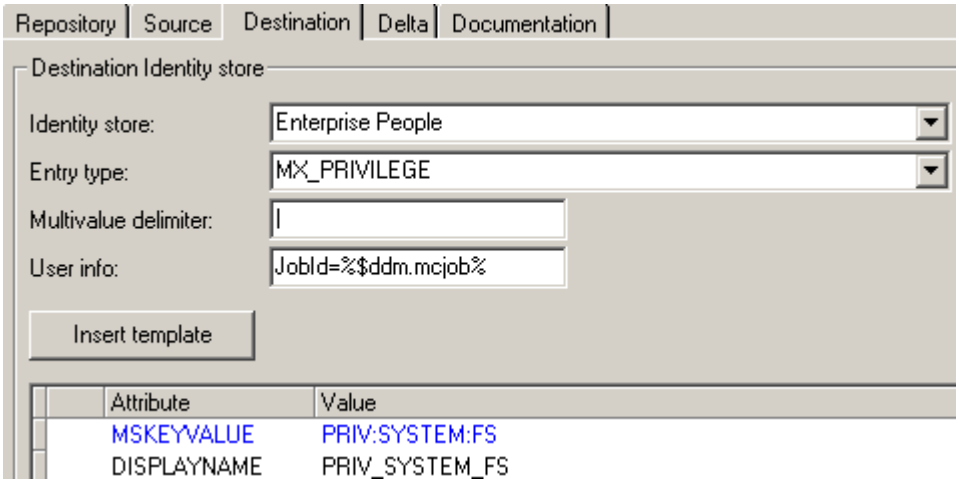


Defining modify task and modify task trigger attributes in the Identity Center

When modifying an attribute in the Identity Center, this modification will be reflected in the target system. The defined modifiable attributes must be defined as the modify task trigger attributes (with the corresponding modify task defined) in the Identity Center in order for the change to be reflected in the target system. This can be done in two ways:

- Link attribute by attribute to the modifying task: Select a modifiable attribute in the attribute list under the node "Identity store schema" in the Identity Center Management Console. View the properties of the attribute and in the "Event tasks" tab define the task "<Repository definition name>_Edit user" as the modify task. Then choose "OK". This has to be done for each modifiable attribute in the list. If the attribute is not in the list then you must create it. In order to create an attribute, select "Attributes" node and then **New\Identity store attribute…** from the context menu.

- Define the modify task and the trigger attributes on the privilege: The privilege for this purpose is a system privilege and has a standard naming PRIV:SYSTEM:<Repository definition name>. An example of how this privilege could be created is shown in the figure below:



Once the privilege is created, view the privileges under the node "Privileges" under "Identity store metadata" in the Identity Center console tree.

Another way of creating the privilege would be from the console tree in the Management Console:

Select the privilege "PRIV:SYSTEM:<Repository definition name>" and view its properties. In the tab "Tasks", define the task "<Repository definition name>_Edit user" as the modify task. Then define the modifiable attributes as the modify task trigger attributes in the list below (select "Trig event" for the desired attributes). Choose "OK" to save the data and to close the dialog box.

Creating an account privilege

Run the job "Create Account Privilege". With the execution of this job an account privilege will be created for the provisioning and de-provisioning tasks. Every entry in the Identity Center that has this privilege will be created in the target system, and when the privilege is removed from an entry in the Identity Center then the corresponding entry in the target system will be removed too.

Functionality validation of the Identity Management Connector

Before the validation testing is performed make sure that the validation job and tasks are correctly installed and configured according to documentation delivered by the partner, i.e. after the successful installation and configuration of the connector (and once the step for the account privilege creation has been completed) the validation tests validate that the following functions of the Identity Management connector work:

- Read and search for entries: reading the initial entries (provided by the partner) and testing the search functionality.
- Create a new entry (add user): testing the user provisioning to the target system/application.
- Modify an entry: testing the modifying of some attributes (which must be specified by the partner to be certified) for an entry
- Delete an entry: testing the user de-provisioning from the target system/application.

These four test cases cover the basic functionality to verify that the connector works properly.

The tasks can be executed from the SAP NetWeaver Identity Management User Interface, connected to the Identity Center, as well as from the Identity Center Management Console. For the sample connector, the test case for the search functionality will be executed from the Identity Center Management Console and others from the User Interface.

Search operation

The search job will be executed from the Identity Center Management Console. The result returned by the search job is a real data in the target system, which we can compare to the expected data.

To test this job the target system is populated with some initial data provided by the partner (can be found in the sub-folder *Data* of the folder *Initial Data* in the package). See document *<Repository definition name> initial data installation.pdf* for details about the installation of the initial data.

A file describing the entries of the initial data is available in the package. The file provided by the partner is called *Initial entries.txt* (or .csv) and can be found in the folder *Initial Data*. The number of entries must be at least 300 and the file must have the following configuration:

Entry_1_Dn
Attr_1_1_Name: Attr_1_1_Val
Attr_1_2_Name: Attr_1_2_Val
.
.
Attr_1_M_Name: Attr_1_M_Val
.
.
.
Entry_N_Dn
Attr_N_1_Name: Attr_N_1_Val
Attr_N_2_Name: Attr_N_2_Val
.
.
Attr_N_P_Name: Attr_N_P_Val

where *Entry_<X>_Dn* is the distinguished name for the entry number *X* and *Attr_<X>_<Y>_Val* is the value for the attribute *Y* of the entry *X*. The entry and the attribute representations must be exactly the same as in the file created as the result of the search operation.

The *Read Entries* pass of the search job reads the target application initial entries stored in LDAP. Make sure that the initial entries provided are correctly installed before running the job. When read, the entries are then written to a file for verification by the pass *Write Result*. In the search job it is necessary to define the desired file which the entries will be written to.

Do the following:

1. Select the pass "Write Result" of the job "Search Job" in the console tree.
2. Select the "Destination" tab.
3. Choose "…" to the right of the "File Name" field, and specify the file to be used.
4. Choose "Apply".
5. Run the job (choose "Run now" in the details pane (the "Options" tab)). The job may run for some time. When the job is completed, it is possible to see the log entry and the number of entries read.

   Note:
   You may need to refresh the log to see the result when the job is completed, either by selecting the auto refresh function or by choosing "Refresh" manually.
6. Validate that the number of entries read matches the specification from the partner and enter the result (OK/Failed) into the form (Item **3**) to be a part of the test report.
7. View the obtained result file and compare with the content of the file *Initial entries.txt* (or .csv). Verify that the first *n* rows match the specification from partner and enter the result (OK/Failed) into the form (Item **4**) to be a part of the test report.

This job will be used to verify add, modify and delete operations in the target system.

Adding the user

In this case an entry will be provisioned to the target system. The test case will be executed adding the account privilege to an Identity Center entry (entry type *MX_PERSON*). After the account privilege has been added the entry will be added to the target system. This test case must be executed for every entry type described by the partner in the documentation.

The task *<Repository definition name>_Add user* creates the user in the target application when the privilege *PRIV:<Repository definition name>_Account* is assigned the user, i.e. to run the task assign the privilege to the user by using the User Interface task *<Repository definition name>_Edit user*. Do the following:

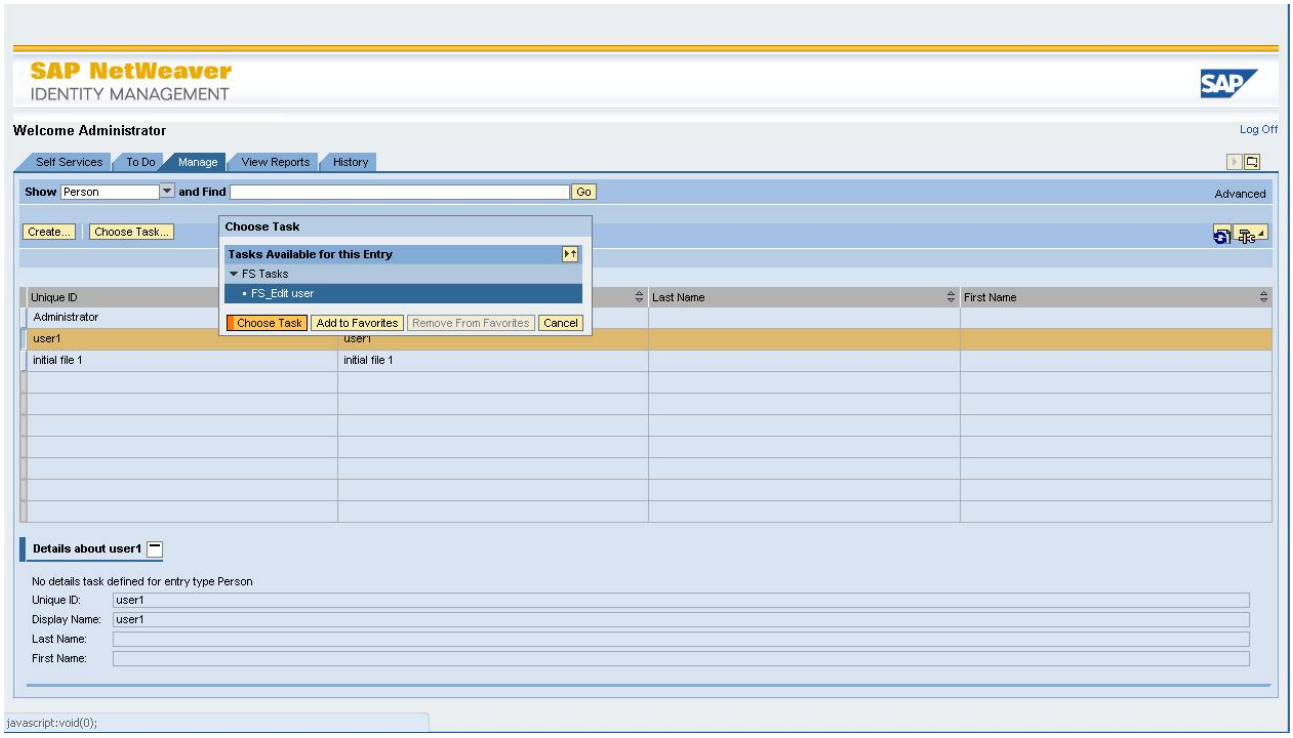
1. Start the Identity Management User Interface. Log in with the administrator user and select the "Manage" tab.
2. Find and select the "John Doe" user and choose "Choose Task".
3. Expand the "User Interface tasks" folder and select the task "<Repository definition name>_Edit user".

   Note:
   Choosing "Add to Favorites" you can add a task button for easier access to the task.
4. Choose "Choose Task" and the "<Repository definition name>_Edit user" task will open in a new window.

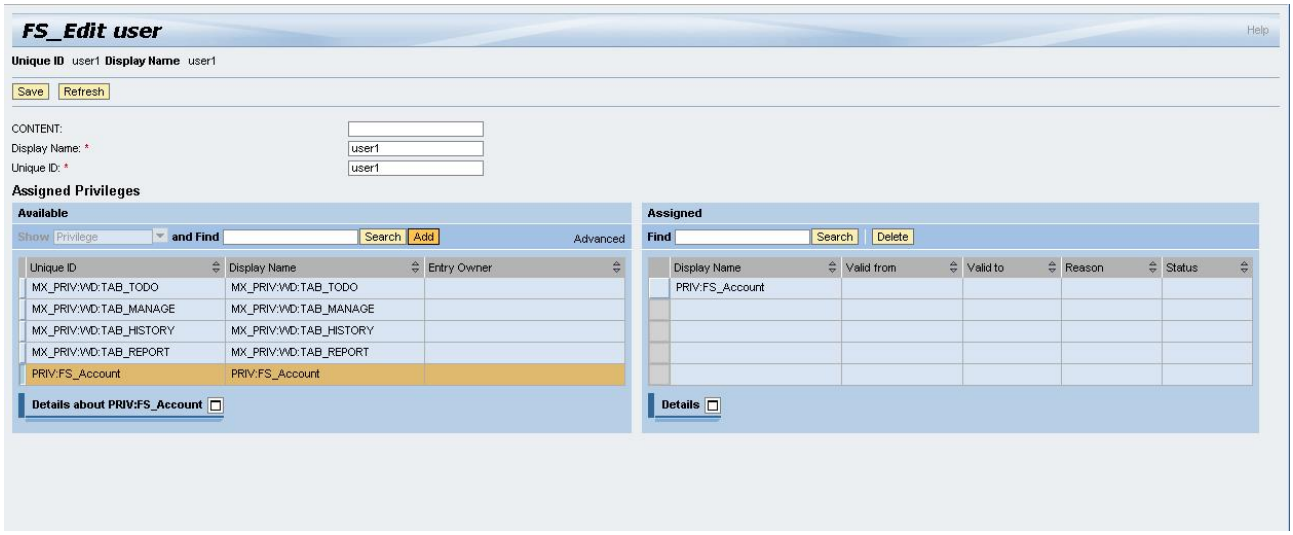
5. Choose "Search" in the left pane (**Available**) to list all the available privileges that can be assigned to the entry.
6. Select the privilege "PRIV:<Repository definition name>_Account" and choose "Add". The privilege is now added.
7. Choose "Save" to save the changes and close the task.
8. Verify, according to specification from partner, that the user "John Doe" is created in the target application. Use the search job to verify that the entry has been added, i.e. inspect the generated search result file checking for the existence of the entry. Also check that no unexpected extra entries, missing entries or attribute errors/modifications have emerged. Enter the result (OK/Failed) into the form (Item **5**) to be a part of the test report.

An example using the sample connector, where a User Interface task is used to add the account privilege to entry "user1", is:

1. Select the entry "user1" in the User Interface (the "Manage" tab).
2. Choose "Choose Task"
3. Navigate to and select the task "FS_Edit user".



4. Choose "Choose Task". This will open the task in own window.



Add the privilege "PRIV:FS_Account" to the user.

5. Choose "Save", then close the task window.

The entry "user1" should now be provisioned. The next step is to check that the file was added, using the search job in the Identity Center Management Console, i.e. run the search job and manually inspect the generated search result file checking for the existence of the entry. Also check that no unexpected extra entries, missing entries or attribute errors/modifications have emerged.

Modifying the user

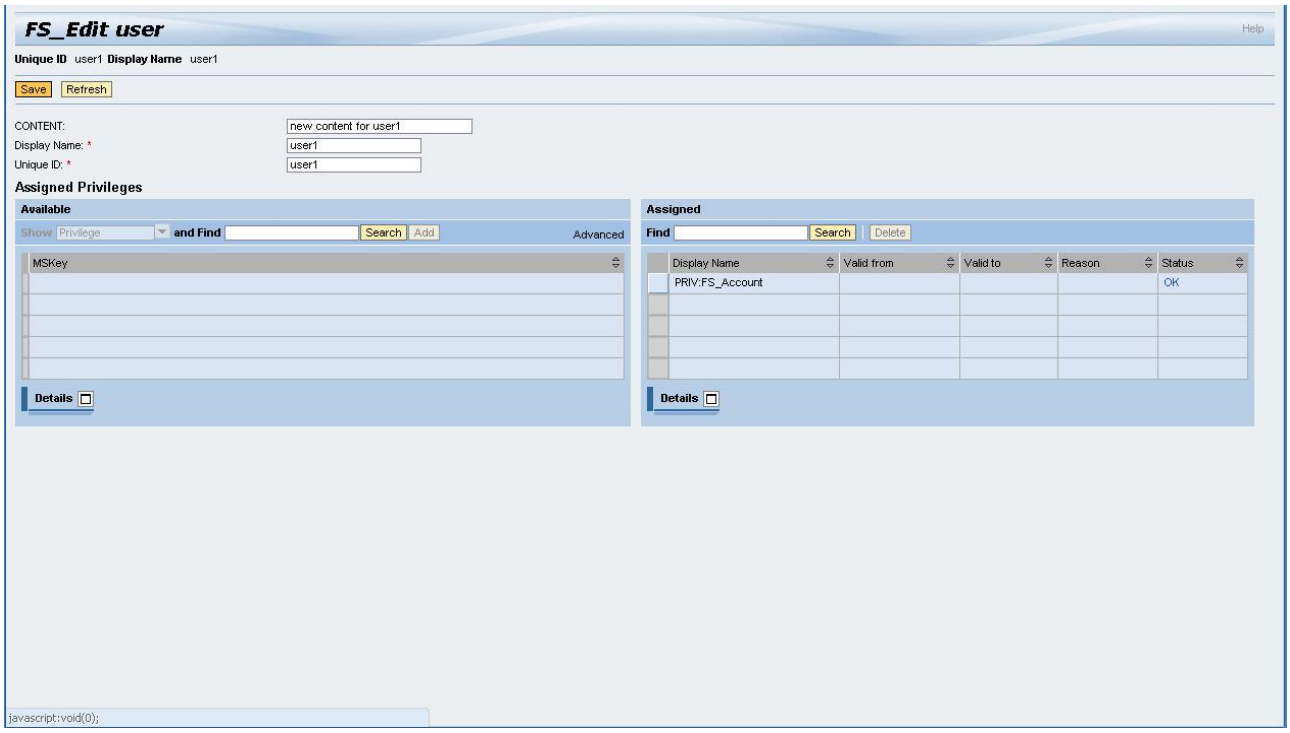
In this case some modifications will be made for every attribute described by the partner in the documentation. The partner must describe the properties of every attribute as an attribute can for instance be mandatory, single-value, multi-value, etc. Use the User Interface task *<Repository definition name>_Edit user* to modify one or several attribute(s) of user *John Doe* (a telephone number (attribute *MX_PHONE_PRIMARY*), e-mail address (attribute *MX_MAIL_PRIMARY*) etc). Do the following:

1. Select the user "John Doe" in the "Manage" tab of the User Interface, then select "<Repository definition name>_Edit user". This will open the task in a new window.
2. Add or alter value of an attribute in the correct field.
3. Choose "Save" to save the changes and close the task.
4. Verify, according to specification from partner, that these attributes are updated in the target application. Use the search job to verify that the entry/attribute has been modified (inspect the generated search result file checking the modification). Also check that no unexpected extra entries, missing entries or attribute errors/modifications have emerged. Enter the result (OK/Failed) into the form (Item **6**) to be a part of the test report.

In the sample connector there is only one attribute to modify – the attribute *CONTENT*. Make sure that the attribute exists in your identity store, and add the attribute to the attribute list of the User Interface task *FS_Edit user*.

To modify this attribute, do the following:

1. Select the entry "user1" in the User Interface (the "Manage" tab) and open the task "FS_Edit user" for this entry.



   To modify the attribute "CONTENT", enter the data in the "CONTENT" field.

2. Choose "Save" and close the task window.
3. Compare the expected data with the result obtained by the search job.

Deleting the user

This will test the de-provisioning of entries in the target system. Remove the account privilege from an entry that has it and the entry should be removed from the target system. The task *<Repository definition name>_Delete user* deletes the user from the target application when the privilege *PRIV:<Repository definition name>_Account* is removed from the user, i.e. to run the task remove the privilege from the user by using the User Interface task *<Repository definition name>_Edit user*. Do the following:

1. Select the user "John Doe" in the "Manage" tab of the User Interface, then select "<Repository definition name>_Edit user". This will open the task in a new window.
2. Select the privilege "PRIV:<Repository definition name>_Account" in the right pane (**Assigned**) and choose "Delete". This will remove the privilege.
3. Choose "Save" to save the changes and close the task.
4. Verify, according to specification from partner, that the user "John Doe" is no longer present in the target application. Use the search job to verify that the entry has been removed (inspect the generated search result file checking that the entry is removed). Enter the result (OK/Failed) into the form (Item **7**) to be a part of the test report.

For the sample connector, the process of de-provisioning will be the opposite of the one described for adding the user. In this case the account privilege *PRIV:FS_Account* must be removed from the entry *user1*, using the User Interface task *FS_Edit user*. Compare the expected data with the result obtained by the search job.

Section 3: Test report (form)

This table/form is used to log the results of the tests and will be a part of the final test report:

Item nr.	Name		Validation focus	Status (OK/Failed)	Comments
1	**Documentation/the package**		Check that the documentation exists/is provided. Also check that the recommended structure and naming of the files and folders in the package is followed		
2	**Technical implementation and configuration**		Check the installation process and the customizing/configuring as documented, and verify that it works.		
	Functional correctness:		Check the functional correctness of selected use cases.		
		Use cases:			
3		*Number of entries read*	Check that the search operation result file contains the correct number of entries read from the repository definition, i.e. contains all entries available in the target application.		
4		*Entries match*	Check that entries in the search operation result file match those in the target application (as described in the partner documentation).		
5		*Add (create) the user*	Check that the user is created (exists) in the target application.		
6		*Modify the user*	Check that the modified attributes are correctly updated in the target application.		
7		*Delete the user*	Check that the user is removed from the target application.		