EVIDEN

Identity and Access Management

DirX Identity

Service Management

Version 8.10.14, Edition March 2026

All product names quoted are trademarks or registered trademarks of the manufacturers concerned.

© 2026 Eviden

All Rights Reserved

Distribution and reproduction not permitted without the consent of Eviden.

Table of Contents

Preface

This document describes a set of use cases that explain how to use specific features of DirX Identity. It helps users to model their use case with DirX Identity and to set up and run their DirX Identity system.

The goal of this document is to explain the integration of service management into DirX Identity.

It consists of the following chapters:

- Chapter 1 provides an overview on the described use cases.
- Chapter 2 elaborates about DirX Identities internal ticket processing.
- Chapter 3 explains how to work with Service Management as a source
- Chapter 4 explains how to provision offline systems through manual provisioning

DirX Identity Documentation Set

*Version 8.10.14 | Build 1858 | Date 2026-03-26 *

The DirX Identity document set consists of the following manuals:

- *DirX Identity Introduction*. Use this book to obtain a description of DirX Identity architecture and components.
- *DirX Identity Release Notes*. Use this book to understand the features and limitations of the current release. This document is shipped with the DirX Identity installation as the file **release-notes.pdf**.
- *DirX Identity History of Changes*. Use this book to understand the features of previous releases. This document is shipped with the DirX Identity installation as the file **history-of-changes.pdf**.
- *DirX Identity Tutorial*. Use this book to get familiar quickly with your DirX Identity installation.
- *DirX Identity Provisioning Administration Guide*. Use this book to obtain a description of DirX Identity provisioning architecture and components and to understand the basic tasks of DirX Identity provisioning administration using DirX Identity Manager.
- *DirX Identity Connectivity Administration Guide*. Use this book to obtain a description of DirX Identity connectivity architecture and components and to understand the basic tasks of DirX Identity connectivity administration using DirX Identity Manager.
- *DirX Identity User Interfaces Guide*. Use this book to obtain a description of the user interfaces provided with DirX Identity.
- *DirX Identity Application Development Guide*. Use this book to obtain information how to extend DirX Identity and to use the default applications.
- *DirX Identity Customization Guide*. Use this book to customize your DirX Identity environment.
- *DirX Identity Integration Framework*. Use this book to understand the DirX Identity framework and to obtain a description how to extend DirX Identity.
- *DirX Identity Web Center Reference*. Use this book to obtain reference information about the DirX Identity Web Center.
- *DirX Identity Web Center Customization Guide*. Use this book to obtain information how to customize the DirX Identity Web Center.
- *DirX Identity Meta Controller Reference*. Use this book to obtain reference information about the DirX Identity meta controller and its associated command-line programs and files.
- *DirX Identity Connectivity Reference*. Use this book to obtain reference information about the DirX Identity agent programs, scripts, and files.
- *DirX Identity Troubleshooting Guide*. Use this book to track down and solve problems in your DirX Identity installation.
- *DirX Identity Installation Guide*. Use this book to install DirX Identity.

- *DirX Identity Migration Guide*. Use this book to migrate from previous versions.

Notation Conventions

Boldface type
In command syntax, bold words and characters represent commands or keywords that must be entered exactly as shown.

In examples, bold words and characters represent user input.

Italic type
In command syntax, italic words and characters represent placeholders for information that you must supply.

[ ]
In command syntax, square braces enclose optional items.

{ }
In command syntax, braces enclose a list from which you must choose one item.

In Tcl syntax, you must actually type in the braces, which will appear in boldface type.

|
In command syntax, the vertical bar separates items in a list of choices.

...
In command syntax, ellipses indicate that the previous item can be repeated.

userID_home_directory
The exact name of the home directory. The default home directory is the home directory of the specified UNIX user, who is logged in on UNIX systems. In this manual, the home pathname is represented by the notation *userID_home_directory*.

install_path
The exact name of the root of the directory where DirX Identity programs and files are installed. The default installation directory is *userID_home_directory*/**DirX Identity** on UNIX systems and **C:\Program Files\DirX\Identity** on Windows systems. During installation the installation directory can be specified. In this manual, the installation-specific portion of pathnames is represented by the notation *install_path*.

dirx_install_path
The exact name of the root of the directory where DirX programs and files are installed. The default installation directory is *userID_home_directory*/**DirX** on UNIX systems and **C:\Program Files\DirX** on Windows systems. During installation the installation directory can be specified. In this manual, the installation-specific portion of pathname is represented by the notation *dirx_install_path*.

dxi_java_home
The exact name of the root directory of the Java environment for DirX Identity. This location is specified while installing the product. For details see the sections "Installation" and "The Java for DirX Identity".

tmp_path

The exact name of the tmp directory. The default tmp directory is /tmp on UNIX systems. In this manual, the tmp pathname is represented by the notation *tmp_path*.

tomcat_install_path
The exact name of the root of the directory where Apache Tomcat programs and files are installed. This location is defined during product installation.

mount_point
The mount point for DVD device (for example, **/cdrom/cdrom0**).

1. Overview

Identity management comprises the integration of source systems as well as target systems. Setup of such an integration can cost a lot of effort and can require much time.

Many customers have external service management systems in place (for example ticket systems like Remedy or HP OpenView) that manage all administration tasks with a group of administrators by hand.

To speed up the integration of an Identity system a common strategy is:

- Integrate only the most important target systems tightly through automatic provisioning. Typical candidates are for example Active Directory, IBM Notes and SAP ECC UM.
- Integrate the existing ticket system as a source system for DirX Identity and process these tickets automatically.
- Use DirX Identities feature for manual provisioning to integrate all or many of the existing target systems.

This approach is much faster and brings more benefit to customers as if you try to integrate all source and target systems tightly.

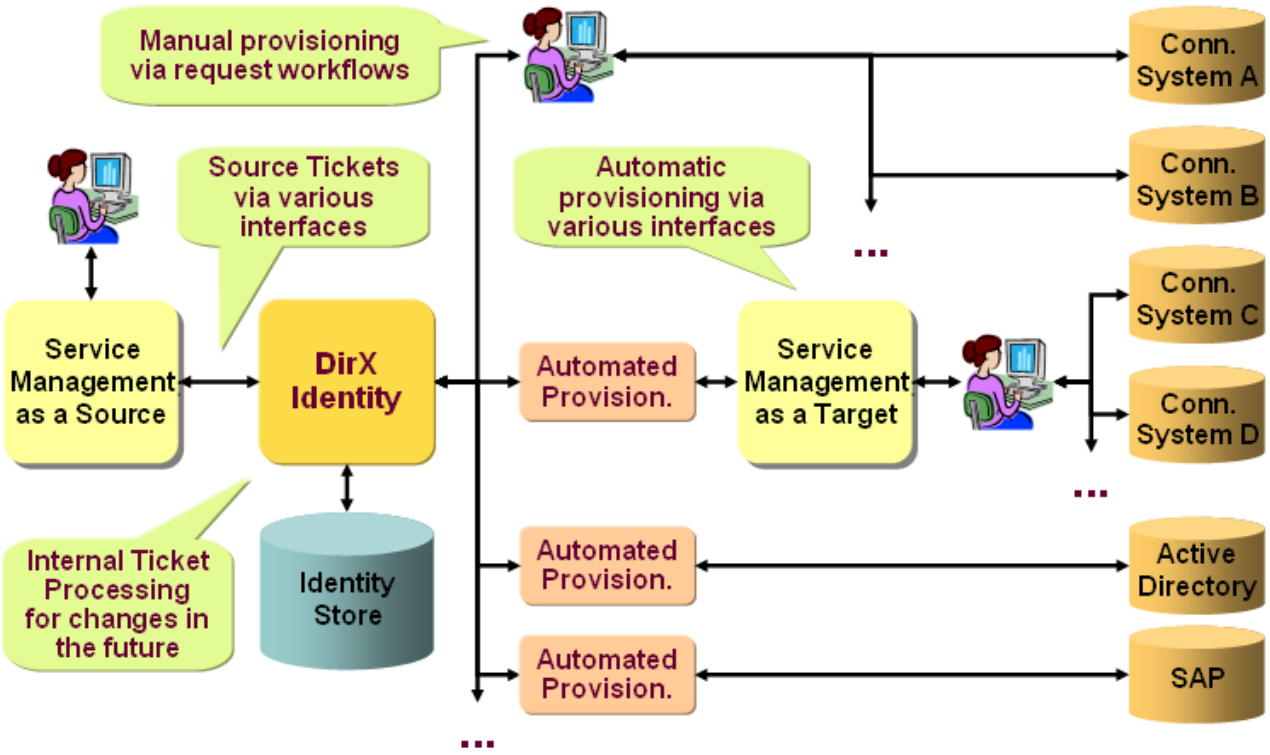
Additionally time related processes require an internal service management component in the identity management itself. For example the move of a user from one organizational unit to another one or from one location to another one is well-defined and happens at a specific due date. The identity system must be able to 'buffer' change requests for some time as orders or tickets.

This document explains the typical use cases and their application in customer environments.

Note that we do not repeat information that is available in other parts of the DirX Identity documentation. Instead, we reference it.

1.1. Use Cases

The following figure shows all relevant use cases regarding service management or ticket systems.



1.1.1. Service Management as a Source

If a customer has a ticket system already in place that is used for any type of request, it makes sense to reuse that system as a source for orders. Use a specific subset of tickets to trigger specific identity management actions. Examples of such tickets are user creation or modification requests or requests for privileges.

1.1.2. Internal Ticket Processing

Most actions in an identity management system are expected to happen immediately in real-time but there are situations where a change is requested for some specific time in the future. Assume a move of a person from an organizational unit or a location to another one.

In the first case the organization data like the department number and related data changes at the date where the move happens, in the second case the location data like country, city and room number and other related data changes.

This information is stored in the format of DirX Identity's internal tickets and processed at the right time.

1.1.3. Service Management as a Target

At the backend customers typically have lots of connected systems that have to be managed. Some of them you can connect with automatic provisioning but this makes only sense if the amount of users (accounts) in these systems is more than 1000 users and the frequency of change is high.

Typically the customer has a number of administrator or administrator groups in place that manage systems with only few users or low change frequency manually.

If the customer has a ticket system in place that is already used for manual provisioning of these connected systems, use automatic provisioning via tickets. Local administrators work on these tickets from the ticketing system and confirm the tasks.

Note: Currently DirX Identity has no default connectors to commonly used ticket systems (Remedy, HP Open View). Thus you need to implement a custom connector within your customer project.

1.1.4. Manual Provisioning of Offline Systems

If the customer has no ticket system in place, you can use the manual provisioning approach via DirX Identity request workflows. In this case the provisioning workflows for synchronization establish for each event a request workflow. The administrators of the target systems get these add, modify and delete requests, perform them manually and confirm the task. This sets the according states in the target system objects (accounts, groups and memberships).

2. Service Management as a Source

This chapter describes the use case where an already established service management system (ticket system) at the customer site is used as input for identity management relevant actions. Examples are creation and modify requests for users or privileges or requests for privilege assignment.

2.1. Detailed Definition

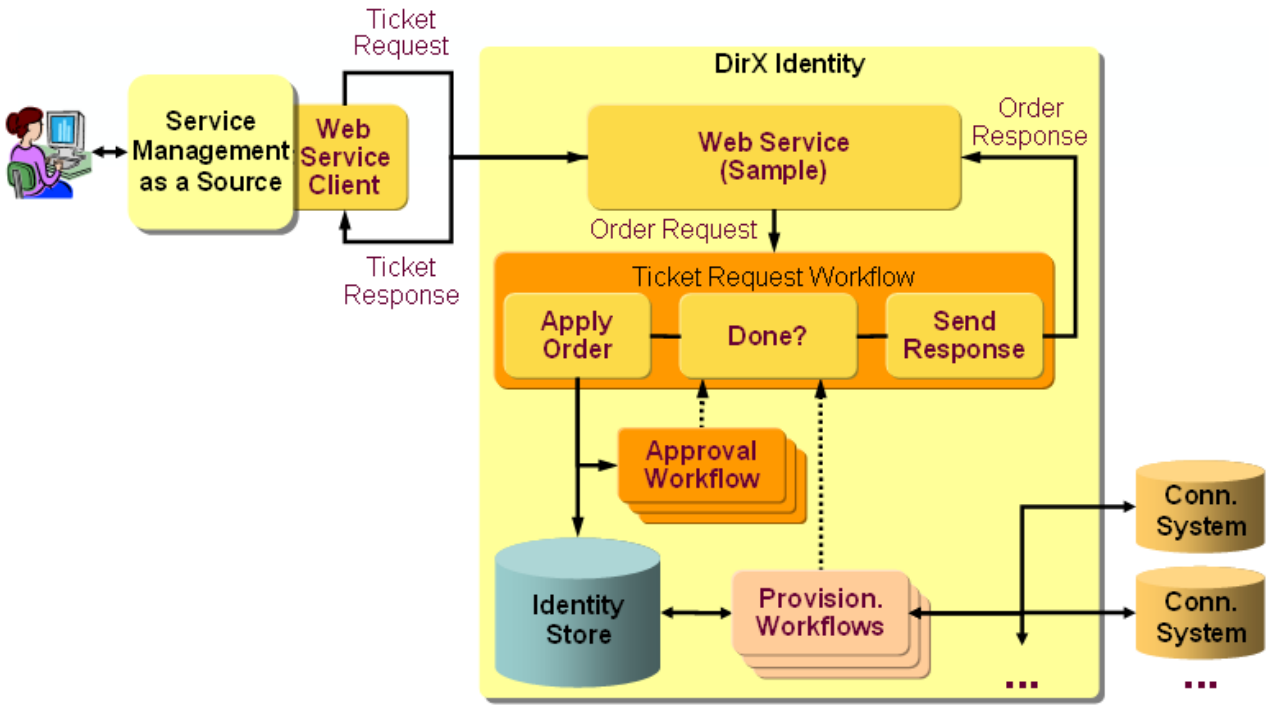
This use case assumes that the customer has already a comprehensive ticketing or service management system in place that is used for all kinds of requests as there are for example:

- Orders for new hardware
- Orders for new software or software upgrades
- Requests for any type of material to be ordered, for example office utilities, printer cartridges.
- Requests for access to specific resources for example file shares or special functionality in software systems (for example Active Directory, SAP).

Especially the last type is subject to be managed fully automatically by an identity management system. This can help automating the provisioning process and is also important for complete auditing in that area.

Because the employees of the company are used to this ticketing system and its user interface it might be difficult to replace it with a new user interface of the identity management system. Thus a better approach is to connect the ticketing system with DirX Identity.

The next figure shows the scenario in more detail:



This scenario assumes that the ticketing system (Service Management as a Source) is integrated via a web service interface:

- A user creates a ticket in the service management system.
- The system detects that this is a topic that can be sent to the Identity Management system. It generates a web service request (the **Ticket Request**) and sends it to the corresponding **Web Service**.
- The Web Service processes the ticket and converts it into DirX Identity's internal order format and starts the relevant **Ticket Request Workflow** if necessary.
- Next it sends a **Ticket Response** with the result (ticket processing ok or not ok) to the calling client. The ticket request operation is complete.
- In DirX Identity the Ticket Request Workflow is processed. This is completely configurable by the customer. The workflow can start sub workflows (typically **Approval Workflows** for assignment approval) and can indicate provisioning requests that result in **Provisioning Workflow** activities.
- The Ticket Request Workflow can be configured to watch the completion of the sub workflows as well as the completion of all provisioning requests (activity **Done?**).
- After completion a **Send Response** activity could actively send a response to the calling client but normally the client is not prepared receiving not requested responses. Instead of that the Service Management system sends status requests either from time to time to check the open tickets or if the user wants to know the ticket status.

As stated above, the described scenario assumes that a web service interface is used between the Service Management system and DirX Identity. The DirX Identity DVD comes with a sample of such an interface that can be used to interface a specific Service Management system in a customer environment. Read the follow-on tutorial "Working With Source Tickets" in the *DirX Identity Tutorial* to perform the example use case in the sample domain to understand the feature.

Of course there are other interface technologies that could be used:

- File-based interfaces in various formats (CSV, LDIF etc.)
- JMS-based interfaces
- A pure Java interface

We recommend using a web service interface.

2.1.1. Evaluation of this Use Case

This section elaborates on advantages and disadvantages of this solution.

Advantages

- User interface for end users does not change.
- End users have the same interface for all requests in the company.
- Processes in the company stay almost the same.

Disadvantages

- Integration between the ticketing system and DirX Identity requires considerable effort.

2.1.2. Additional Documentation

Besides the information in this document you can find additional documentation in the DirX Identity documentation:

- Read the follow-on tutorial "Working With Source Tickets" in the *DirX Identity Tutorial*. Perform the example use case in the sample domain to understand the feature.
- Read the description of the "Process Ticket Workflow" workflow in the *DirX Identity Application Development Guide*.
- Find JavaDoc for our sample source ticket application on the DVD in the folder: Documentation → DirXIdentity → ServiceMM → SampleSourceTicketApplication.

2.2. Set Up and Configuration

Perform these tasks:

- Implement an interface between the ticketing system and DirX Identity. Use the sample interface delivered with DirX Identity on the DVD.
- Configure one or more corresponding Process Ticket workflows in the Provisioning view that handle ticket requests.
- Test the solution thoroughly.

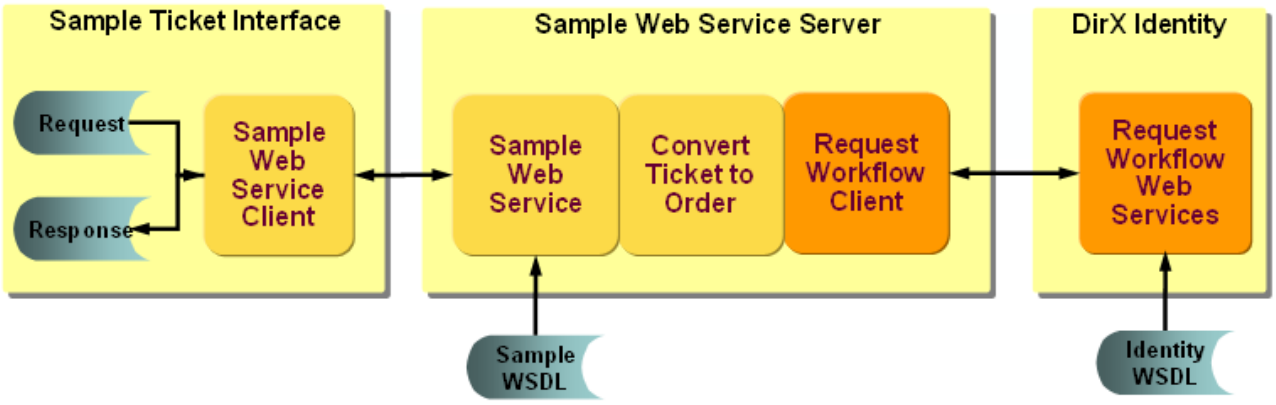
These parts must fit together to work correctly.

2.2.1. Implementing the Interface

Before we explain how to implement the interface, we explain the sample interface that comes with DirX Identity.

2.2.1.1. Understanding the Sample Interface

The sample web service interface architecture delivered with DirX Identity in the folder **Additions → WorkingWithSourceTickets** is shown in the next figure:



It comprises these parts:

- DirX Identity provides **Request Workflow Web Service** that allow starting and controlling request workflows. This is an official interface of DirX Identity described by a WSDL. For more information see the chapter "Workflow Management Web Services" in the *DirX Identity Integration Framework Guide*.
- The **Sample Web Service** shows how a custom web service interface could work. It can run either as standalone application or inside Tomcat. The **Sample Web Service** demonstrates how a custom web service could behave. It has its own WSDL. It is a slightly enhanced SPML format. It consists of:
- The **Request Workflow Client** interface that uses the Request Workflow Web Service.
- The client gets orders from the **Ticket Converting** module. This component builds orders from the incoming tickets and vice versa. The order format must comply with the format that is required by DirX Identity.

To allow easy generation and modification of various ticket formats, a **Sample Ticket Interface** is provided. It reads requests from files, passes them to the Sample Web Service and returns responses.

The sample web service delivery comprises these parts:

- A **lib** folder that contains the sample web service and the sample client.
- The **Sample Web Service** that was derived from SPML and is located between the service management system and DirX Identity. The web service can be started in two ways:
- A start script for standalone application: **runServer.bat** (Windows only).

- A **sourceTicketing.war** file that enables to deploy and run the web service in Tomcat.
- A start script to run the **Sample Web Service Client**: **ticketClient.bat** (Windows only). This client simulates a customer service management system that sends tickets to the sample web service and receives the corresponding responses.
- A script that executes the client with a specific sample ticket: **CreateUserWithRoles.bat** processes the ticket **sampleTickets/CreateUserWithRoles.xml**. See the DirX Identity Tutorial's section 'Working with source tickets' for a detailed description of this use case.
- Another script **CheckStatus.bat** executes the **sampleTickets/CheckStatus.xml** ticket to evaluate the status of the previous request.
- Find more sample tickets in the folder **sampleTickets**. The tickets work in the My-Company sample domain and comprise:
- Creation tickets for users (**createUser.xml**), roles (**createRole.xml**), permissions (**createPermission.xml**) and groups (**createGroup.xml**).
- Creation tickets for privilege assignments (**addRoleAssignment.xml** and **addPermissionAssignment.xml**).
- A creation ticket for a user with assignments: **createUserAddRoles.xml**.
- Modification tickets for users (**modifyUser.xml**) and roles (**modifyRole.xml**).
- A modification ticket for a role assignment (**modifyRoleAssignment.xml**).
- Deletion tickets for users (**deleteUser.xml**), roles (**deleteRole.xml**) and permissions (**deletePermission.xml**).
- Deletion tickets for privilege structures (**deleteRoleAssignment.xml** and **deletePermissionAssignment.xml**).
- A sample for a cancel requests for a role assignments (**cancelAddRoleAssignment.xml**).
- A status request for a user creation request (**statusCreateUser.xml**).

It is easy to modify the tickets so that they are applicable for your domain but it is not very probable that the file based SPML-like XML interface is what you need for interfacing your service management system to DirX Identity.

The sample web service command line is (can be displayed with runServer.bat –help):

runServer.bat [-host *host*] [**-port** *port*] [**-wfhost** *wfhost*] [**-wfport** *wfport*] [**-wfuser** *wfuser*] [**-wfpwd** *wfpwd*] [**-wfpath** *wfpath*]

-host *host* - sample web service host name

-port *port* - sample web service port

-wfhost *wfhost* - request workflow web service host name (runs in Java-based server)

-wfport *wfport* - request workflow web service port

-wfuser *wfuser* - request workflow web service user name

-wfpwd *wfpwd* - request workflow web service user pwd

-wfpath *wfpath* - path to workflow to be used for ticket processing in Identity

Example: runServer.bat -host myIdsJHost -port 40999

The sample web service client command line is:

ticketClient.bat [-logDir *logdir*] [**-genReqId**] [**-user** *user*] [**-pwd** *pwd*] [**-endpoint** *endpoint*] [*spmlRequestFile* [**-sleep** *sec*] [**-status** *reqnum*] [**-cancel** *reqnum*] ...]

-logDir *dir* - directory to log requests and responses (no logging by default)

-genReqId - generate dynamic request ids (false by default)

-user *user* - user name for sample web service

-pwd *pwd* - password for sample web service

-endpoint *endpoint* - address of sample web service

spmlRequestFile - file containing the ticket request to be processed

-sleep *sec* - number of seconds to wait before next operation

-status *reqnum* -

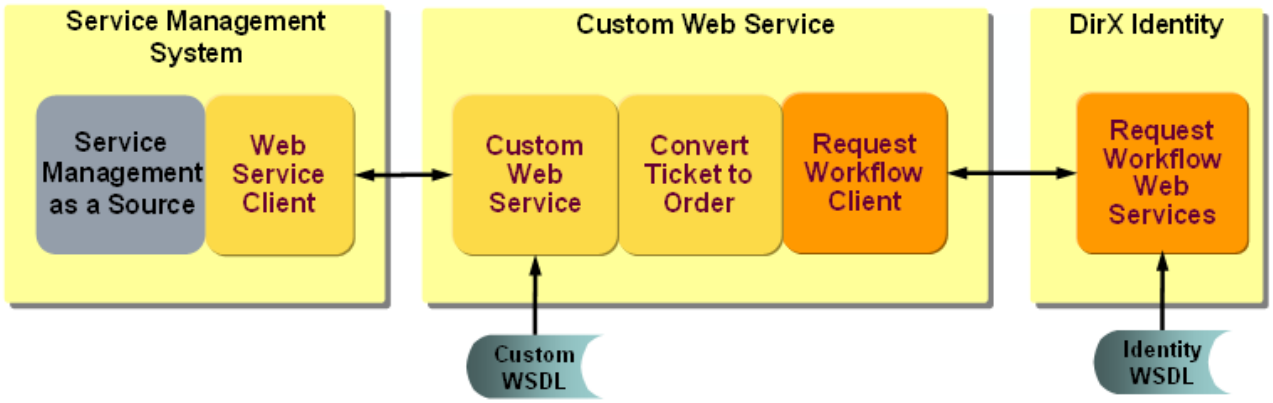
-cancel *regnum* - index of data request on command line that should be canceled

regnum - index of data request on command line the status or cancel request is requested for (positive number means that the index counts from the beginning, negative number means that the index counts from the end of the command line).

Example: ticketTest.bat createUser.xml -user cn=User,cn=My-Company -sleep 3 createUser.xml -sleep 10 -status 1 addRole.xml

2.2.1.2. Implementing Your Own Interface

First you have to decide about the interface technology. As stated above we recommend using a web service based interface because this is neutral regarding programming languages and you can reuse many parts of the sample interface.



On the DVD we provide the source code for our sample web service and client in the folder:

`Additions → ServiceMM → SampleSourceTicketApplication`

The corresponding JavaDoc you can find on the DVD in the folder:

`Documentation → DirXIdentity → ServiceMM → SampleSourceTicketApplication`

As described above, you can play with the ready-to-use sample application.

Design your own interface and use parts of the delivered sample application to implement it:

- Design your Custom WSDL.
- Think about deployment, for example whether you want to use a base technology like Tomcat.
- Create the **Custom Web Service** component (use the existing source code from the web application).
- Change the **Ticket Converting** component so that it converts your ticket format to the order format of DirX Identity and vice versa. This means that you have to create a class that implements the interface **TicketTransformer**. Pass the converted ticket orders to the TicketProcessor instance and pickup responses. Convert responses with the TicketTransformer back to original format and return them to the client. The method **SPMLRequestPortTypeImpl.processSpmlRequest()** shows the core of such processing.
- Create a **Web Service Client** based on your Custom WSDL and integrate it into your existing Service Management solution.
- Test the whole solution thoroughly.

2.2.2. Configuring Process Ticket Workflows

Set up the necessary request workflow templates for manual provisioning:

- In the **Provisioning → Request Workflows** view create a workflow folder for all request workflows that you need for ticket processing (for example **Ticket Processing**).
- Copy the workflow template **Service Management → Process Ticket** from the **Default** tree.
- Rename the copied workflow to reflect its use. Name it according to the source ticketing system from where it is used.
- Adapt the workflow configuration as required.

2.2.3. Ticket Configuration

This chapter discusses various aspects of the ticket content. The ticket must comprise all information that is important for DirX Identity to process the ticket correctly as there are:

- A **ticket identifier**. This identifier is important if you want to abort the ticket later on or if

you need to request the ticket status. In our sample tickets this is the **requestID** attribute.

- Alternatively you could use the correlation ID that is passed to the calling application in the ticket response. In our sample tickets this is the **correlationID** attribute of a ticket response.
- Of course the ticket requires an **identifier for the object** that shall be processed. Typically you will use a DN to define additionally the path in the Identity Store. If it is not possible to provide a path within the ticket, you need to perform some calculation in the web service.
- The ticket defines also all **attributes of the object** as well as links to other objects.
- It is good practice to provide the **ticket request workflow** that shall be used to process the ticket according to its type. Ticket types and the setup ticket request workflows must fit together.

2.3. Running the Use Case

Create samples of all ticket types in your source ticket system that you want to process. Test all intended use cases thoroughly. Typical use cases are:

- Create an object (user, role, permission, group). Optionally a user creation request can contain privilege assignments.
- Modify an object (user, role, permission, group). Optionally a user modification request can modify privilege assignments.
- Delete an object (user, role, permission, group).
- Status requests for all types of requests.

2.4. Alternative or Extended Configurations

The default workflows coming with DirX Identity show only specific use cases. Here are some examples of possible modifications.

2.4.1. Building Separate Request Workflows

The default sample ticket processing workflow handles additions, modifications and deletions. If you want separate processing for different types of tickets, create multiple request workflow definitions and then calculate the request workflow DN in your web service.

The workflow that will be used to process tickets for the provided samples is evaluated in the following steps:

1. Check whether the operational attribute ***workflowPath*** of the SPML request is present. For an example see the **createUserWithRoles.xml** sample ticket. This enables to specify a workflow for each individual request at the client side.
2. Check whether the command line parameter **–wfpath** *path* of the SPML web service server is set (runServer.bat).

3. If nothing is configured, the Request Workflow Client module uses per default the **Definitions/My-Company/Service Management/Process Ticket** path.

You can also create request workflow definitions for example for different object types or according to other criteria and then calculate the correct request workflow definition to start in your web service corresponding to the created ticket.

2.4.2. Using a File-based Interface

Instead of using a web service, you can use a file-based interface. You source ticket system creates tickets in some format in files and puts it to a request transfer file folder.

You have to implement a service that regularly checks for incoming files (tickets), processes it and passes it over to DirX Identity where it starts the corresponding request workflow definition. If the start was ok, delete or move the original ticket file.

After the ticket is processed, write a response to a response transfer file folder where the source ticket system can look up these results from time to time or on request.

As you can see there are many ways to build interfaces between a source ticket system and DirX Identity. At the end there is always a conversion of the ticket into DirX Identity's order format and the start of the ticket processing workflow.

3. Internal Service Management

This use case explains how to use the internal service management component of DirX Identity.

3.1. Detailed Definition

In most cases you want changes in your identity solution to be effective immediately in real-time but there are situations where it makes sense to delay changes to some time in the future. If you have no means to delay these changes you have to note the changes somewhere, wait until the time comes and then you can perform the changes at the right time.

Of course this procedure is not convenient at all. Use DirX Identity's internal service management component to manage such situations.

Typical situations are:

- A person moves from one department to another one. This change includes a change of his privileges. In this case his data, for example his department number as well as some privileges will change at the expected time of the move because the tasks in the new department might be different.
- Someone changes his location. This results in a change of his address that means eventually the country, the location, the building and the room number. Additionally, his communication data like the telephone number could change.

In both cases it is possible to define the necessary changes some time before the move or change. DirX Identity services process all changes fully automatically at the defined due date.

3.1.1. Evaluation of this Use Case

This section elaborates on advantages and disadvantages of this solution.

Advantages

- Fully and smoothly integrated solution. No external components required.
- Low effort to setup.
- Complete audit integration.

Disadvantages

- Restricted customization capabilities compared to external service management solutions.

3.1.2. Additional Documentation

Besides the information in this document you can find additional documentation in the

DirX Identity documentation:

- Read the follow-on tutorial "Working with Internal Tickets" in the *DirX Identity Tutorial*. Perform the example use case in the sample domain to understand the feature.
- Read the description of the relevant workflow in the *DirX Identity Application Development Guide*. Read the section "Process Tickets Internal Workflow" in the chapter "Understanding the Maintenance Workflows".
- You can find additional information in the chapter "Managing Tickets" in the *DirX Identity Provisioning Administration Guide*.

3.2. Set Up and Configuration

You have to set up

- The Process Internal Ticket workflow that will process your tickets.
- A daily schedule that runs this workflow regularly.

3.2.1. Setup the Process Internal Ticket Workflow

Copy the prepared template **ProcessTicketsInternal** Workflow into your solution.

- In the **Connectivity** view group click the **Global View**.
- We assume that you have already established a corresponding connectivity scenario for your provisioning domain.
- Select your scenario and click the workflow line between the two Identity Stores.
- Select **New** from the line's context menu.
- Choose **ProcessTicketsInternal** from the list and click **Next**.
- Adapt the **Name** and **Description** fields and set the **Active** flag.
- Click Finish.

Your instance of the ticket processing workflow for internal tickets is ready to run.

3.2.2. Setup a Schedule

To setup a schedule for the previously copied workflow perform these steps:

- Select the **Expert View**.
- Create under the **Schedules** node in the respective scenario folder for your provisioning domain a new schedule.
- Set the **Workflow** link to the ticket processing workflow and set a **Start Time**. It makes sense to run the workflow during the night to reduce load during daytime. Note that the **Time Interval** is already set to 24:00:00 which is correct for a daily running workflow.
- Do not set the **Active** flag at this time because we want to test our solution first.

3.3. Running the Use Case

To run the use case you need to create and change objects with a **Due Date**. Here are some useful examples:

Changing a user's attribute:

- Change a user's room number and set the Due Date to today.
- View the newly created ticket with the DirX Identity Manager.
- Run the ticket processor workflow to process the ticket.
- Check the user and the ticket after processing.

Adding a role without approval to a user:

- Assign a role that is not flagged for approval to a user and set the Due Date to today.
- View the newly created ticket with the DirX Identity Manager.
- Run the ticket processor workflow to process the ticket.
- Check the user and the ticket after processing.

Adding a role with approval to a user:

- Assign a role that is flagged for approval to a user and set the Due Date to today.
- View the newly created ticket with the DirX Identity Manager.
- Check that a corresponding approval workflow was started that is referenced from the ticket.
- Approve the assignment request.
- View the user: nothing happened up to now.
- Run the ticket processor workflow to process the ticket.
- Check the user and the ticket after processing.

If you want to run other ticket examples, perform tests with these objects and attributes.

After doing these tests, activate the previously established schedule and your solution is ready to run.

3.4. Alternative or Extended Configurations

For this use case there are currently no alternative or extended configurations available.

4. Manual Provisioning of Offline Systems

This chapter describes the use case where various connected systems in the customer environment are to be connected via manual provisioning.

4.1. Detailed Definition

This use case assumes that there are connected systems that are not yet automatically managed from DirX Identity. Typical reasons are:

- The number of users (accounts) in the connected system is low or the change frequency in that system is too low. The effort to connect this system with automatic provisioning is too high.
- The customer wants to integrate the system as fast as possible into the identity management solution. This establishes a common view on all systems and adds noticeable value in the compliance area (all actions are tracked via auditing). In a later stage of the project the system could be integrated more tightly with automated provisioning.

This use case assumes that the connected system is modeled as target system of type RequestWorkflow in DirX Identity. If the system already exists, an initial load process is necessary to fill the target system with accounts, groups and memberships.

The system is treated as any other target system in DirX Identity, for example the accounts have to be associated with users, the groups have to be integrated into the privilege model or can of course be used for direct group assignment.

From now on this target system can be used as any other one. All actions that occur as there are for example account or group creations or modifications or management of group memberships create approval workflows where the administrators of the connected system are the approvers (let's better say workers).

The administrators work on their tasklist, see the task requests (for example an account add request with all attributes) and perform each required task on the connected system side. After they have done it successfully (for example they added the account), they approve the task and this sets the relevant states in the corresponding object in the target system. All actions are audited given that the audit policies are set correctly.

4.1.1. Evaluation of this Use Case

This section elaborates on advantages and disadvantages of this solution.

Advantages

- Fast integration into the identity management solution.
- Low effort for integration.
- Only slight changes to administrative tasks (going away from paper to request workflow tasks).

- Ability to audit all actions regarding this target system.

Disadvantages

- No automatic checks possible whether the actions were correctly performed by the relevant administrator.
- No way to validate additional manual actions in the connected system.

4.1.2. Additional Documentation

Besides the information in this document you can find additional documentation in the DirX Identity documentation:

- Read the follow-on tutorial "Using Manual Provisioning" in the *DirX Identity Tutorial*. Perform the example use case in the sample domain to understand the feature.
- Read the description of the relevant workflows in the *DirX Identity Application Development Guide*. Read about the "Service Management Workflows" in the chapter "Understanding the Target System Workflows" and the "Manual Provisioning Workflow" in the chapter "Using Request Workflows".

4.2. Set Up and Configuration

You have to set up

- The target system that represents the offline system.
- Set up the initial state of the target system to reflect the state of the offline system (manual initial load).
- Configure a corresponding Manual Provisioning workflow in the Provisioning view
- Configure the **Service Management** workflows in the Connectivity view

All parts must fit together to work correctly.

4.2.1. Setting up the Target System

For each offline target system create a corresponding target system:

- In the **Provisioning** view group open the target system wizard.
- Select **Service Management (type = RequestWorkflow)** as template.
- Enter a name and a description for the offline system.
- Set a cluster and domain name that is used by the corresponding Java-based workflows for reference. If you have a set of offline systems use as cluster the set name (for example MyOfflineSystems) and as domain the name of the offline system.
- Set the host and port address of the Java-based server where the request workflow engine runs.
- Click through the rest of the pages and then click **Finish**.

A new target system is created.

4.2.2. Initial Load of the Target System

If your offline system already exists (which is probable), you need to load the existing information (accounts, groups and memberships) into the newly created DirX Identity target system. Use one of these methods:

- Enter the information manually. Create all accounts and groups and then add the accounts as members to the relevant groups. This is of course only possible for a small amount of data.
- Export the data from your offline system into any of the formats DirX Identity can read (LDIF, CSV, XML). Set up and configure a workflow that imports this information into the target system.
  Note: we recommend building a validation workflow to initially load this data. The advantage is that you can run the workflow again from time to time to validate whether the administrators work correctly.

4.2.3. Configuring the Workflows in the Provisioning View

Set up the necessary request workflow templates for manual provisioning:

- In the **Provisioning → Request Workflows** view create a workflow folder for all request workflows that you need for manual provisioning (for example **Provisioning**).
- Copy the workflow template **Service Management → Manual Provisioning** from the **Default** tree.
- Rename the copied workflow to reflect its use. Name it according to the target system where it is used.
- Adapt the workflow configuration as required.

4.2.4. Configuring the Workflows in the Connectivity View

During target system creation, the corresponding workflow was already created. It is located in the folder named **Service Management**. Perform these steps to configure the workflow:

- Rename the workflow folder and the workflow according to the created target system. Best is to use the name of the offline system.
- Open the workflow wizard and
- General info: Set the active flag and set a timeout for the workflow.
- Join Activity General Info:
  - Either keep the resource family **Request_Workflow** or define your private one.
  - Set the error handling timing parameters.
- Join Activity Controller Parameters:
  - Set the **Write Audit Log** flag to enable status entry generation.

- Request Workflow Settings:
  - Set the **Primary** and **Secondary Workflow DN**s.
  - Adapt the **Account Mapping to CS**, **Group Mapping to CS** and **Membership Mapping to CS** steps. These are the attributes that are presented to the administrators in Web Center.
- Adapt and extend the mapping as required.
- Set up the resource family **Request_Workflow** (or the one you defined above) at your Java-based Server configuration object.
- Restart the Java-based Server.

4.3. Running the Use Case

To run the use case you need to create and change objects (groups and accounts) in your target system:

- Create a group. The creation request starts a Java-based workflow that itself starts a corresponding request workflow.
- Check the corresponding workflow entry in the **Connectivity → Monitor** view. It contains information about the objects to create or to change.
- Check the relevant request workflow entry in the **Provisioning → Request Workflows → Monitor** view.
- Assign the group manually to a user that does not yet have an account. View the new workflow entry in the **Connectivity → Monitor** view. Check also the corresponding request workflow entry in the **Provisioning → Request Workflows → Monitor** view.

4.4. Alternative or Extended Configurations

The default workflows coming with DirX Identity show only one possible use case. Here are some examples of possible modifications.

4.4.1. Building Separate Request Workflows

The default sample workflow handles additions, modifications and deletions. If you have specialized administrators, you could build three different workflows that each handles only additions or modifications or deletions. Set in this case the **When applicable** section only for one operation.

Of course you can use other criteria to separate the workflows. For example you could distribute the work to the administrators based on the accounts organizational unit or locality attribute (use the **Condition** section in the **When applicable** area). Prerequisite is that you inherit the attribute from the user to the account.

4.4.2. Using Different Workflows for Accounts and Groups

You can use different workflows for accounts and groups. Set the relevant parameters in

the real-time workflow wizard's **Request Workflow Settings** step.

- Set the Primary Workflow DN field to the workflow that shall be used for accounts. Note that this workflow also handles memberships because these are configured at the account side.
- Set the Secondary Workflow DN field to the workflow that shall be used for groups.

DirX Product Suite

The DirX product suite provides the basis for fully integrated identity and access management; it includes the following products, which can be ordered separately.

DirX Identity

DirX Identity provides a comprehensive, process-driven, customizable, cloud-enabled, scalable, and highly available identity management solution for businesses and organizations. It provides overarching, risk-based identity and access governance functionality seamlessly integrated with automated provisioning. Functionality includes lifecycle management for users and roles, cross-platform and rule-based real-time provisioning, web-based self-service functions for users, delegated administration, request workflows, access certification, password management, metadirectory as well as auditing and reporting functionality.

DirX Directory

DirX Directory provides a standards-compliant, high-performance, highly available, highly reliable, highly scalable, and secure LDAP and X.500 Directory Server and LDAP Proxy with very high linear scalability. DirX Directory can serve as an identity store for employees, customers, partners, subscribers, and other IoT entities. It can also serve as a provisioning, access management and metadirectory repository, to provide a single point of access to the information within disparate and heterogeneous directories available in an enterprise network or cloud environment for user management and provisioning.

DirX Access

DirX Access is a comprehensive, cloud-ready, scalable, and highly available access management solution providing policy- and risk-based authentication, authorization based on XACML and federation for Web applications and services. DirX Access delivers single sign-on, versatile authentication including FIDO, identity federation based on SAML, OAuth and OpenID Connect, just-in-time provisioning, entitlement management and policy enforcement for applications and services in the cloud or on-premises.

DirX Audit

DirX Audit provides auditors, security compliance officers and audit administrators with analytical insight and transparency for identity and access. Based on historical identity data and recorded events from the identity and access management processes, DirX Audit allows answering the “what, when, where, who and why” questions of user access and entitlements. DirX Audit features historical views and reports on identity data, a graphical dashboard with drill-down into individual events, an analysis view for filtering, evaluating, correlating, and reviewing of identity-related events and job management for report generation.

For more information: support.dirx.solutions/about

EVIDEN

Eviden is a registered trademark © Copyright 2026, Eviden SAS – All rights reserved.

Legal remarks

On the account of certain regional limitations of sales rights and service availability, we cannot guarantee that all products included in this document are available through the Eviden sales organization worldwide. Availability and packaging may vary by country and is subject to change without prior notice. Some/All of the features and products described herein may not be available locally. The information in this document contains general technical descriptions of specifications and options as well as standard and optional features which do not always have to be present in individual cases. Eviden reserves the right to modify the design, packaging, specifications and options described herein without prior notice. Please contact your local Eviden sales representative for the most current information. Note: Any technical data contained in this document may vary within defined tolerances. Original images always lose a certain amount of detail when reproduced.