

Virtual system

system, virtual

What is a virtual system?

A virtual system is not directly connected for online management. The virtual system is basically only a registration mode, where for each system change is generated the implementation request (notification) that is assigned to the particular administrator. This administrator must provide that the change is made to the target system. In other words, IdM "knows" what the user should have on the system for accounts and permissions, but on the real system this is executed by the **implementer** (administrator). The reason may be the need to manage a large number of systems without the need for demanding integration.



The quickstart for creating a new virtual system with default mapping, you can use tutorial: [Virtual system - New virtual system in 5 minutes](#).

The usage and configuration of virtual systems is described in the [Admin tutorial](#).

Notifications

After the request for updating virtual system is created, the notification is sent to all implementers.

The virtual system notifications are sent on the topic:

```
VirtualSystemModuleDescriptor.TOPIC_VS_REQUEST_CREATED=vs:vsRequestCreated
```

You can create your own template (email, sms, console), this template only have to listen on correct topic.

As default was implemented email notification 'vs:vsRequestCreated' for new virtual system requests. This notification is by default automatically connected to this topic. Template for this notification is saved in VS module as XML:

```
Realization/backend/vs/src/main/resources/eu/bcvsolutions/idm/templates/VsRequestCreated.xml
```

How does it work inside?

Virtual system is based on four entities (tables):

- **VsAccount** keeps all confirmed accounts in VS. It contains only basic information (UID, System ID, Enable). It must be equal to the state of accounts on the target system → shows "True".

- **VsAccountFormValue** contains dynamic values for VS account.
- **VsRequest** keeps all requests for VS. It contains not-yet-implemented wish on the target system.
- **VsSystemImplementer** contains implementers for VS. Implementers can be defined directly or via role.

Virtual connector

For virtual system was created new connector **virtual-system-basic** (BasicVirtualConnector.class). This connector provides an interface for external communication with the virtual system.



Virtual connector is the first implementation of the connector in the 'czechidm' connector framework.

Connector following operations of the 'czechidm' connector framework:

- **lcCanRead**,
- **lcCanCreate**,
- **lcCanUpdate**,
- **lcCanDelete**,
- **lcCanGenSchema**,
- **lcCanSearch** (Search in VS is supported only during reconciliation, it returns all accounts. No connector filter can be now used.)

Virtual connector configuration

Configuration of the connector is represented by POJO class **BasicVirtualConfiguration**. This class contains attributes annotated with '@lcConfigurationClassProperty'. The annotation adds metadata as '**displayName**', '**face**', '**order**', '**helpMessage**', '**required**'. Values from this configuration plus metadata create final configuration for the virtual system. This is displayed on the system configuration tab as you can see in the chapter [Virtual systems configuration](#).



Virtual connector returns accounts from inner table VsAccount and **merges changes from all unresolved requests** for the same account. Virtual connector doesn't return implemented "True" on target system, but the state when all requests are successfully implemented!

Change of user name (UID)

User name of an identity is by default mapped as UID in VsAccount. Special situation happens, when the username (UID) is changed in IdM. When the UID attribute is changed on other connectors (Ldap, Table, ...), new UID is returned as the result from the connector Update method. But the virtual

connector works asynchronously, so no result is returned and `SysSystemEntity.UID` is not updated.

This state is resolved in virtual connector in the following way:



When the VS request containing the change of the UID **is marked as implemented**, then ACC module is called directly from the VS module and **`SysSystemEntity.UID`** is updated.

Update configuration of system (connector) and VS module



This processor is implemented since version 8.2.0.

Implementers and EAV definition for VS is not updated in initialization of VS connector anymore. For update configuration was in the version 8.2.0 created processor invoked on the create or update system in the IdM. This processor ensures update of VS systems directly. It means since that version is not necessary to invoke change of VS configuration via IC module (for example by using the "Test system" button).

For this is used '**`SystemEventType.CREATE`**, **`SystemEventType.UPDATE`**, **`SystemEventType.EAV_SAVE`**' events and processor '**`system-save-processor`**'.

Integrity between ACC and VS module (deleting the virtual system)



This integrity is implemented since the version 7.5.0.

Before the system is deleted in ACC module, it's necessary to call VS module and ensure the deletion of connected entities (on the deleted system).

For this is used '**`SystemEventType.DELETE`**' event and processor '**`system-delete-processor`**'.

Sequence of executing operations in delete system processor in VS module:

1. Check existing unresolved VS requests - If an unresolved vs request exists, then it isn't possible to delete the system (it throws an exception).
2. Delete all archived VS requests for the system.
3. Delete all VS accounts for connector key from the system.
4. Delete VS account form definition for the system.
5. Delete all VS system implementers for the system.

Known limitations



Implementers in the VsRequest are filled only in the requests invoked via REST API. Max first **10** implementers are returned in the one request (via REST API)



Max first **50** implementers are used as recipients in the request notification!

- Only ACCOUNT object type is now supported in VS. It means that only identity provisioning is fully supported.
- ACC and VS module integrity check for deleting systems is implemented since the version 7.5.0. Previous versions don't clear all connected entities when deleting a virtual system.
- Initialization of data for the VS module is not implemented now. → It's necessary to create a role for virtual system implementers, add permissions for VsRequest to this role and set the evaluator by implementers manually.

From:
<https://wiki.czechidm.com/> - IdStory Identity Manager

Permanent link:
https://wiki.czechidm.com/devel/documentation/modules_vs/dev/virtual-systems

Last update: **2018/07/09 13:17**

