Tree structures: indexing and searching

tree, indexing, synchronization, sync, provisioning

To make queries in an efficient manner, we provide an index next to the tree structure with the following advantages:

```
  • the possibility to ask about the children of any element of the tree
with one query (all the children in the downward direction)
  • likewise, the possibility to ask about all the parents of any element of
the tree with one question
```

You can search identities relying on an index based on:

```
  • organizational structure (through a contractual relationship)
  • the display of the identity position within the organizational structure
```

Rebuilding an index

There are situations when you need to re-index the organisational structure. To do so, you can employ the RebuildTreeNodeIndexTaskExecutor task, where you need to enter the code of the structure which should be re-indexed.

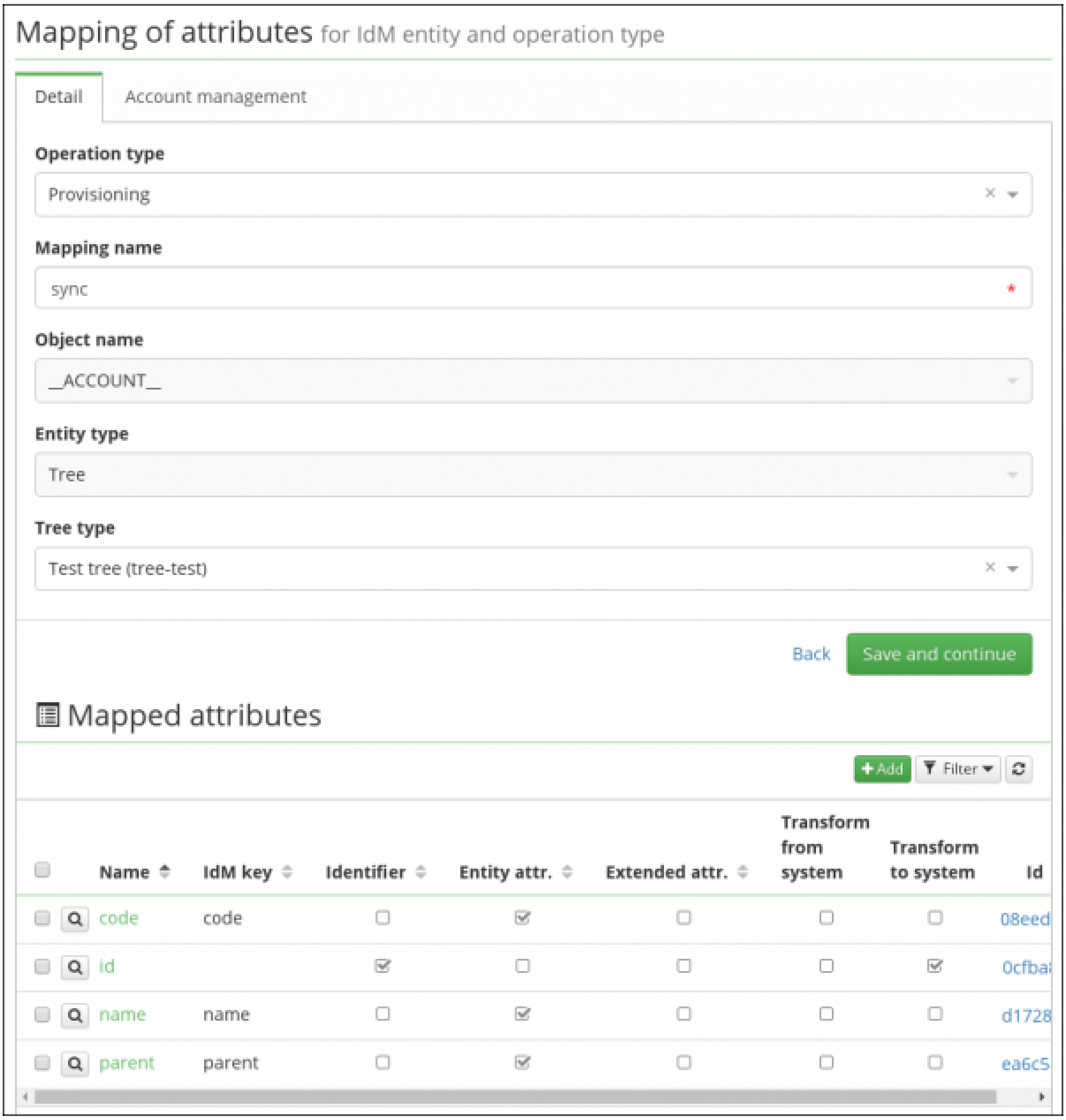
Synchronization - tree nodes

sync, tree

Just like identities, you can also synchronize the tree structure. Here are steps to follow in order to synchronize the organizational structure:

1. First you need to create a type of structure. Say, **ORGANIZATIONS**.
2. On the system, create a mapping for an entity type **Tree**. Select the type of structure you want (**ORGANIZATIONS**).
3. Create the **Required** mapping attributes:
    - **identifier** - The attribute contains an unambiguous element identifier from the end system. Used for parenting. It must be labeled **identifier**.
    - **code** - The code is a unique identifier in the tree.
    - **name** - Name of the element. This name will be displayed in the tree component.
    - **parent** - An attribute pointing to your parent. From mapping point of view, this attribute must return **identifier** parents, Identifier from the end system. From the IDM view, parent contains a reference to the parent object, ie IdmTreeNode. Searching and conversion between system identifier and parent IDM (IdmTreeNode) do synchronization automatically.
4. Create **optional** mapping attributes. A typical example is some **extended attribute**.
5. We create sync to select the mapping you created. The tree structure synchronization configuration is similar to the synchronization of identities. In addition, it is possible to define (in the form of a Groovy script) how to "know" **the roots of the tree**.
6. **Start synchronization**.

Tree sync is always run as **reconciliation** at this time. It means setting your own filter or token will have no effect.



Mapping of attributes for IdM entity and operation type

Operation type

Provisioning

Mapping name

sync *

Object name

__ACCOUNT__

Entity type

Tree

Tree type

Test tree (tree-test)

Mapped attributes

□		Name	IdM key	Identifier	Entity attr.	Extended attr.	Transform from system	Transform to system	Id
□	🔍	code	code	□	☑	□	□	□	08eed
□	🔍	id		☑	□	□	□	☑	0cfba
□	🔍	name	name	□	☑	□	□	□	d1728
□	🔍	parent	parent	□	☑	□	□	□	ea6c5

How to synchronize all nodes under a node already existing

Sometimes you need to synchronize all nodes from the source system by embedding them underneath a node which already exists in IDM. The good news is you can.

(TO BE COMPLETED)

Provisioning of tree nodes

provisioning

Tree provisioning behaves differently than the provisioning of identities. The main difference is the absence of a separate account management mechanism.

In the case of trees, account management is directly linked to the creation / modification / deletion event of the tree node.

(TO BE COMPLETED)

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

Permanent link:
https://wiki.czechidm.com/devel/documentation/organization/adm/tree_structures

Last update: **2019/02/14 14:47**