CMDB

Configuration and Use of the CMDB of *Xpert.NET*

Table of Contents

1 Introduction

1.1 Purpose of the Document

Xpert.NET is a very dynamic help desk solution that can be configured in various directions. Daily routine and experience have shown that users get acquainted with the most important modules and configurations quickly. Nevertheless, some questions regarding the functionalities of the *Xpert.NET* modules often remain. Thus, the individual modules cannot be used to their full potential.

This document was devised in order to help you with the orientation in and the configuration of the Configuration Management Database. For this purpose, it will describe the configuration as well as its integration into the entire *Xpert.NET* system step by step.

1.2 Addressees of the Document

This document mainly addresses administrators of *Xpert.NET*.

Nevertheless, end users may also find helpful advice on handling the module here. As an administrator, you can compile a slim document for your end users by leaving out passages only necessary for the administration of the module. However, there are smaller, slimmer documents available for users and members of support.

1.3 Remarks on the Content of this Document

This document describes all the functions of the CMDB module and its extensions. available for the end user in *Xpert.NET*. Nevertheless, the range of functions may vary due to configuration, licensing, and version. If you miss certain functions listed in this document in your *Xpert.NET* installation, please contact our support directly.

1.4 Overview of the CMDB

With this module, configuration items (CIs) and their relations between each other can be managed easily and structured according to their status, category, and zone in the Configuration Management Database (CMDB). Moreover, the CMDB can be used as a resource for an automated asset management like MS SMS etc., so that *Xpert.NET* can be used for the inventarization of hard- and software. Thus, any data field of a CI can be displayed on the CMDB tab of any ticket. A member of the service desk will have a clear overview of all the configured data of a system in the ticket view at any time. Via the available interfaces, any databases can be connected to *Xpert.NET* and the CMDB as well.

The Configuration Management Database of *Xpert.NET* is based on the principle of the IT Infrastructure Library (ITIL v3), whereupon a Configuration Item can represent any item within an organization. This item has to be contained in another item or a zone (managed geographically or organizationally) and can contain further items or be linked to them in another manner. Every item has a lifecycle defining its possible statuses.

The available types of CIs are defined in schemas. These schemas are ordered hierarchically and hand down data fields to all their sub-schemas. In a schema, the fields themselves as well as the default values for fields and the CIs' details are defined. Rights within the CMDB can be assigned to schemas and zones.

2 Basic Configuration

For an operational Configuration Management Database (CMDB), the following configuration settings are necessary:

- Various life cycles containing information on reachable statuses and status transitions of CIs;

- CI schemas containing settings of CI actions, fields, and SLA information;

- CI types with visibility settings of the schemas' fields;

- zone schemas with zone-specific dynamic fields and master fields;

- a zone hierarchy depicting the organizational or geographic structure of the enterprise.

2.1 Life Cycles

The lifecycle of a configuration item (CI) is defined by the statuses to be passed through and reached as well as their dependencies. Thus, structured status dependencies have to be planned for every life cycle beforehand. It is necessary to define start and end statuses, to make these statuses reachable, and to make sure that every status has further reachable statuses (except the end status).

A life cycle defines such a set of statuses and their transitions and thus, when an affected CI is to have which status. The life cycles can be managed via SETTINGS -> CONFIGURATION MANAGEMENT (CMDB) -> LIFE CYCLES.

Here, it is possible to define that a CI can, for example, only be moved from the status *New* to the status *Active* via the status *Test*. It is necessary to consider how many types of CIs should and must use a life cycle of their own.

2.1.1 Configuring a Life Cycle

Clicking the *New life-cycle* button will open a new window, in which a **Name** and a **Description** for the new life cycle is to be entered. The localization buttons will be displayed, as soon as multiple languages have been activated on the *Xpert.NET* system.
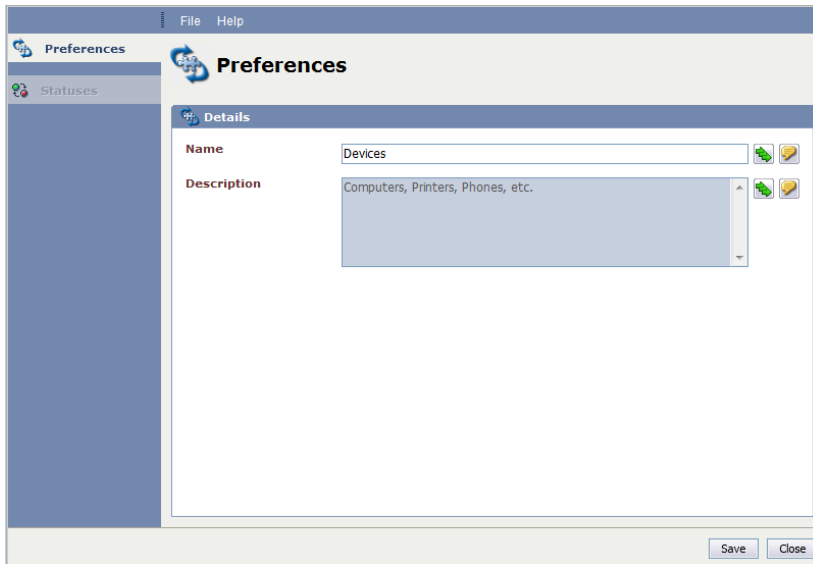
Figure 2.1:

Only after a click on the *Save* button, the life cycle will be created and the *Status* tab will be available. Via the button *New status*, as many statuses as necessary can be created here.
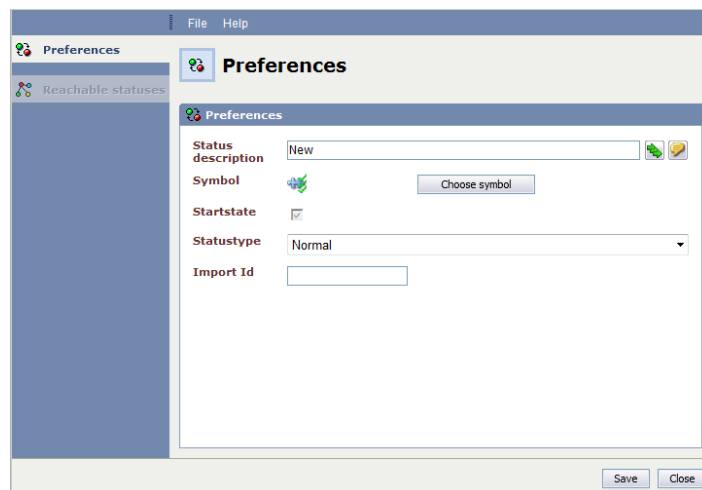
Figure 2.2:

The following configuration has to be made for every status:

Status description: The name of a status as it will be displayed in the configuration item later on. The localization buttons will be displayed as soon as multiple languages have been activated on the *Xpert.NET* system.

Symbol: The icon set here illustrates the respective status and will be displayed for a better visualization of the status in the detail view, for example. A predefined set of icons is available. However, any other icons can be used as well. They have to be .gif files at a size of 20 x 20 px.

Start state: Every CI obtains the start status in a life cycle first. There can only be one start status in a life cycle.

Status type: The status type is a rough categorization of the freely definable statuses. The list of status types cannot be edited.

Import ID: This field can be left empty. Import IDs are used for a distinct identification. If imports are to be executed, further informatiocan be found in Chapter 5.

After saving the status preferences, the *Reachable statuses* tab will be available. It can be ignored until all of the required statuses have been created.

2.1.2 Configuring the Status Dependencies

After all of the designated statuses have been created, the transitions between the individual statuses have to be defined. It is possible to define, which statuses can be reached from which statuses. For this purpose, the initial status has to be opened via a double-click and the tab *Reachable statuses* is to be selected. On this tab, all statuses to be reachable from this status can be added by clicking the button *New reachable status*. Please note that workflows may be triggered by status transitions (only workflows of the *CI* type are available here).

A reachable status can be removed by selecting the respective option in the *Actions* menu.

Example for Software

Status: *New*

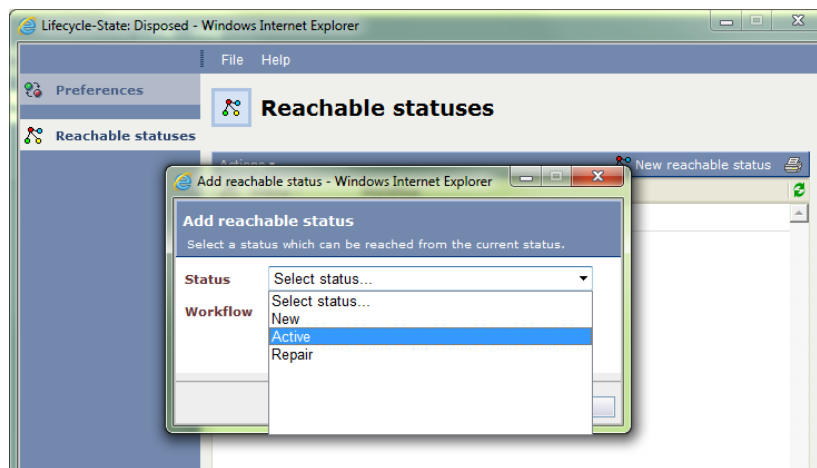Reachable statuses: *Active*, *Repair*

Figure 2.3: Reachable statuses

After all of the statuses have been set up, the schemas have to be configured.

2.2 CMDB Schemata

In the CMDB schema configuration stage several schemas can be consolidated. On one hand, the CI schemata and the corresponding types are configured, on the other hand, zone schemas and the default hierarchy of the zones is managed.

2.2.1 The Difference Between CI-Schemas and CI-Types

Just like in a ticket schema, all settings, fields, actions, and Service Level Agreements for CIs to be based on this schema later on can be defined and and saved in a CI schema.

The distinction between schemas and types has been made mostly due to clarity. While all configurations of the CIs are made in the CI schema, the configuration of the types determines, which tabs and fields will be displayed in the CIs later on.

> **Please note**
>
> A CI schema contains all definitions necessary for creating CIs. Thus, the attributes (fields) and field types have to be planned thoroughly - subsequently modified field types may lead to problems with the field value compatibility. For identifying a CI, it is advisable to create a logical pattern!

2.2.2 Schema Hierarchy

The hierarchy of the CMDB schemas can be created as a flat or a deep hierarchy. In contrast to zones and their hierarchy, a subdivision into various types and subtypes of items is established via the schema hierarchy. The deeper and more precise the hierarchy, the less defining is required for the individual CIs later on.

Example for a Flat Hierarchy

- Devices
    - Computer

Example for a Deep Hierarchy

- IT devices
    - Computer
        * DELL
            · Precision 380

> **Hint**
>
> Drag and drop can be used for the schema configuration. When an element is moved, the inheritance will be adjusted automatically.

2.2.3 Configuring a CI Type

CI types are used for creating hierarchies. Thus, *Network printers* may be found under *Printers*, which itself is subordinate to the type *Devices* , for example. Not every one of those types has to be assigned with a CI schema.
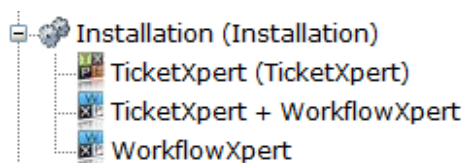
Figure 2.4: A CI schema with sub-types, which do not have their own CI schema

For example, only the main type could be linked to a CI schema. The sub-types could then be configured differently than the main type but still use the same CI schema. Whenever a schema is created for one of the sub-types, it will be created as a sub-schema of the schema linked to the main type automatically. As long as a type is not linked to a schema, it can be moved within the tree hierarchy via drag and drop arbitrarily. The moved type will then use the respective schema of the new main type. As soon as a type and schema have been linked, the type cannot be moved anymore.

When a new type is created via the *Tasks* menu, a dialog for configuring the new type appears.

Name: The type's name. The localization buttons will be displayed as soon as multiple languages have been activated on the *Xpert.NET* system.

Import ID: This information can be left empty. Import IDs are used for a distinct identification. If imports are to be executed, further information can be found in Chapter 5.

Symbol: The symbol that illustrates the type. A predefined icon set is available, but other icons can be used as well. They have to be .gif files at a size of 20 x 20 px.

After using the *Save* button, the *Properties* tab will be available. The visible tabs for this type can be configured here. In every detail dialog of a configuration item using this type, only the tabs defined here will be displayed. If, for example, the tab *Attachments* is not needed, it can be simply faded out here.

When changing the visibility settings for tabs and fields in a hierarchically superioir type, the visibility settings can be applied to subordinate types as well.

The *Fields* tab cannot be used before a CI schema has been linked to the type. The dynamic fields of the schema can be faded in and out via checkboxes here.

2.2.4 Configuring a Schema

For the creation of a CI schema, it is necessary to have at least one existing type that has not been connected to a schema yet. If no free type is available, a new one has to be created (see Section 2.2.3). Only then a CI schema can be created for this type.
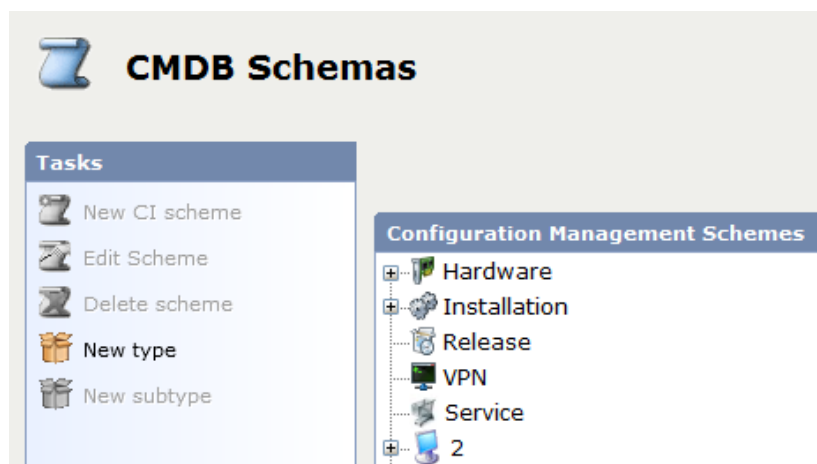
If types are available, CIs can be created for them. For this purpose, the type or the sub-type the schema is to be created for has to be selected. A click on *New CI schema* in the *Tasks* section on the left will open a dialog, in which all of the required details for the schema can be specified. After saving, the remaining tabs will become available for the respective schema:
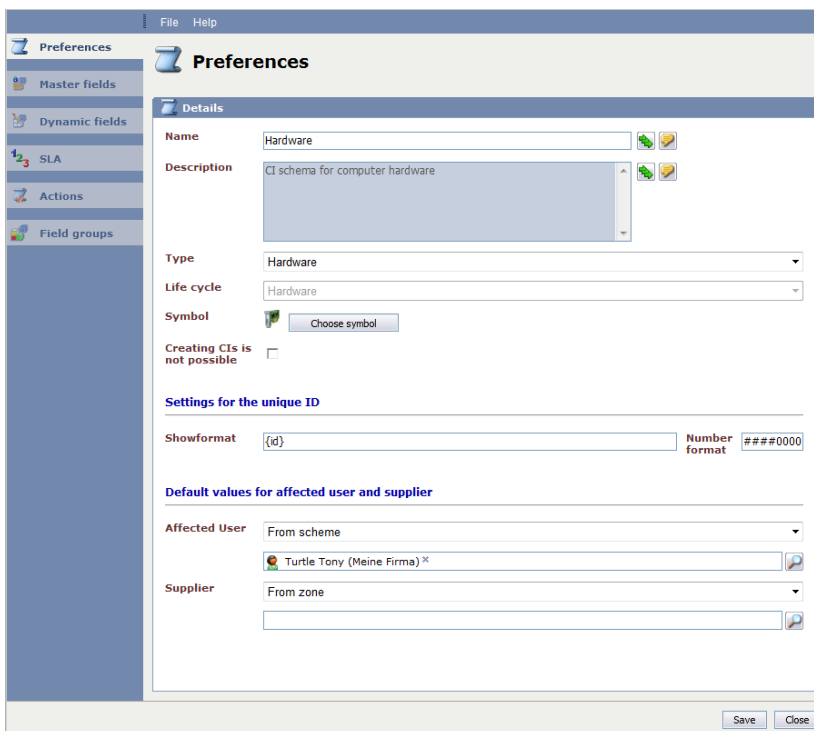
Figure 2.6:

Preferences

On the *Preferences* tab, the only tab that has to be filled in before saving, the basic configuration of the CI schema has to be selected:

Name: The name of the schema. The localization buttons will be displayed as soon as multiple languages have been activated on the *Xpert.NET* system.

Description: The text stored here is purely descriptive and describes the schema. The localization buttons here will be displayed as soon as multiple languages have been activated on the *Xpert.NET* system as well.

Type: Select the respective type of the schema here. If there is no sub-type available for the selected type for the schema, this entry can be ignored. If there are sub-types available, they can be selected here as an alternative. The CI schema will then be linked to this type.

Life cycles: Defines a corresponding previously created life cycle.

> **Warning**
>
> If CIs have been already created with this schema, the life cycle cannot be changed any more.

Symbol: An icon illustrating the CI schema. Alternatively to the already existing schema icons, further icons can be uploaded.

Creating CIs is not possible: This option prevents CIs being created with the schema. This option can be used if this schema is only to be used for importing CIs. Otherwise, this setting can be ignored.

Settings for the unique ID: This option configures the display format and the number format for every CI schema individually (cf. numerical ranges for tickets). The default settings can be used or customized.

Note on Unique ID and Visible ID

Expenses use a serial ID solely composed of numeric characters - without a prefix or suffix. Thus, alpha-numeric values should be used during the definition and assignment of IDs to CIs. If only numeric characters are used as the serial ID, it may happen that the CI instead of a possibly existing expense with the same ID will be preferred when opening it via the direct access. Thus, it may become impossible to open the expense via the direct access.

Default values for affected user and supplier: Here, defaults can be set for affected users and suppliers. (Which affected user, which supplier, or which owner is to be set in new CIs if no other selection has been made?)

After all specifications have been made, clicking on *Save* will save the settings for this schema. Now, all the other tabs on the left side will be available and can be configured:

Master Fields

This tab allows for creating master fields and for displaying inherited master fields. Master fields are used for entering information that cannot be modified during CI creation later on and is valid for all CIs of this schema, e.g. telephone numbers of the service provider for CIs of this schema.

The button *New field* allows for creating master fields. The **Name** of the master field has to be defined here as well as the **Value** for the field (e.g. a hotline number) are to be defined. If the option **Inherit** is activated, this master field is inherited by all schemas that are below this schema in the hierarchy.

If master fields have already been created, they can be deleted via the *Actions* menu.

Inherited master fields: On this tab, all the master fields inherited from superordinate schemas will be displayed. They cannot be changed on this tab. In order to change them, the respective superordinate schema, in which the master field has been created, has to be selected.

Dynamic Fields

On this tab, dynamic fields can be created and inherited dynamic fields can be displayed. Dynamic fields are used for data that is different for every CI. Dynamic fields function

similarly to the ticket fields, i.e. all the fields to be available for dynamic data to the future CIs have to be defined here.

The button *New field* is used for creating the required dynamic fields. The following information for the respective dynamic field is to be entered here:

- **Name**: Specifies the name of the dynamic field as displayed left of the dynamic field in the CI.

- **Info text**: This text is displayed as a mouseover text for the question mark symbol, which is located right next to the dynamic field in the CI.

- **Type**: The following predefined field types can be selected here:

    - **Text**: Any text can be entered into this field.

    - **Number**: Only numbers can be entered into this field.

    - **Date**: Only date values can be entered via a date input field here.

    - **True/False**: The dynamic field is turned into a checkbox that can be set active or inactive. If an entry for this field type is necessary, this checkbox always has to be activated.

    - **Selection field**: This type is used for creating a preconfigured list to select values from during zone creation.

    - **Service time**: This field allows for selecting a service time for the Service Level Management.

    - **Link**: Via this type, a link can be entered into the field. If an arbitrary text is entered into the dynamic field, it will be be automatically transformed into a link after saving the settings.

    - **Text Area**: Instead of a text field, a larger text area is made available, which is able to contain larger amounts of text.

    - **User Browser**: Via the user browser, *Xpert.NET* users can be entered into the dynamic field.

    - **Localized text**: This type allows for entering text in multiple languages into the field.

    - **Localized textarea**: Via this field type, larger amounts of text in multiple languages can be entered into the field.

    - **Table**: This field type is used to create a table with multiple columns (the content of a row in the configuration will be displayed in a column later on). In the zone, multiple rows can be entered for every column.

    - **IP Address**: This field type allows for entering an IP address. It is necessary for the action *External application*, for example .

- **Format**: If a certain format is to be specified for the selected type, it can be entered here. Examples for possible formats can be displayed via the small help icon on the right.

- **Default value**: If a preset value is to be displayed in this field later on, it can be entered here.

- **Input is required**: If this field is activated, the dynamic field turns into a required field and has to be filled in before the CI creation can be completed.

Inherited dynamic fields: Dynamic fields will be passed onto CI schemas subordinate in the hierarchy automatically and will be displayed in those. However, they cannot be edited further.

Dynamic fields can be arranged in field groups. If field groups have already been created, they can be customized via the top menu of the schema dialog. Fields can be assigned to field groups via drag and drop.
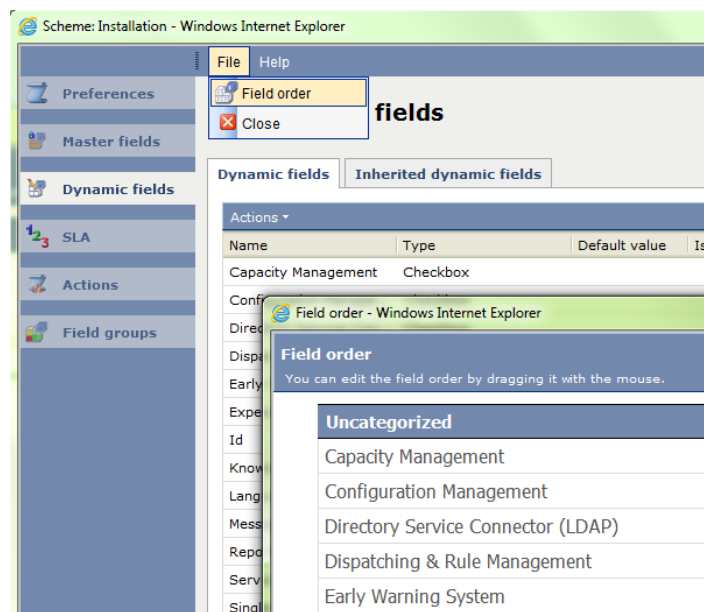

Figure 2.7:

SLA

The *SLA* tab allows for creating one or multiple Service Level Agreements in a CI schema. If a SLA service or priority is specified here, it will be taken into account for new tickets as soon as a CI using this schema is included in the ticket.

> **Please note**
>
> During the later calculation of SLM priorities, SLAs of linked CIs will be taken into account as well.

Actions

This tab allows for adding actions to a CI schema, comparable to ticket actions. They can be used in CIs depending on the status and user.

The actions to be configured on this tab are detailed in Chapter 3 of this document.

Field groups can be added via this tab. As soon as these field groups have been created, the existing dynamic fields can be grouped into these field groups. In a configuration item, these field groups will be displayed as tabs under FIELDS -> DYNAMIC FIELDS.

After the configuration of the schema has been finished, clicking on *Save* and then *Close* closes the window.
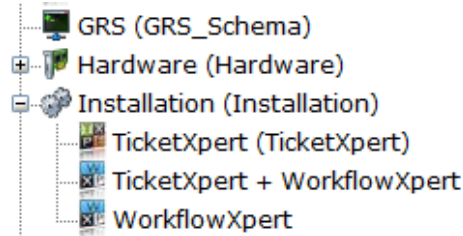

Figure 2.8: For types already linked to a schema, the schema will be displayed in parentheses

In the schema overview, all of the created schema types and CI schemas will now be listed independently from each other.

2.2.5 Zone Schemas and Default Hierarchy

Zone schemas are used for entering information specific for the zone on the one hand and for creating a specific default hierarchy on the other hand. This hierarchy in turn is needed for the CI zone import, provided that it is to be based on this default hierarchy.

> **Please note**
>
> Plese note that complications may arise during the import if the default hierarchy is deleted oder modified. If the existing default hierarchy is to be deleted or modified, please contact your project leader first!

Zones can be created in various depths and complexities. Additionally, the zone structure can depict either an organizational or geographic structure and thus differs from the schema hierarchy which is subdivided into various sorts and types of items.

Generally, the predefined zone schemas *Campus*, *Building*, *Floor*, and *Room* are available as a template and as an interlinked default hierarchy in an empty CMDB.

However, if other zone schemas are to be added or if a proprietary zone schema is to be used, proceed as follows:

Under SETTINGS -> CONFIGURATION MANAGEMENT (CMDB) -> CMDB SCHEMAS the entry *Zone schemas* has to be selected in the tree. The task list on the left now provides the button *Add zone schema* for creating additional zone schemas:

The following information is to be entered in the now opening dialog:

Name: The name of the zone. Localization buttons will be displayed as soon as multiple languages are activated in the *Xpert.NET* system.

Description: The text stored here is purely descriptive and describes the zone. The localization buttons will be displayed here as well as soon as multiple languages are activated in the *Xpert.NET* system.

Symbol: The icon for the zone schema. The icon can be used for an easier identification during the creation of the zone hierarchy later on. Instead of the preset default symbols, further symbols can be uploaded as well.
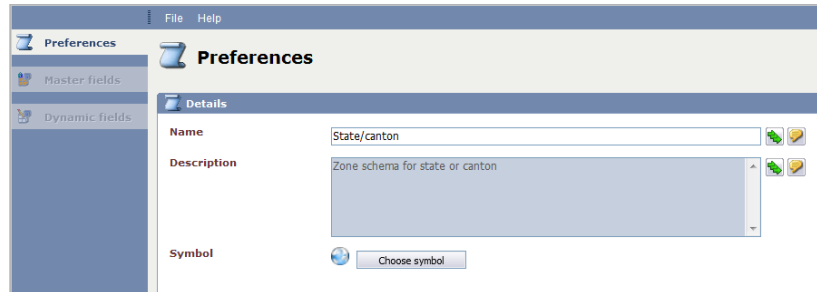

Figure 2.9:

After the first saving, two additional tabs for master and dynamic fields will be available.

Master Fields

The function of master fields is an information overview, similar to those in CI schemas, which displays the same predefined content for every zone based on this schema.

Via a click on the button *New field*, a new master field can be created. For this purpose, the **Name** of the master field is to be entered as well as the **Value** for this field. If the option **Inherit** is activated, this master field will be inherited by all zones within the CMDB structure that are below a zone that uses this schema in the hierarchy.

If master fields have already been created, the can be removed via the *Actions* menu.

If there are multiple master fields available, their position can be changed via the two arrow buttons in the center of the screen.
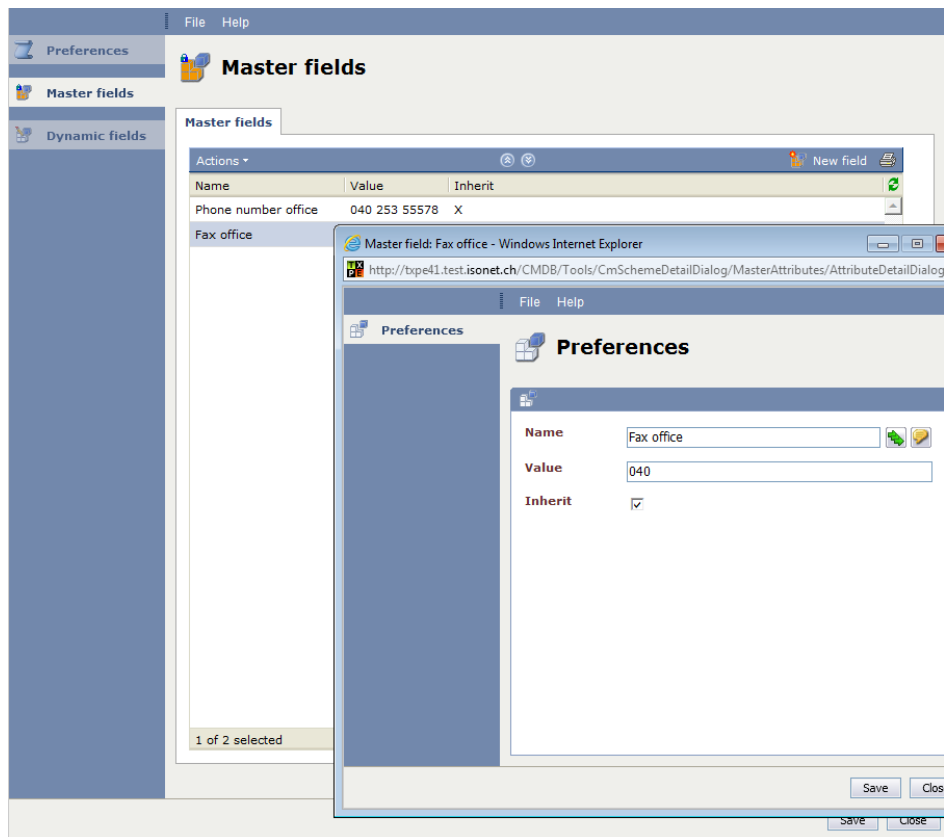
Figure 2.10: Zone master fields can be inherited as well

Dynamic Fields

Using dynamic fields, specific content can be managed and saved for every zone, e.g. a zone specific location ID. During a later import, CIs can be automatically assigned to the zones via such a zone dynamic field (depending on the version, only if the zones have been imported themselves).

Dynamic fields are created via the button *New field*. The *Actions* menu, which is only accessible if at least one field has been created, can be used to delete dynamic fields.

If there are multiple dynamic fields available, their position can be altered via the arrow buttons in the center of the view.

The following information has to be entered for creating a new dynamic field:

- **Name**: The name of the field as displayed in the respective zone left of the field later on. Localization buttons will be displayed as soon as multiple languages are activated in the *Xpert.NET* system.

- **Info text**: This text will be displayed as a mouseover text for the question mark symbol, which is located right next to the dynamic field in the zone.

- **Type**: Various predefined types can be selected here:
  - **Text**: Any text can be entered into this field.

- **Number**: Only numbers can be entered into this field.

- **Date**: Only date values can be entered via a date input field here.

- **True/False**: The dynamic field is turned into a checkbox that can be set active or inactive.

- **Selection field**: This type is used for creating a preconfigured list to select values from during zone creation.

- **Service time**: This field allows for selecting a service time for the Service Level Management.

- **Link**: Via this type, a link can be entered into the field. If an arbitrary text is entered into the dynamic field, it will be be automatically transformed into a link after saving the settings.

- **Text Area**: Instead of a text field, a larger text area is made available, which is able to contain larger amounts of text.

- **User Browser**: Via the user browser, *Xpert.NET* users can be entered into the dynamic field.

- **Localized text**: This type allows for entering text in multiple languages into the field.

- **Localized textarea**: Via this field type, larger amounts of text in multiple languages can be entered into the field.

- **Table**: This field type is used to create a table with multiple columns (the content of a row in the configuration will be displayed in a column later on). In the zone, multiple rows can be entered for every column.

- **IP Address**: This field type allows for entering an IP address.

- **Format**: If a certain format is to be specified for the selected type, it can be entered here. Examples for possible formats can be displayed via the small help icon on the right.

- **Default value**: The value entered here will be displayed in the respective field later on and can be subsequently edited.

- **Input is required**: If this field is activated, the dynamic field turns into a required field and has to be filled in before the CI creation can be completed.
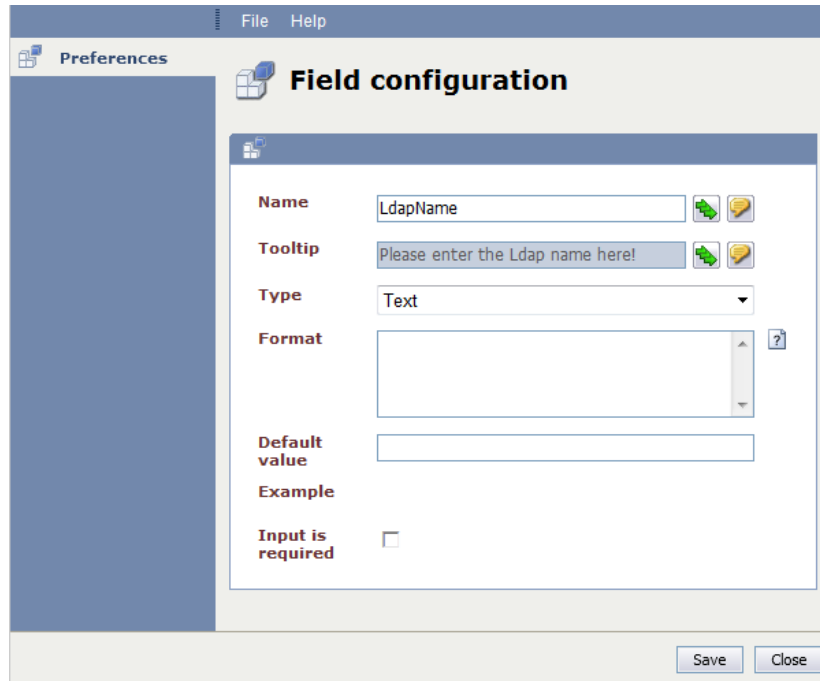
Figure 2.11:

2.3 Relation Types

Relation types allow for reproducing relations that can exist between real objects within the CMDB of *Xpert.NET* as well.

New relation types can be created via the button *New relation type*. Relation types not needed any more can be deleted via the *Actions* menu.

The following settings can be made in the configuration dialog for new relation types:

Name: The name of the relation type as displayed as soon as a CI is related to another CI. Localization buttons will be displayed as soon as multiple languages have been activated in the *Xpert.NET* system.

Description: A descriptive text for the relation type. Localization is possible here as well.

Direction: Relations with a direction can be seen from both sides. Example: The relation type *Contains* should be defined as *Is contained in* from the other side.

Reversed relation name: If the option for *Direction* has been activated, the relation name for the reversed view can be entered in this field.

3 CI Actions

This chapter details the configuration and application of the CI actions. After adding a CI action for the respective CI schema on the *Actions* tab, the respective configuration dialog will appear.

On the *CI statuses* tab, it is possible to define, in which status of a CI this action is to be visible and executable. The action will then be displayed in the CI detail dialog on the *Actions* tab or can be accessed in the respective tab or via the *Actions* menu.

In doing so, the status(es) during which the action is to be have to be specified under *Status*. Additionally, the button *Add user* has to be used in order to add the user(s) or the group that can execute this action. If the action is to be available for the same users during other statuses, a new status can be defined in the configuration dialog and then the previous status can be selected in the field **Take over from**. A click on *OK* will apply the configuration to the new status.

In using the *Add user* button, the following options are available:

Owner of the CI: Members of the group set as the owner of the CI can execute the action as soon as the previously defined status has been reached.

Creator of the CI: The user having created the CI or having been set as creator of the CI can execute the action.

Affected user of the CI: The user or all members of the group set as affected in the CI can execute the action.

Select user: A specific user or group can be selected via the user browser.

If a user or a group is to be added to all configured statuses, the option **Apply to all statuses** can be used. However, this option only applies to already configured statuses. If another status is edited after using this option, the user or group will not be added automatically there.

If a detailed configuration of an action is necessary, various configuration options can be found on the *Configuration* tab. These options will be explained individually in the following sections.

3.1 Change Visible ID

The displayed name of a CI (e.g. Workstation 23) can be changed afterwards via this action.

Configuration

This action does not have to be configured separately.

When a CI, in which the current user can execute this action, is opened, a checkmark icon will appear behind the current Visible ID on the *Details* tab of the CI.
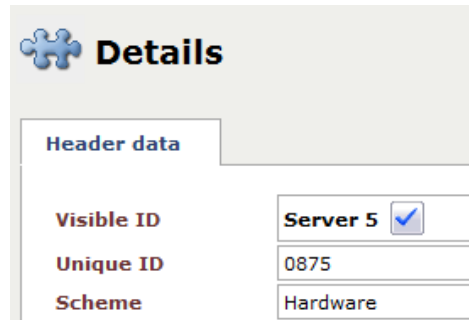

Figure 3.1:

After a click on this icon, the field will change into a text field and the name can be edited. After a second click on the icon, all changes will be applied. Clicking the x icon discards all changes.

3.2 Change Affected User

Users and groups linked to the CI as affected users or groups can be changed via this action.

Configuration

This action does not have to be configured separately.

Executing the Action

If this action is available for a CI, additional affected users or even groups can be added on the *Affected users* tab.
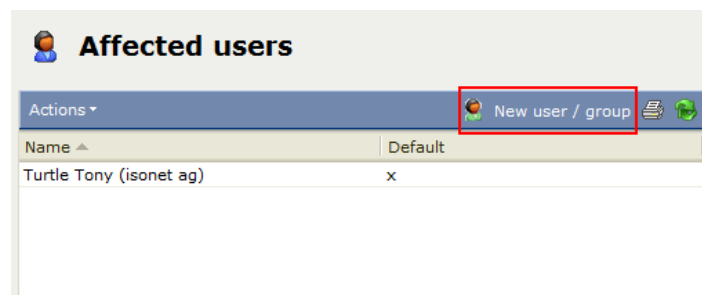

Figure 3.2:

The button *New user/group* opens a new dialog window, in which uses and groups can be added. The *Actions* menu allows for removing affected users or groups or for setting them as the new default values for affected users. Each user and each group not set as the default will only be another affected user/affected group.

3.3 Link to CI

Via this action, CIs can be interlinked using the available CI relation types. It is possible for a CI to contain another one (e.g. a computer can contain software or hard disks, which have been recorded in the CMDB as individual CIs), or for multiple CIs to be linked with each other (thus, a CI *Computer* can be linked to a CI *Monitor*). Furthermore, CIs can be copies of another CI or a variant thereof as well (e.g. in case of software).

If a CI is to be created in the tree view via the context menu (right click on a CI -> CREATE NEW CI), this actions has to be available to the user, because the new CI will be immediately linked with the clicked one. This action is necessary for linking CIs via drag and drop as well.

Configuration

By means of the option **Search only Visible ID**, the behavior of the CI Browser during performance of the action will be changed. Instead of all fields of a CI, only the Visible IDs of the CIs will be searched for all entries typed into the search mask.

Executing the Action

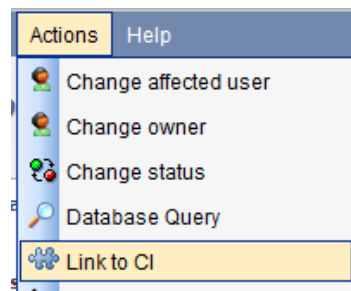In order to execute the action, *Link to CI* has to be selected from the top menu under *Actions*.

Figure 3.3:

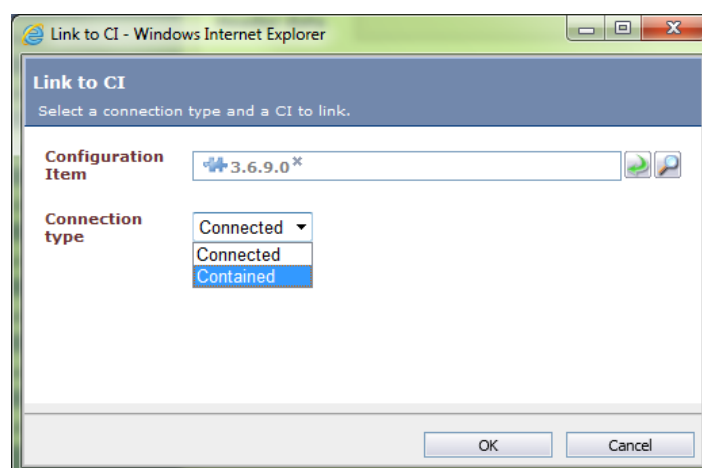In the now opening dialog, the designated CI and the desired connection type can be specified.

Figure 3.4:

A subsequent click on *OK* establishes the connection between the two CIs.

3.4 Database Query

Via this action, a database query can be executed from a CI. In order to display the data, a separate pop-up window will open, which will show the data in read-only mode. Thus, additional information on the CI, which can be gained from other data sources, can be displayed.

Configuration

Connection String: The connection string to the data source is to be specified here. Potential connection strings can be found on the following website:
http://www.connectionstrings.com/.

Query: The actual SQL query is defined here.

Figure 3.5:

View: The option *View* defines, whether and in which order the queried data is to be displayed. For example:

```
<dataview>
  <columns>
    <column name="DataId" column="DataId" />
    <column name="ObjectGuid" column=" ObjectGuid" />
  </columns>
</dataview>
```

For every column to be displayed in the result, a column definition (*column*) has to be specified here. The attribute *name* determines the column header to be displayed here. The attribute *column* links the view to the query. Thus, only titles contained in the query's result can be used here.

The configured action can be found in the top menu under *Actions* in the configuration item.
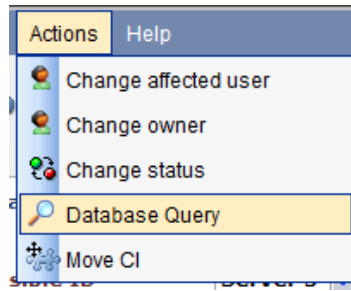

Figure 3.6:

After this menu item is selected, a new dialog window with the query's result will open (read-only).

3.5 Change the CIs Type

This action allows for changing an existing CI's type.

Configuration

This action does not have to be configured separately.

Executing the Action

The action can be executed in the same way as the action *Change Visible ID*. On the *Details* tab, click on the check mark behind the CI's type and select a new type for the CI. Clicking on the check mark a second time applies the change. Alternatively, the change can be discarded via a click on the cross icon.

> **Please note**
>
> The type of the CI can only be changed, if the following condition has been fulfilled: There has to be a main type with multiple sub-types in the CMDB configuration stage. Only the main type has a CI schema. If there is a CI with this schema, the type can be changed to the sub-types and back to the main type again.

3.6 Change Owner

Via this action, the owner of a CI can be changed afterwards.

Configuration

This action does not have to be configured separately.

Executing the Action

The action is executed as soon as the option *Change owner* is selected under *Actions* in the top menu.
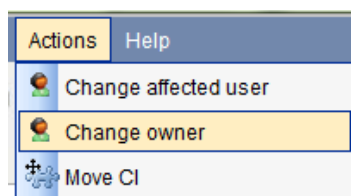
Figure 3.7:

In the subsequently opening dialog, the group to become the new owner of the CI can be selected. As soon as the dialog has been confirmed via a click on *OK*, the selected group is set as the new owner.

3.7 Start External Application

This action can start an external application either through a JavaApplet or VBScript. For example, a remote desktop connection to a predefined IP address can be established. In the process, the IP address can be read out from a CI field dynamically via expression.

Configuration

In order to configure the action, the following information has to be entered:

Loader type: As a loader type, either JavaApplet or VBScript can be selected. For security purposes, ActiveX may not be allowed within the network. For this reason, it is possible to switch the loader to Java.

Command: The default setting is the Windows remote desktop connection. If the command is to be customized, enter *mstsc /?* into the Windows command prompt to display further options.

In the command prompt window, the following command for a RDP connection is filled in by default:

```
mstsc /v:{$CI.Fields!ip.Text$}
```

In place of the expression {$CI.Fields!ip.Text$} it is possible to enter {$ CI.Fields["ip address"] $}, if the field name for the IP address is named ip address. Otherwise, each available expression can be used (see *Expression Reference* documentation).

Left icon: Via a click on the button *Choose menu icon*, the icon to be displayed in the menu can be set.

Executing the Action

By selecting the option *Start external application* under *Actions* in the top menu of the CI dialog, the command entered during the configuration is executed.

3.8 Add Comment

This action allows for adding comments to a CI.

Configuration

This action does not have to be configured separately.

The *Comments* tab provides the button *New comment*. Via this button, a comment can be entered just like in a ticket.

3.9 Change Supplier

The user specified as the supplier of a CI can be changed via this action afterwards.

Configuration

This action does not have to be configured separately.

Executing the Action

After clicking on *Actions* in the top menu of the opened CI and selecting the action *Change supplier*, a new dialog will open, in which the user to become the supplier of the CI can be selected.

3.10 New File

Via this action, file attachments can be added to a CI and deleted again as well.

Configuration

This action does not have to be configured separately.

Executing the Action

On the *Attachments* tab, the button *New file* is now available. Uploading files works in the same way as for tickets, for exampole. Files not needed any more can be deleted via the *Actions* menu.

3.11 Change Status

The current status of a CI can be changed with this action, irrespective of the workflow.

Configuration

This action does not have to be configured separately.

Executing the Action

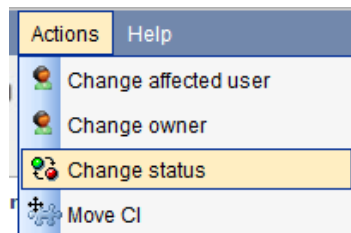The action *Change status* can be found under *Actions* in the top menu of the opened CI.

Figure 3.8:

In the subsequently opening dialog, a new status can be entered for the respective CI, depending on the life cycles.

4 Using the CMDB

4.1 Layout of the CMDB

The following figure illustrates the layout of the CMDB stage:
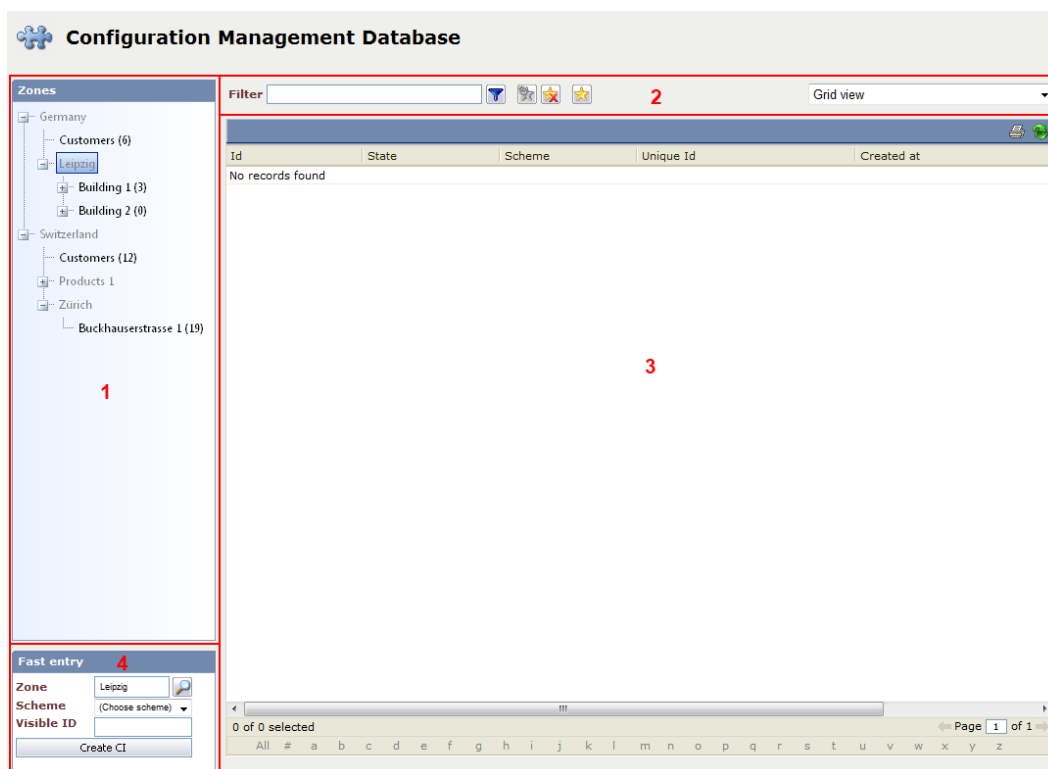

Figure 4.1: The CMDB window: (1) zone tree, (2) filters and views, (3) CI overview, and (4) fast entry

On the left side, the zone-tree displaying all of the zones (logical and physical ones) can be found. If they contain CIs, their amount will be displayed in a tool tip faded in by hovering the cursor over the respective zone. Via the context menu, zones can be created, edited, and deleted as well as new CIs created.

Right next to the zone-tree, the filter and the view settings are located. If a filter for CIs has already been set, those filter settings can also be saved as a favorite. The individual filter options are detailed in the *Search and Filter* documentation. Additionally, various view settings are available here:

Grid view: Displays all CIs of the current zone on the same level in a table. Connections to other CIs will not be displayed in this view.

Grid view (with contained CIs): All of the CIs of the current zone and the CIs contained by them are displayed in a table.

Grid view (with sub zones): Displays all CIs of the current zone and all of its sub-zones in a table. CIs with connections to the CIs in this zone will not be displayed here as well.

Grid view (with contained CIs and sub zones): Displays all CIs and additionally the CIs contained by other CIs and sub-zones.

Tree view: Displays all CIs in a tree view, starting from the current zone. If a CI contains other CIs, they will be displayed as child elements in the expandable tree view structure. Additionally, CIs not located in the zone but linked to a parent CI in the current zone will be displayed as child elements in the tree view as well. The top level of the hierarchy contains all CIs, which do not have parent elements or whose parent element is not located in the same zone.

Below the tree view, the CI overview can be found. It displays the CIs according to the view options described above. The CI dialog can be opened via a double click on a CI. The bar at the bottom of the dialog allows for displaying only CIs beginning with a certain letter or number.

Bottom left, the fast entry is located. CIs can be created quickly by specifying the zone, the schema, and the designated visible ID here. Via a click on the *Create CI* button, a CI is created and can be edited later on.

The filter directly above the CI overview can be used for finding CIs with a specific content quickly. For information on the filter options, the *Search and Filter* documentation is available.

> **Please note**
>
> The filters are only available in the grid views.

4.2 Managing the Zones

Using the predefined or custom zone schemas, a zone hierarchy can be assembled via the CMDB stage (*CMDB* module in the bottom bar).

In addition to normal zones based on zone schemas as described in the basic configuration, it is possible to create so-called top-level zones.

4.2.1 Top-Level and Sub-Top-Level Zones

Unlike the normal zones, top-level zones are "logical". The physical zones like *Campus* or *Building* can be nested into these logical zones, e.g. *Region*, *Country*, or *Continent*. The logical zones can be nested arbitrarily, but cannot contain any CIs themselves. For the sake of clarity, they are displayed in gray in the zone tree.

In order to create a new top-level zone, the context menu in the zone-tree can be opened (right-click on an existing zone). A click on *New top-level zone* in the menu creates a new top-level zone. The dialog opening subsequently only requires a **Name** for the top-level zone. After a click on *Save*, the top-level zone will be created always at the very top of the hierarchy. As soon as this zone has been created, further top-level or sub-top-level zones

can be added. Sub-top-level zones cannot contain CIs themselves, but are not limited to the top hierarchy level and thus can be created below normal zones. However, a zone CIs can be created in can be created here directly as well.
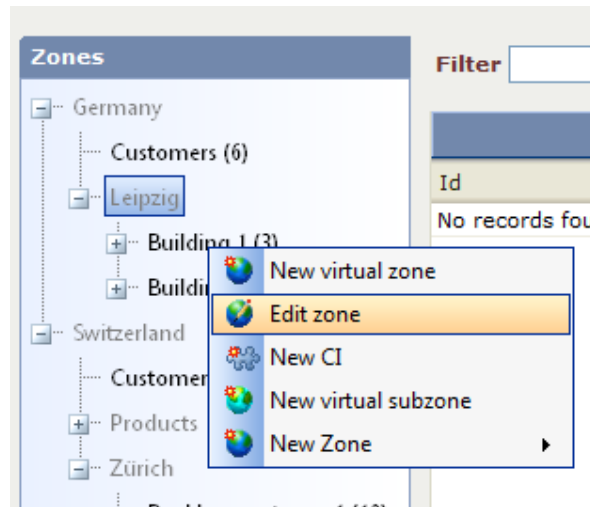

Figure 4.2:

4.2.2 Zones

If the zone hierarchy has not been customized, zones can be created via one of the four zone schemas: *Campus*, *Building*, *Floor*, and *Room*. If further or other zone schemas have been configured, they are available in the context menu. These zones are displayed in black in the zone tree and can contain CIs.

> **Please note**
>
> Top-level zones can only be created by root administrators. If the zone-tree does not contain any zones yet, the notification *No zones available* is displayed. Via a right click on this entry, new zones can be created using the context menu.

When a new zone is created via the context menu, it is necessary to define the zone schema the new zone is to be based on first. This determines the dynamic and master fields that can be displayed and filled out in the zone.

Subsequently, a new dialog window opens, in which the name and other zone details can be entered.

Name: The name of the zone as displayed in the zone hierarchy. The localization buttons will be displayed when multiple languages have been activated on the *Xpert.NET* system.

Description: Contains a descriptive text for the zone that can be localized as well.

Default: Affected user: Defines a default value for new CIs. The user or group defined in this field will be set as the default affected user or group for each CI created in the zone.

Default: Supplier: Defines a default value for new CIs. The user defined in this field will be set as the default supplier for each CI created in the zone.

Minimal number of CIs/Workflow: Defines a minimum number of CIs. If the amount of CIs falls below this value, the selected workflow (type: CI Zone) will be executed and the zone will be moved into this workflow as a workflow item.

Maximal number of CIs/Workflow: Defines a maximum number of CIs. If the amount of CIs exceeds this value, the selected workflow (type: CI Zone) will be executed and the zone will be moved into this workflow as a workflow item.
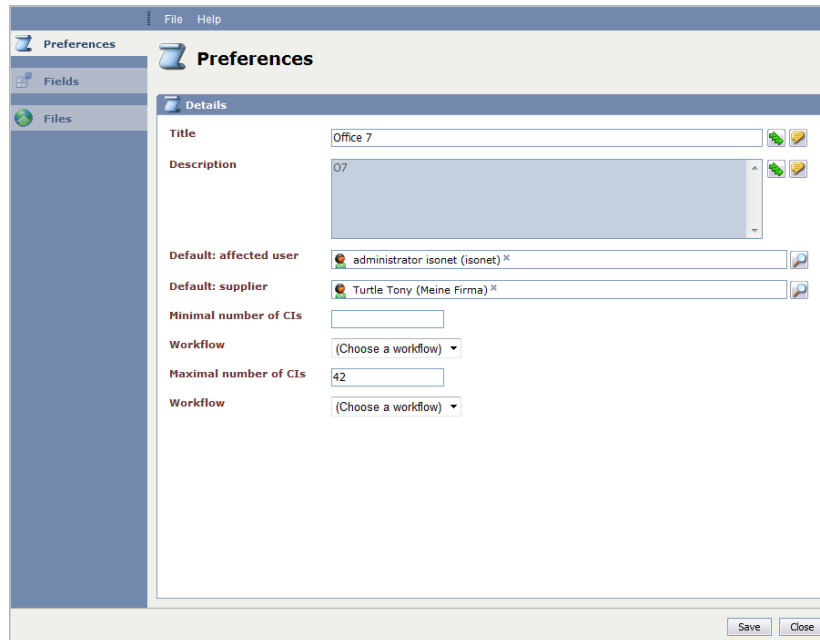

Figure 4.3:

After all values have been entered, the zone is created via a click on *Save*. The tabs *Fields* and *Files* will now be available. Under *Fields*, the dynamic fields can be edited and the master fields viewed. Under *Files*, file attachments to be used for all CIs in this zone or for a zone in general can be uploaded.

4.3 Creating and Managing Configuration Items (CIs)

After the zone hierarchy has been created, CIs can now be generated via the respective wizards. They can be created using the context menu of the zone tree, the fast entry, or the context menu of the tree view.

> **Hint**
>
> Relations between CIs can be created via drag and drop. For this purpose, click on a CI in the tree view and drag it onto another CI. For multi-page drag and drop of objects, use the mouse wheel or the cursor keys.

When a new CI is created via the context menu in the tree view or in the zone tree, a new window for selecting the zone and schema will open. Furthermore, a visible ID can be entered. Via a click on *OK*, the CI will be created.

Afterwards, the CI overview containing the following tabs (if a CI is created via the fast entry, it will be created with only the information entered into the fast entry and has to be opened manually) is opened:

Details: Contains several header data of the CI, the creation date, the visible ID, the creator and other information. The *Zones* tab allows furthermore for managing the affiliation of a CI to the different zones and for editing the visible zone via the Actions menu.

Affected users: Displays users and groups affected by the CI (e.g. the user using the CI "Computer 5"). New users can be added via the *New user/group* button, if the respective action has been configured for the CI schema.

Files: The file attachments for the respective CI can be viewed here. Deleting and uploading files is only possible if this action has been configured for the respective CI schema.

Fields: The master and dynamic fields are composed of the current schema's fields as well as the fields of all superordinate schemas. The master fields, however, are read-only. The dynamic fields can be edited and adjusted here. For this purpose, the button *Edit* below the dynamic fields can be used. In the process, all dynamic fields can be edited simultaneously. After a click on the button *Save field value*, all entries will be saved.
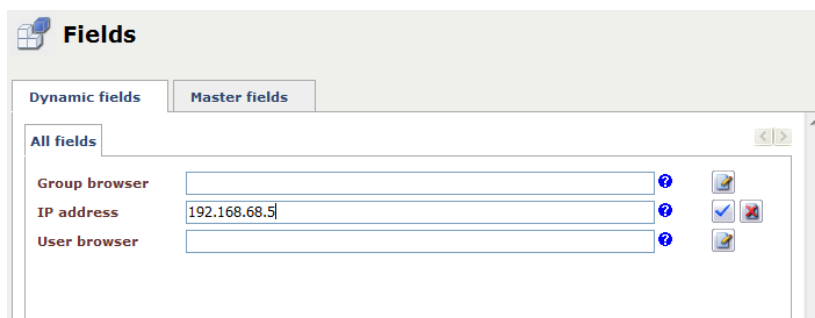

Figure 4.4:

Relations: On this tab, all contained and connected CIs are displayed and can be opened via a double click.

Tickets: Displays all created tickets that are linked to the CI or that contain a reference to the CI (e.g. if the CI has been added to the ticket via a ticket action).

Services: This tab is only displayed if the Service Portfolio has been activated. It displays all linked services of the Service Portfolio. Additionally, it is possible to add a new service via the *Actions* menu, to mark the CI as a service CI when adding it to a service, or to remove the connection to services.

History: Contains a chronological log of all changes to the CI.

Comments: Existing comments on CIs are displayed here. If the action has been configured, new comments can be added as well.

SLA: If the CI is connected to a Service Level Agreement, it is displayed here. New SLAs can be added here as well. For example, if a ticket is created and linked to the CI simultaneously, its SLA can be used to calculate response times as well.

4.3.1 Visible ID and Unique ID

For every CI, there are two types of identification available: the visible ID and the unique ID. If no information is entered during the CI creation, the unique ID is generated from the configuration settings in the CI schema and the visible ID is set to the unique ID. An arbitrary text can be entered for the visible ID; it does not have to be distinct. It is furthermore possible to enter an arbitrary text for the unique ID as well. During the creation process of the CI, however, the system checks whether the unique ID is distinct over the whole CMDB. If it is not, an error message will be displayed.

Alternatively, it is possible to transorm the visible ID into an additional distinct ID as well via a configuration in the database.

4.4 Moving CIs and CI Relations

CIs can be moved between different zones. Whenever a CI is moved from one zone to another, all relations to other CIs (parent, child, etc.) will be sustained. No relations will be deleted, but linked CIs will not be moved as well. This means that a CI can be related to CIs in other zones. Thus, when CIs are moved manually, it is necessary to check whether the relations are still up-to-date afterwards.

5 Import

Within the CMDB, zones as well as CIs can be imported. Affected users, suppliers, and owners are filed in the User Management and cannot be imported via the CMDB. For those entities, it is necessary to use the import function (LDAP) of the User Management and/or to create the respective users and groups manually.

5.1 Requirements

The following sections describe the import of CIs. It is assumed that an existing, empty CI database is available.

5.2 CI Import

Before the CI import can be started, it is necessary to create the respective structure, consisting of the life cycles, schemas, zone schemas, and zones. The CI import is configured under SETTINGS -> CONFIGURATION MANAGEMENT (CMDB) -> IMPORT. Via the button *New Import*, a new CI import can be configured.

5.2.1 Import Configuration

In the following dialog, the basic information for the import has to be entered:

- Name

- Description

- Import type (*Import Configuration Items* has to be selected here)

- Schema (into which CI schema is to be imported?)

- Zone (into which zone are the CIs to be saved by default?)

- Method (defines what will happen, if the imported CI does already exist in the CMDB):

  - Ignore existing CIs or create new CI: the existing CI is not changed, no new CI is created;

  - Replace all existing CIs or create new CI: the existing CI is changed;

  - Create new CI: creates a new CI in any case;

  - Delete all existing CIs and create new CI;

  - Replace all existing CIs without creating new CI: just like *Replace all existing CIs or create new CI*, but no new on CI is created;

  - Delete all existing CIs or create new CI: deletes the existing CI and no new CI is created.

- Data Source (depending on the system, miscellaneous database selection options are available here)
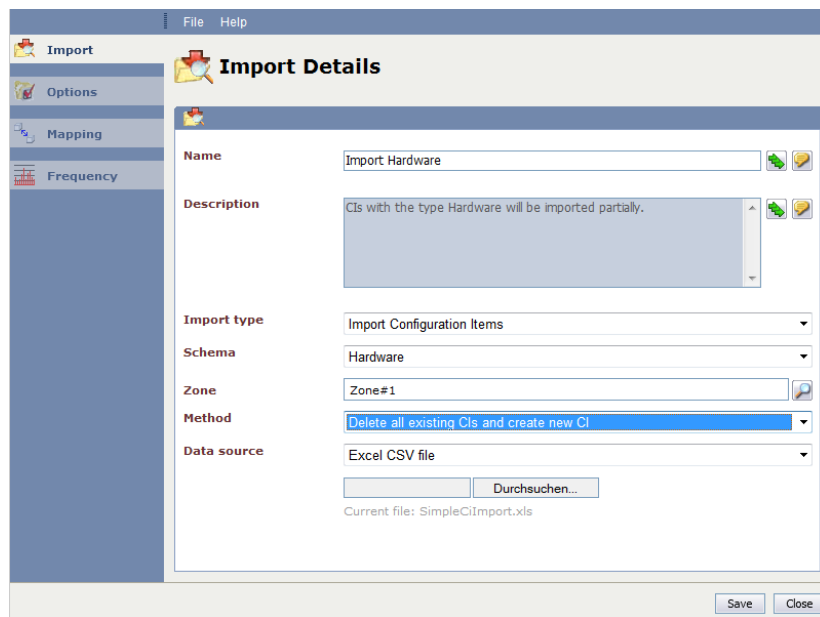

Figure 5.1:

Depending on the data source, further information may be required.

Depending on the database type selected, a valid connection string is to be entered. Examples for connection strings are available under www.connectionstrings.com. In case of connection problems: The connection has to be established from the CMDB application server to the database server (service *isonet Agent*). As the next step, a SQL query has to be entered for the import. An arbitrary amount of columns can be returned by the request, but there should be at least one distinct column, which can be used as a primary key.

The following data sources can be used:

- Active-Directory

- Excel CSV file

- Microsoft SQL Server Compact Data Provider

- Odbc Data Provider

- OleDb Data Provider

- OracleClient Data Provider

- SQLClient Data Provider

As an example, Excel/CSV and Active Directory data sources will be illustrated here.

Excel CSV file: Simply select the Excel or CSV file, from which to import. For this purpose, click on *Browse...*, select the file from the local computer, and confirm the selection with *OK*.

Active Directory: This data source type can be used for Active Directory and most of the other directory services as well. Here, it is crucial to enter the credentials for the LDAP

access and to select an authentication type (for unsecured connection types, FastBind is recomended; in case of failure, contact the directory services administrator or try other authentication types).

Next, the LDAP path to the hierarchy level to be imported has to be entered as a Connection string. After a click on the *Refresh LDAP Object-classes* button, the available object classes will appear in the drop-down-list. If an error is displayed in red, the entries have to be checked. In case of connection problems: The connection has to be established from the CMDB application server to the specified directory server (service *isonet Agent*).

5.2.2 Options

After saving the data, the other tabs on the left side will be available. On the *Options* tab, the following settings are available:

Delete relations: The import makes sure that CIs will only have the newly imported relations. If this option is not checked, old relations will always be left unchanged.

Detect conflicts: This option activates the conflict detection system. More information on this can be found in Section 5.5.

5.2.3 Mapping

The mapping makes sure that

- the contents of the various columns of the import data source are written into the correct fields of the CI schema,
- the uniqueness of the imported CIs is ensured (primary key column), and
- the connections between CIs will be established (foreign key).

Via the button *New Mapping*, new entry are created in the mapping table. The following setting options are available at the beginning:

Mapping class: The *Configuration Item* has to be selected here. *Zone* is necessary in order to move a CI into a zone automatically. It is to be noted that depending on the CMDB version, the mapping class *Zone* can only be used, if the zones have been imported as well.

Key type: Various key types can be selected here:

- **Primary key**: The unique column of the import.
- **No key**: Use this key for filling in fields without a key function.
- **Foreign key**: Foreign key are used for establishing connections between CIs.
- **Secondary key**: Serves as an alternative primary key.

Primary Keys

An import that is supposed to create new CIs requires a primary key. After changing the key type to *Primary key*, the following or a similar window can be seen:

Figure 5.2:

The following fields are required for creating a primary key mapping:

Import schema: The CI schema to be used.

Attribute: The dynamic field of the CI schema, in which the primary key column is to be imported, must be selected here.

External ID: The respective column in the import data source.

No Key

Presumably, there will be some columns in an import, which are only to be transferred into the respective dynamic fields. For this purpose, the key type *No key* has to be used:

Figure 5.3:

The following fields are necessary, in order to create a normal field mapping (without key):

Import schema: The CI schema to be used.

Attribute: The dynamic field of the CI schema, into which the field is to be imported.

External ID: The corresponding column in the import data source.

Predefined value: If this option is checked, a static value will be entered into the CI field. Then no column will be referenced from the import data source for the mapping.

At the beginning, the following special attributes are available in the attribute list:

Type: Sets the type of the CI to a type covered by the schema. The content of the import column (External ID) has to match the import identification of the type. Thus, each type to be set by the import has to have a matching import identification.

Status: Sets the status of a CI to a status of its life cycle. The content of the import column (External ID) has to match the import identification of the status. Thus, each status to be set by the import has to have a matching import identification.

Supplier: This sets the supplier of the CI. The supplier can be a user or a group. At least one attribute of the user or group has to be selected (multiple selection is possible), in which the system can search for the content of the import column (External ID).

Affected user: This sets the affected user of the CI. The affected user can be a user or a group. At least one attribute of the user or group has to be selected (multiple selection is possible), in which the system can search for the content of the import column (External ID).

Owner: This sets the owner of the CI. Only a group can be owner. At least one attribute of the group has to be selected (multiple selection is possible), in which the system can search for the content of the import column (External ID)

Visible ID: The visible ID is used for the user-readable (ideally: meaningful) ID of the CI. It is recommended to map a distinct column with the visible ID. If no visible ID is filled in, it will obtain the same number that is used for uniqueness internally (can be configured in the CI schema) automatically.

> **Please note**
>
> The visible ID mapping always has to be made in addition to other mappings (Primary key, No key).

Foreign Key

In order to create relations between various CIs, the key type *Foreign key* is necessary. It is possible to link any CIs to each other - the manually created ones as well.

Figure 5.4:

The following fields are required in order to create a mapping via foreign key:

Parent schema: The schema of the CIs to be linked.

Attribute: The attribute of the CIs to be linked, on which the connection is to be based.

Import schema: The schema of the CIs to be imported.

External ID: The corresponding column in the import data source.

Relation group: This option is used for grouping individual foreign key configurations. It is recommended to enter the value from *External ID* here.

Relation type: Specifies the relation type the CIs are to be linked with.

Please note, that a foreign key mapping only creates a relation, but does not fill in dynamic fields. If the column from the import source is to be filled into a field as well, an additional *No key* mapping is to be defined.

Secondary Key

Sometimes, a primary key does not suffice for the localization of existing CIs, e.g. if two seperate imports, each with its own primary key, are to update the fields of the same CI. If an additional secondary key mapping is defined, the import first searches for an existing CI with the matching primary key, and if no matching CI can be found, via the configured secondary key.

Figure 5.5:

The following fields are required for a secondary key mapping as well:

Import schema: The CI schema to be used.

Attribute: The dynamic field of the CI schema, into which the secondary key column is to be imported, has to be selected here.

External ID: The corresponding column in the import data source.

5.2.4 Frequency

Another option is to set imports to be performed periodically. This tab can be ignored, if the import is only to be started manually.

In order to configure a regular automatic import, the option Active must be activated as a first step.

Figure 5.6:

The following options are available for setting an exact time for the import:

After another import: The import starts as soon as the specified other one has been finished.

At a scheduled time: Here, an arbitrary amount of dates, on which the import is to start, can be defined as a schedule:

Figure 5.7:

As an alternative, the time interval option can be used, as shown in the figure below:

Figure 5.8:

5.2.5 Examples of Use for the CI Import

Importing CIs from a Data Source

The goal is to import CIs from an arbitrary data source, during which user-defined fields are to be mapped.

During the configuration of a new import, the zone the CIs are to be imported into is be selected, along with the desired data source (database, Excel, etc.). In this example, no automatic zone assignment is performed yet. Additionally, the schema the CIs are to be created for has to be selected.

For this purpose, select the option *Replace all existing CIs or create new CI* as a method. It is the recommended method for standard cases.

Using the *Mapping* tab makes sure that CIs are created the right way. First, a primary key mapping has to be created. For every dataset to be imported, the importer first checks via the primary key, whether a CI already exists. If no CI with the current value as the primary key exists, a new CI will be created. If, however, an existing CI is found, it will be overwritten. For this purpose, a column (External ID) is to be selected from the import data source, which is known to be distinct for the specified CI attribute!

Usually, additional columns of the import data source are mapped to certain CI fields. For this purpose, a mapping of the type *No key* has to be created for every field and the External ID (field from the data source) has to be selected as well as the field of the CI schema the data is to be filed into.

The visible ID is a specific feature: It is a special field for identifying a CI via a meaningful, preferably distinct key. It is not obligatory for the visible ID to be distinct for all existing CIs (the CMDB creates another distinct ID for each CI on its own), but it is recommended.

It must be ensured that a mapping on the visible ID has been created for every import that is used for creating new CIs. Of course, the visible ID can equal the primary key; however, a separate mapping must be created for the visible ID regardless.

Importing Relations Between CIs

Relations can be created by using the mapping key type *Foreign key*. For example, if computers and monitors have been imported, the field *Display* will be imported, in which the serial number of the connected display is specified – which matches the field *Serial ID* of the monitor CI. In order to connect these two CIs, a new mapping has to be created with the key type *Foreign key* for the computers during the import. The schema used for monitors is selected as a parent schema and the serial ID as an attribute. The import schema is left on *Computer*. Now the external ID (field in the import data source) must be set to *Display* (or the column containing the serial number of the display).

In the process, the field for the relation group can be ignored and any character string, e.g. the external ID, filled in.

Finally, the relation type has to be selected, e.g. *Connected*.

Importing CIs with a Data Crossover with Another Import

It may happen that a company has multiple databases, which return different information for the same datasets (CIs). For instance, the facility database of the accounts department can return different data than the automatic inventory scan software for IT equipment, regardless of the fact that both data sources partially refer to the same CIs.

Via the mapping type *Secondary key*, two imports can be configured, which maintain different fields of the same CI. It is, however, important that both import data sources have a corresponding field, in addition to their respective primary key fields.

Example

- Facility database: Primary key *FacilitiesID*,
- Inventory scanner: Primary key *Hostname*,

- Corresponding field: *SerialNumber*.

For the primary keys, the data within the import is distinct; there are, however, gaps in the serial numbers - for some facilities, the serial number is missing, and the scanner cannot read out the numbers of all the IT equipment.

In the last step, two imports have been defined, *Facilities data* and *IT equipment*, each with the desired number of fields (*No key*) as well as one primary key each (*FacilitiesID* and *Hostname*).

In order to avoid that every import creates only its own CIs and instead only updates an existing CI's data in case of matching serial numbers, a new secondary key mapping is created for each of the two imports. The goal is that every one of the imports searches for an existing CI via the serial number first, and if none can be found, the respective primary key will be tested. Thus, the configuration has to look as follows for every import:

- Facilities data: Primary key *SerialNumber*, secondary key *FacilitiesID*
- Scanner: Primary key *SerialNumber*, secondary key *Hostname*

Whenever a CI is found via the serial number, its fields will be overwritten. If no CI can be found via the serial number (or if the import dataset does not include a serial number), the secondary key is used.

> **Warning**
>
> If both imports overwrite the same fields, import conflicts will be generated (if the conflict detection is activated).

Placing CIs in Zones

In order to be able to store CIs not only in the standard zone but in any existing zone, the corresponding zone information has to be entered into the import data source. This could be, for example, a zone code like *DE_HH_123*. It is, however, important that exactly the same codes have been written into a user-defined field in the zones. (A seperate field has to be created in the zone schema for this.)

For an automatic assignment, a new entry has to be added to the import mapping table. Select the mapping class *Zone*, then the zone schema to be used for the automatic assignment, and finally the field in this zone schema containing the zone code, which is also used by the import data source.

Assigning Statuses

The life cycle status of a CI is handled like a normal field. For this purpose, create a *No key* mapping and to use the status as attribute.

For the status-mapping to work, the correct import identifications have to be entered into the life cycle status.

If the import identification matches the value from the import data source, the CI's status will be set to the new status automatically.

Synchronizing two Imports with Conflict Detection

As soon as the CMDB has been filled with the initial data, the continuing individual imports will only be used for synchronizing the CMDB with the import data sources. In a case like 5.2.5.3 (multiple imports write into the same CI) conflicts may be generated (see Section 5.5).

The conflict detection is activated by default for every import. For some initial imports or for imports only used to correct a field, it may be sensible to deactivate the conflict detection temporarily. The import then will change the data of an existing CI without generating conflicts.

> **Warning**
>
> Use the conflict detection, if you expect that data differences for the same CIs may exist in your import data sources and you would like to detect and correct them.

5.3 Zone Import

In order to configure a zone import, a new import has to be created via Settings -> Configuration Management (CMDB) -> Import.

5.3.1 Import Configuration

The basic information for this import has to be entered in the following dialog:

Name: The name of the import.

Description: A short descriptive text for the import can be entered here.

Import type: *Import zones* must be selected here.

Schema order: Which available standard hierarchy is to be used for the zone import?

Schema: This is the zone schema to be used for the import.

Zone: This is the zone, in which the new zones will be generated.

Method: Defines, what happens if the import zone already exists.

- **Ignore existing zones or create new zone**: The existing zone is not altered, no new zone is created.

- **Replace all existing zones or create new zone**: The existing zone is updated with more recent data.

- **Create a new zone**: Creates a new zone at any case.

- **Delete existing zones and create new zone**.

- **Replace existing zones but do not create new zone**: Like *Replace all existing zones or create new zone,* but no new zones are created.

- **Delete existing zones or create new zone**: Deletes the existing zone; no new zone is created.

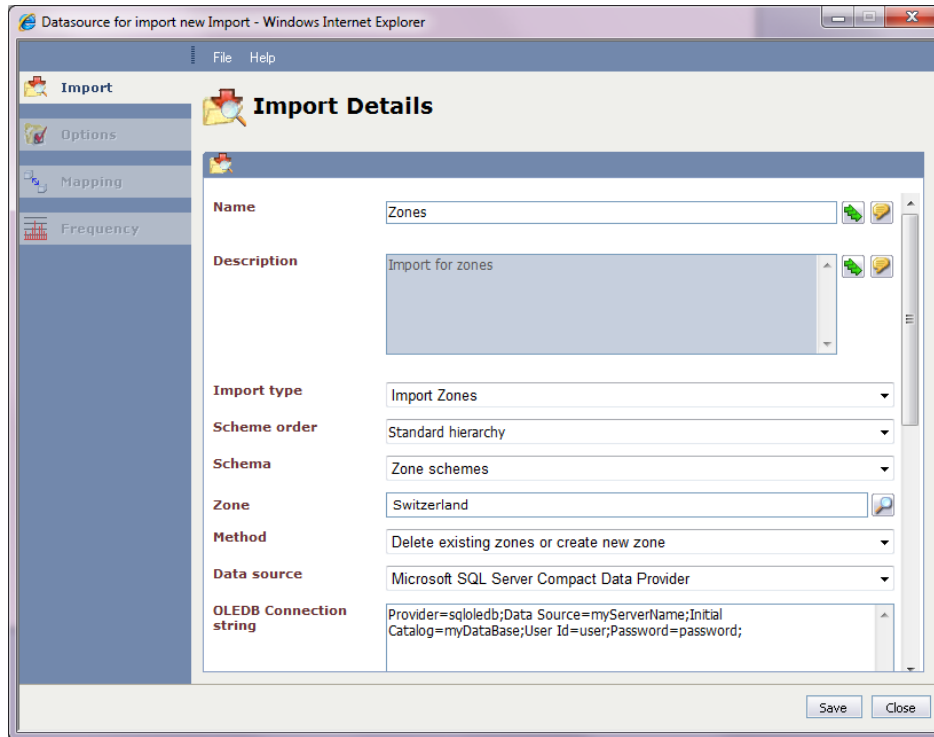**Data source**: The source of the origin data (e.g. LDAP) of the import.

Figure 5.9: Depending on the data source, additional information may be required

5.3.2 Options

After the data has been saved, the other tabs on the left will be available for configuring the import. On the *Options* tab, the following options are available:

Disable the zone hierarchy: This option deactivates the automatic assignment of zone schemas (standard zone hierarchy).

> **Warning**
>
> If this option is not activated, a standard zone hierarchy is necessary, otherwise no import can be executed. If the standard zone hierarchy is to be deleted, please contact your project leader first!

The options *Delete relations* and *Detect conflicts* may be available here as well. In this case, the zone import is an older version – the import must be saved without change. The options mentioned above should subsequently disappear, as the system has refreshed the import.

5.3.3 Mapping

The mapping makes sure that

- the contents of the various columns of the import data source are written into the correct fields of the zone,

- the uniqueness of the imported zones is assured (primary key column), and

- relations between zones will be established for the hierarchy (foreign key).

A click on *New Mapping* creates a new entry in the mapping table. The following setting options are available at the beginning:

Mapping class: Only *Zone* is available here.

Key type:

- **Primary key**: The distinct column of the import.
- **No key**: This key type is used for filling in fields without a key function.
- **Foreign key**: With this key type, connections between zones can be established.

Primary Keys

An import, which is to create new zones, requires a primary key. After the key type has been set to *Primary key*, the following window is displayed:
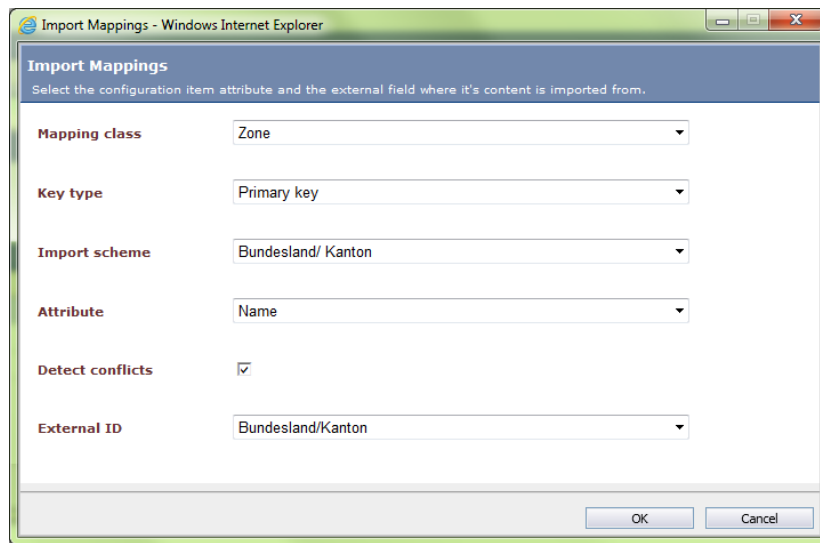

Figure 5.10:

The following fields are required for creating a primary key mapping:

Import schema: The zone schema to be used.

Attribute: The dynamic field of the zone schema the primary key column is to be imported into has to be selected here.

External ID: The corresponding column in the import data source.

No key

Presumably, there will be some columns in an import that are only to be transferred into the respective dynamic fields. For this purpose, the key type *No key* can be used:
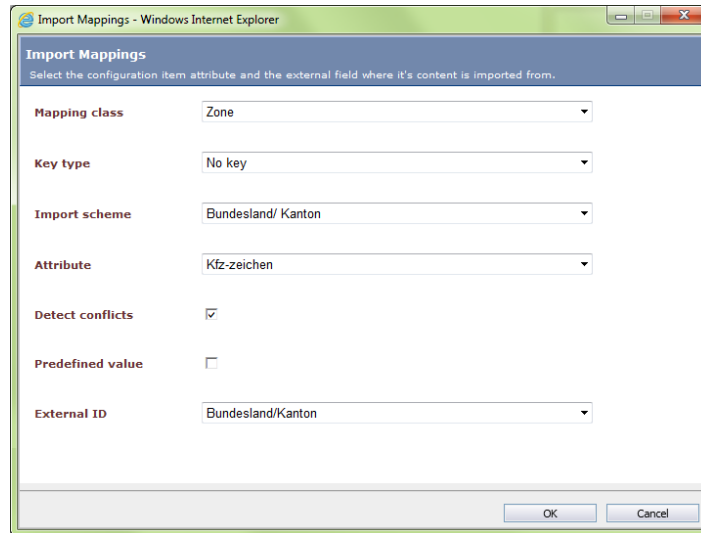
Figure 5.11:

The following fields are necessary in order to create a normal field mapping (without key):

Import schema: The zone schema to be used.

Attribute: The dynamic field of the zone schema, into which the field is to be imported.

External ID: The corresponding column in the import data source.

Predefined value: If this option is checked, a static value will be entered into the zone field. Thus, no column will be referenced from the import data source for the mapping.

Foreign Key

In order to create relations between different zones, the key type *Foreign key* is needed.

The following fields are required for creating a mapping via foreign key:

Linked import: The zone import this mapping refers to.

Linked mapping: The primary key entry in the mapping table of the linked import.

External ID: The corresponding column in the import data source.

Please note that a foreign key mapping only creates a relation, but does not fill in dynamic fields. If the column from the import source is to be filled into a field as well, an additional *No key* mapping has to be defined for this purpose.

5.3.4 Examples of Use for the Zone Import

Basic Import of Zones with the Same Schema, Without Standard Hierarchy

If zones are to be imported without regard for the standard zone hierarchy, the following procedure is necessary, if these two criteria match the case:

- The zone hierarchy is not limited and displays a different depth for every virtual zone.

- Only an universal zone schema is to be used.

In this case, the option *Disable the zone hierarchy* must be selected in the import options in the zone import.

If the import data source does not contain any hierarchy information (e.g. parent zone), it is only possible to import the zone as flat. The procedure is easy here:

- A primary key mapping must be created on the distinct zone label.

- Furthermore, a mapping must be created on the Visible ID for the zone to be named.

- Optional: Creating an arbitrary amount of *No key* mappings for user-defined fields.

If the import data source contains a column with hierarchy information (e.g. parent zone), this information can be used by creating a *Foreign key* mapping onto this field. The rest is processed automatically, provided the entries in the field *Parent zone* match the name of the zone.

Import with Automatic Schema Assignment Using the Standard Hierarchy

The import is more complex, if the imported zones are supposed to be created using a specific zone schema each (e.g. if different schemas are used for City, Building, Floor, Office, etc.). This can be taken into account during the import, if a corresponding standard zone hierarchy has been defined. For this purpose, a zone hierarchy has to be defined in the CMDB schema administration by moving the zone schemas onto the entry *Standard hierarchy* via drag and drop .

> **Warning**
>
> Only use your defined hierarchy during the import. The zones will be assigned to the corresponding zone schemas due to their hierarchical level. The imported hierarchy cannot be deeper than the standard hierarchy.

> **Warning**
>
> You have to create its own field mapping for every zone schema, so the import mapping is more complex in this case.

5.4 Manual Execution of an Import

In order to start an already configured import manually, navigate to Settings -> Configuration Management (CMDB) -> Import. Highlight the import to be used and select the action *Execute import* on the left. The following dialog will appear subsequently:
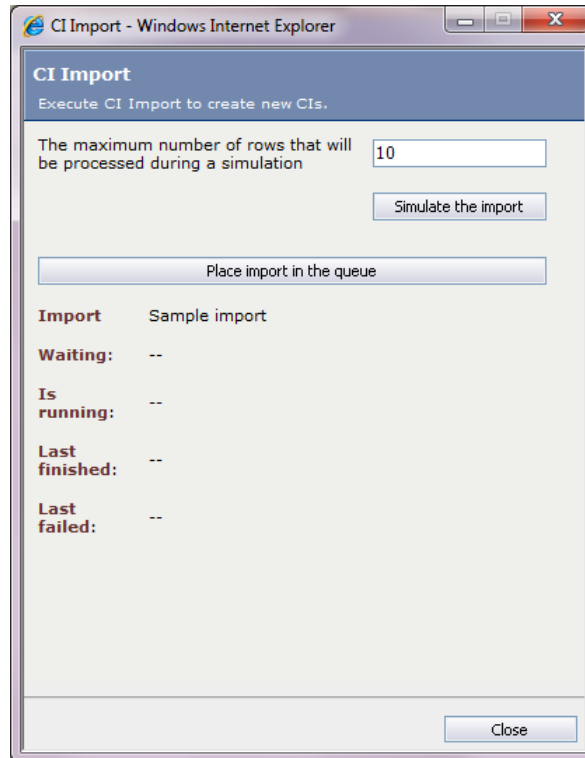
Figure 5.12:

Via the button *Simulate the import,* the import is executed without changing any data. Subsequently, the results of the simulated import will be listed in a table. With this function, the general functionality of the import can be checked.

The button *Place import in the queue* initializes the import's execution. The import will be executed in the background, i.e. if other imports are executed at the moment, it can take some time before it will be an import's turn. This dialog can be closed and opened again later on in order to look after the results.

As soon as the information fields *Waiting* and *Is running* are empty and *Last finished* has been updated, the import has been executed. The CIs or the zones have now been updated in the CMDB.

5.5 Conflicts

The conflict system monitors fields and relations of zones and CIs and makes sure that data is not overwritten unwittingly. If the conflict system notices a possibly forbidden modification, it prevents it and generates a corresponding conflict.

Fields and relations can be found in the following statuses (initial condition):

- Empty,
- last modification by own import (view from import),
- last modification by other import (view from import), and
- last modification by user.

Analogously, the following data modification options are available:

- By user,

- by import.

The conflict trigger mechanism can be found in the following table:

Initial condition	Modification by user	By import
Empty	OK	OK
Own import	Conflict	OK
Other import	Conflict	Conflict
User	OK	Conflict

Conflicts resulting from data modification by the user directly in the user interface are solved on the spot.

Example

A user opens an imported CI and tries to change the content of an imported field. A conflict arises and results in an alarm dialog (Do you really want to change this field?). If the user presses the *OK* button, the conflict is solved directly and the field content is changed. The status of the field is now changed to *Last modification by user*.

Conflicts generated during an import arise in the background and thus cannot be solved in the user interface immediately. Such conflicts are collected by the system in a conflict queue. Every import has its own queue. The following figure shows the solved conflicts of the import *Terminals*:
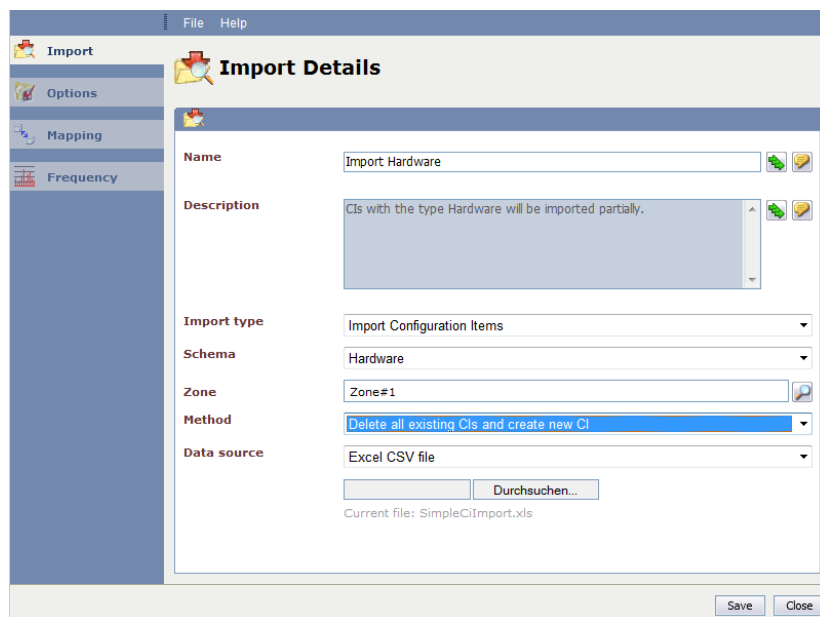
Figure 5.13:

> ▌ **Warning**
> ────────────────
> If a conflict arises, the underlying data change has not been processed! If content is missing in your imported datasets unexpectedly, first check whether there are any conflicts listed!

5.5.1 Solving Conflicts

If there are unsolved conflicts, the respective import has to be highlighted with a left click and then the *Manage conflicts* button on the left is to be clicked. A table with all of the conflicts and the corresponding solution actions will appear.

In order to edit conflicts, they can be highlighted (it is possible to highlight more than one using the [CTRL] key as well) and then the *Open conflict* button needs to be clicked.

In the following dialog, the details to any selected conflict can be seen and one of the following solution options can be selected:

Apply: Solves the conflict. The data change prevented by the conflict will be applied during the next import. The conflict dataset remains unchanged.

Ignore: Solves the conflict. The data change prevented by the conflict will not be applied during the next import. The conflict dataset remains unchanged.

Unsolved: Deletes the conflict dataset. No data change is applied.

The options *Apply* and *Ignore* cause the conflict system to memorize the solution, in order to solve future conflicts in this dataset in the same way automatically. The option *Unsolved* simply causes a deletion of the conflict, i.e. it can appear again later on.

> ▌ **Please note**
> ────────────────
> After clicking on *Apply*, the data change is not performed yet. The import has to be executed one more time!

5.5.2 Conflicts with Visible ID

If the setting for the visible ID has been configured to only be allowed to occur once in the system, and if a CI with a visible ID that already exists on the system is to be imported, a conflict will be triggered. In order to solve this conflict, the CI's visible ID has to be renamed in the source.

6 Statistics and Change Management

6.1 Statistics

Created by	Maik Wisatzke
Creation date	20.04.2009
Doc-ID	DOC-190612-010
Version	2015-1
Status	Approved
Replaces version	3.10-3
Release date	12.06.2015
Valid from	Immediately
Valid until	Cancellation
Document name	CMDB 2015-1_EN

6.2 Change Control

Version	Date	Executed by	Comments
3.0	18.08.2010	Maik Wisatzke	New design/several linguistic revisions/Adjustments for version 3.4
3.1	05.04.2011	Marco Mehl/Maik Wisatzke	Minor adjustments/Actions added
3.2	08.11.2011	Marco Mehl/Maik Wisatzke	Note on direct access and unique/visible IDs; general revision; new figures
3.3	14.05.2012	Maik Wisatzke/Steffi Kurnot	Linguistic revision; features of v. 3.8 integrated
3.4	12.07.2012	Maik Wisatzke	Added Tooltips for Cis; conflicts with visible ID; new method to edit dynamic field values; dependencies of SLM priorities
3.8-1	03.07.2013	Maik Wisatzke	Linguistic revision, Information for moving and linking tickets added
3.8-2	27.11.2013	Anna Hajduk	Linguistic revision
3.8-3	22.01.2014	Maik Wisatzke	New design adapted, new figures
2015-1	12.06.2015	Alexander Schmidt	Updates for 2015