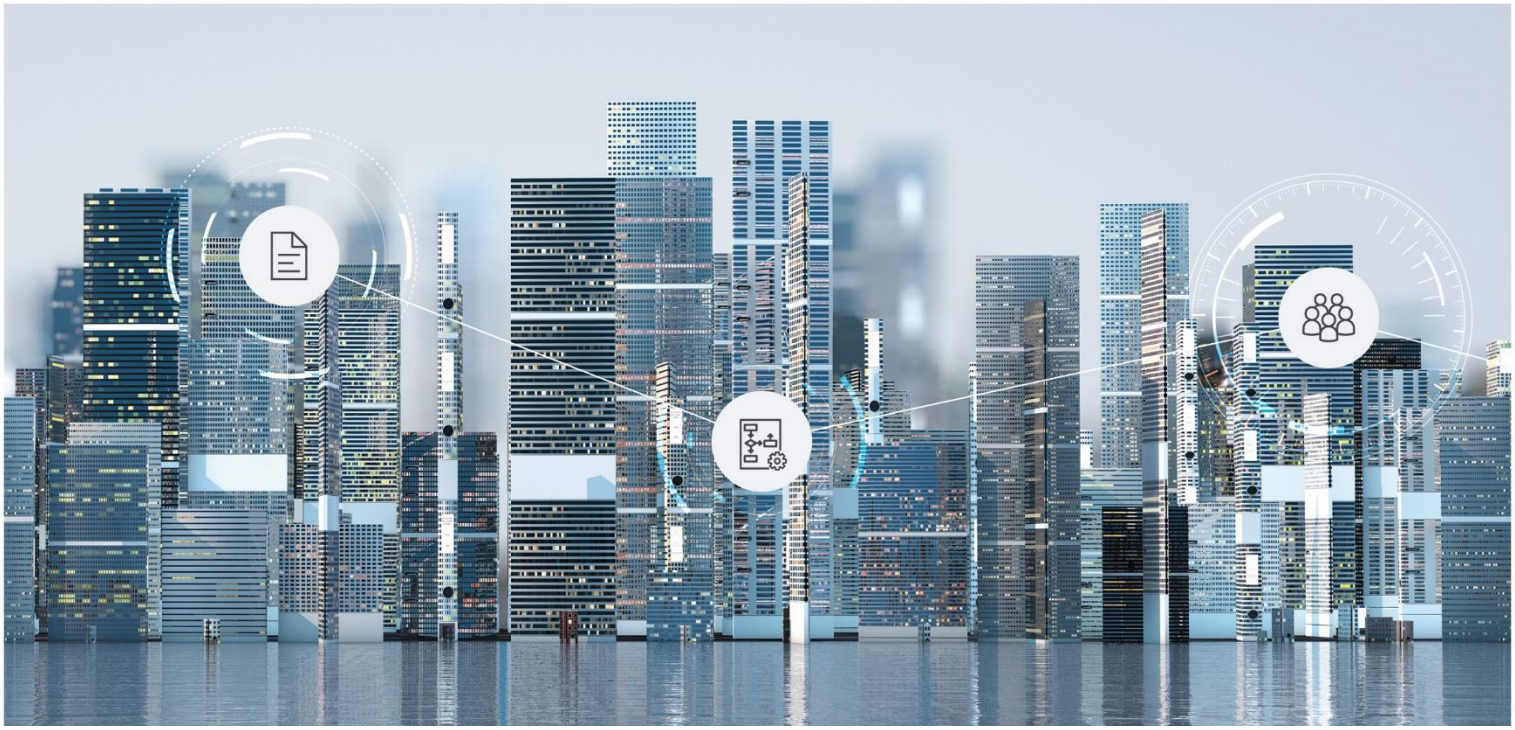
Fabasoft



Administration Help

Fabasoft Folio

Contents

1 Introduction

This document describes administration and configuration tasks in a Fabasoft Folio Domain.

2 Domain Administration

The “Domain Administration” contains the administration and configuration objects of a Fabasoft Folio Domain. When installing a Fabasoft Folio Domain a domain administration is created by default. It is recommended to use this domain administration but you can also create your own (all domain administrations reference the same administration and configuration objects).

Note: Object lists in the domain administration are not refresh automatically for performance reasons. To view the current entries, click “Refresh” in the background context menu,

A domain administration provides following substructures:

- *Administration*
  Provides an administrator-friendly view on the most important user-related objects that can also be found in *User Objects*.
- *Domain Objects*
  Contains all domain-related objects like licenses or services.
- *User Objects*
  Contains all user-related objects like users or user environments.
- *Class Browser*
  All objects in Fabasoft Folio belong to an object class. The class browser shows the object class hierarchy for your information but no administration tasks must be carried out.
- *Configuration Objects*
  Provides an overview of all configurations. More information about the configuration evaluation order can be found in chapter 5.2 “Configuration Evaluation Order”.

3 Common Administration Tasks

The following chapters describe common administration tasks in Fabasoft Folio.

3.1 Managing Users

In order that a user can log on, an own user object and a user environment must exist for the user.

Users are managed here: *Domain Administration* > *User Objects* > *Users*.

Creating a User

When creating a *User* enter at least the surname and the login name.

Editing a User

Use the “Properties” context menu command to edit the user's metadata.

“User” tab

- *Photo*
  A photo of the user.

- *Name and Address*
  Shows the name and address of the user.
- *Active*
  Only users marked as active can log on to the Fabasoft Folio domain. Deactivate any user no longer required.
  Caution: Do not delete user objects that have already been used, but instead deactivate them to avoid inconsistencies. For example, in all Fabasoft Folio objects, the field *Created by* refers to a user object. If this user object is deleted, the shortcut can no longer be resolved.
- *Login Name*
  Defines the login names with which the user can log on (one login name per line). For example, enter the Microsoft Windows domain user name (e.g. `domain1\username`). The user name has to be unique and lower case only.
  Note: To define a login name, which only exists locally on a device, add the prefix `:name_of_local_computer:\` to the login name.
- *First Name/Surname/Middle Initial/Title/Post Title*
  These fields define the names and titles of the user.
  Note: The *Surname* field is mandatory.
- *Abbreviation*
  Defines an abbreviation for the user.
- *Sex*
  Defines the sex of the user.
- *Date of Birth*
  Defines the date of birth of the user.
- *Salutation*
  Defines the salutation that may be used in e-mails.
- *Substitutions*
  Defines the user substitution object (see chapter 3.3 “Managing User Substitutions”).
- *Tenants*
  A Fabasoft Folio Domain can consist of several tenants. Tenants for which roles are defined are automatically entered in this field.
- *Roles per Tenant*
  Assigns roles to the user. Roles are commonly used for defining access rights or workflow participants.
  - *Default*
    If the user does not change the role manually, the user works in the default role. If more than one role is defined as default, the first listed default role will be used.
  - *Position*
    Defines the user's positions.
    Example: Manager
  - *Group*
    Each position is defined for a group.
    Example: Marketing
  - *Tenant*
    Defines in which tenant the role can be used.

    - *User Environment*
      Defines which user environment should be used when the user switches to this role.
    - *Function*
      Defines a short description of the role.
    - *Dispatch Clause*
      Defines the text to be attached to a document signed by the user.
- *Member of*
  Defines the groups to which the user belongs. Groups for which roles are defined are automatically entered in this field.
- *Security Clearance*
  Defines the security clearance of the user. For example, Teamrooms can be secured by security clearances.
- *Organization (Text)*
  Defines the organization of the user as free-text.
- *Function in Organization*
  Defines the function of the user as free-text.

"Address" tab

- *Addresses*
  Defines the addresses of the user.
- *Telephone Numbers*
  Defines the telephone numbers of the user.
- *E-Mail Addresses*
  Defines the e-mail addresses of the user.
- *Web Sites*
  Defines the web sites of the user.
- *Language for Communication*
  Defines the language that should be used for communication with the user.

"Environments" tab

- *Environments*
  Defines the user environments of the user and which one should be used as default.
- *Default Environment Template*
  Defines an environment template. The user's first login triggers a copy of this environment template. This copy is entered in the *Environments* field.
- *Environment Template per Tenant*
  Defines tenant-specific user environment templates. Users working in multiple Fabasoft Folio tenants will have a separate user environment for each specific Fabasoft Folio tenant.
    - *Environment Template*
      Defines the user environment to be used as a template for a tenant-specific environment.
    - *Tenant*
      Defines the Fabasoft Folio tenant for which the template is to be used.
- *Automatically Create Local Environment*
  Selecting "Yes" means that, if a user is not working in the default tenant, a local user environment will be created in the corresponding tenant. This means that the desktop can be generated more quickly than when it is loaded from the user's default tenant. The template for

this user environment is stored in the *Default Environment Template* field. If no template has been defined, the template of the *Group* will be used. If no template is defined in the group as well, it is not possible to create a user environment.

- *Automatically Create Environment in Tenant*
  Selecting "Yes" means that a user environment will be created automatically for the user in the tenant where the user is logged on. The following hierarchy applies for choosing the template:
  1. *Environment Template per Tenant* in the user object
  2. *Environment Template per Tenant* in the group object
  3. *Default Environment Template* in the user object
  4. *Default Environment Template* in the group object
- *Template Collections*
  Defines template collections for the user. The user can use templates from the template collection when creating objects.

Note: If both the *Automatically Create Local Environment* and the *Automatically Create Environment in Tenant* fields are activated, the former field will be ignored.

"Apps" tab

- *Apps Changed on/at*
  Is used to define whether the license cache must be updated.
- *Licensed Apps*
  Defines the apps licensed to the user.

"Collaboration" tab

Shows information about Teamrooms, invitations and apps.

"Workflow" tab

Shows information about workflow tasks of the user.

"Authentication" tab

- *Used Products*
  Shows the software products used by the user.
- *Used Devices*
  Shows the *Devices* where the user has already worked.
- *Authorized Devices*
  Defines devices where the user can log in. The user will not be able to log in on other devices, as identified by the *Device Identification* field of the device object.
  Note: A device object will be created automatically for each device where the user has worked at least once. A *Device* is defined by at least two values:
  - *Device Type*
    Defines the system environment to be used on the device.
  - *Device Identification*
    Defines the device's MAC address.

"Advanced" tab

- *Must Use Role*
  Defines whether a user must always work in a role. If the user does not work in any role, the rights of all groups entered in the *Member of* field will apply.

- *Check License Locally*
  Defines whether the license is always checked locally for this user. This means that the user can log in even if cross-domain license verification is not available. A valid local license must be available for this to happen. If you select "Yes" in this field, you must enter at least one license in the *Cached User Licenses* field.
- *Cached User Licenses*
  Defines the licenses that have been cached for the user. The user can use the licenses entered even if cross-domain license verification is not available.

Deactivating a User

Deactivate any user object that is no longer required. Do not delete user objects that have already been used to avoid any inconsistencies. For example, in all Fabasoft Folio objects, the field *Created by* refers to a user object. If this user object is deleted, the shortcut can no longer be resolved.

1. Navigate to the user.
2. Edit the properties of the user.
3. Disable the *Active* option to deactivate the user.

3.2 Managing User Environments

In order that a user can log on, an own user object and a user environment must exist for the user. In general, you will create one or more user environments manually as templates. The template will be referenced in user objects. When the user logs on the first time, a copy of the template is created automatically and used for the user.

User environments are managed here: *Domain Administration* > *User Objects* > *User Environments*.

Creating a User Environment Manually

When creating a *User Environment* define at least the available user profiles (e.g. end user or administration user) for the user.

If the user environment is not intended as template, edit the properties of the user environment, switch to the "Security" tab and define the respective user as *Owner*.

Creating a User Environment Automatically

It is assumed that a user environment that should serve as template has already been created.

Edit the properties of the user, switch to the "Environments" tab and select a user environment in the *Default Environment Template* field. In this way, a copy of the environment template is created automatically when the user logs on the first time. The copy is entered in the *Environments* field of the user object.

Editing a User Environment's Metadata

Use the "Properties" context menu command to edit the user environment's metadata.

"General" tab

- *Show Hints*
  Defines whether tool tips should be displayed.

- *Read Properties by Default*
  Defines whether an object is opened in read or edit mode, when opening it via the "Properties" command.
- *Show Upload Confirmation*
  Defines whether a confirmation dialog is shown when uploading a file.
- *Show Exit Confirmation*
  Defines whether users are asked to confirm that they wish to exit Fabasoft Folio when they close their web browser.
- *Use Release Version by Default*
  Defines whether release versions should be shown by default.
- *Days after Which a New Version is Automatically Created*
  If an object has not been edited for the period of days defined here, a new version is automatically started the next time the object is edited.
- *Enable Domain Selection During Create*
  The user can select the Fabasoft Folio domain in which the object is to be created.
- *User Profile*
  User profiles can restrict the options of the user (menus, buttons, forms, and object classes for creating or searching).
- *Available User Profiles*
  Shows all user profiles, which are available for the user. If there are multiple user profiles available, the user can choose one of them.
- *Use the PDF Viewer*
  Defines whether the internal PDF viewer should be used.
- *Show Welcome Screen*
  Defines whether the welcome screen should be shown.

"Accessibility" tab

- *Play Acoustic Signals*
  Defines whether a signal sound is played for successful processing steps, errors and questions.
- *Include all Fields in Tab Order*
  If you are reliant on keyboard navigation, activate this option so that read-only fields are also included in the tab order.
- *Use Advanced Mode for Prescriptions per Default*
  Defines whether the tabular mode is used for prescriptions. The default displayed graphical process editor is not suitable for keyboard operation.
- *Show Alternative Text for Highlighted Fields*
  Highlighted fields are marked in color only. Activate this option to show an additional text alternative.
- *Prepare Foreign Language Expressions for Speech Output*
  If you are using speech output, enable this option to correctly pronounce selected foreign language expressions.
- *Preview in the Document View*
  Defines whether the PDF preview or only a preview image will be displayed in the document view. Choose the preview image, if you rarely use the PDF preview and if you want to benefit from a fast loading document view. Note that when choosing the PDF preview the navigation between documents using the keyboard is no longer simple possible in some web browsers

(e.g. with the left and right arrow keys) since you have to leave the PDF first. If you are reliant on keyboard navigation, it is recommended to use preview images.

"Localization" tab

- *Language*
  Defines the language of the user environment.
- *Locale*
  Defines how numbers and dates are displayed.
- *Multilingual Input*
  Defines whether multilingual input is available for multilingual names.
- *Default Currency*
  Defines the default currency. This currency will be used for amounts entered by the user.
- *Disable Currency Symbol*
  Defines whether the currency symbol for the default currency is displayed.
- *Reference Currency*
  Defines the reference currency. This can only be used in conjunction with a *Conversion Table*. A conversion table is used to convert between currencies.
- *Disable Reference Currency Symbol*
  Defines whether the currency symbol for the reference currency is displayed.

"Search" tab

- *Extended Search for Object Pointer Properties*
  Defines whether the "Search" symbol is displayed for object pointer properties. By default, it is displayed.
- *Search Defaults*
  - *Object Limit*
    Defines the maximum amount of the objects that can be found within a search.
  - *Time Limit (sec)*
    Defines how much seconds a search can require.
  - *Query Scope*
    Defines a query scope. For example, a query scope can be used to search in specific domains, COO Stores or COO Services.
  - *Show Query Text*
    Defines whether the "Edit Query" button is available when executing a search.
  - *Show Search Options*
    Defines whether the "Search Options" button is available when executing a search. This button provides further search options that only apply within the current search.
- *Search Forms*
  Defines search forms, which should be available for the user.
- *Search Results*
  Shows search results.

"E-Mail" tab

- *Use Access via Send by Default*
  Defines whether an access token should be included in sent links.
- *E-Mail Settings*

- *Use Formatted Text*
  Defines whether formatted text (HTML text) should be inserted into an e-mail when sending an object. If set to "No", links are inserted into the e-mail in plain text and not as HTML text. It is possible to insert the HTML text into the e-mail using the clipboard. This setting is only considered if Microsoft Outlook is used as default e-mail program on the client.
- *Open "E-Mail (Microsoft Outlook)" as*
  For objects of the *E-Mail (Microsoft Outlook)* object class the content is stored in the Microsoft specific MSG format and additionally in the standardized MIME format. This setting defines in which format this e-mail object should be opened. Select "MIME (Internet Standard)" if *E-Mail (Microsoft Outlook)* objects should be opened, although there is no Microsoft Outlook installed on the client.
  Note: In case the content of the e-mail is not available in MIME format, the MSG content is opened.
- *Open "E-Mail (MIME)" with Microsoft Outlook*
  Defines whether e-mails, which are stored in MIME format, should be converted into the Microsoft specific MSG format when opening them. This enables the possibility to open *E-Mail (MIME)* objects with Microsoft Outlook by default.
  If set to "No", the e-mail is not converted into the MSG format and the default program for files in MIME format is opened. Usually this is not Microsoft Outlook, but e.g. Mozilla Thunderbird.
  Note: In newer versions of Microsoft Outlook, files in MIME format can be opened directly without a conversion (see Microsoft Outlook option "/eml"). If Microsoft Outlook is defined as default for opening files in MIME format on the client, the conversion is not necessary. This setting is only considered if Microsoft Outlook is used as default e-mail program on the client.

"Workflow" tab

- *Activities for Current Role Only*
  By default a user receives all activities, which regard him personally or one of his user roles. This field defines whether only activities for the user's current role are shown in addition to the personal activities.
- *Define Deadlines as Timespan in Days (Instead of a Date)*
  Defines whether deadlines should be defined as timespan or as date.
- *Use Advanced Mode for Participants per Default*
  Defines whether the advanced view is used by default for the participants within a prescription.
- *Show News About New Activities on the Welcome Screen*
  Defines whether new activities are shown in the welcome screen.
- *Automatically Open the Next Activity After Finishing an Activity*
  Defines whether the next activity should be opened automatically when finishing an activity.
- *Period for Statistics*
  Defines the period for the workflow statistic.
- *Workflow Preferences*
  The selected object influences various fields in the workflow, like user, organizational units and distribution lists when forwarding.
- *Workflow Notifications*
  Defines how workflow notifications are carried out.

"Application" tab

- *Local RSS Feeds*
  Defines objects, which should be available as RSS feeds for the user.
- *Favorite Folder (Tasks)*
  Defines the user's favorite folder.
- *User Calendar List*
  Defines the calendar list, which is used for the Fabasoft Integration for CalDAV.
- *Address Book List*
  Defines the address book list for the Fabasoft Integration for CardDAV.

"User Interface" tab

- *Show Search Field*
  Defines whether the search field should be shown.
- *Simple Mode*
  Defines the devices for which the simple mode should be used.
- *Open Documents Read-Only by Default*
  Defines whether an object is opened in read or edit mode by default.
- *Upper Limits For "Most Recently Used"*
  - *Object Classes*
    Defines the maximum number of object classes that will be shown when creating objects, for instance.
  - *Objects*
    Defines the maximum number of objects available, for instance, in dropdown lists.
- *Custom Task Panes, Toolbars and Menus*
  Defines an object of the object class *User Interface Scoping Rule*.
- *Slideshow Interval (in Seconds)*
  Defines after how many seconds a new picture is displayed within a slideshow.
- *Show Tab Icons*
  Defines whether symbols are displayed on tabs additional to the labels.
- *Display Action Texts in Short Form*
  Defines whether only short tips are displayed in the task pane, or a long description is displayed.
- *Show Only Symbols in Portal Page Selection*
  Defines whether only symbols are displayed in the portal header. For example, for portal pages only the symbol is displayed or the symbol and the name of the portal page is displayed.
- *Font Size*
  Defines the font size ("small", "medium" or "large"). By default, the font is displayed small.

"Advanced" tab

- *Login Name*
  Defines the name of the user environment.
- *Class of Desk Object*
  Defines the type of desktop for the user.
- *Desk Object*
  Defines the desk of the user.

Note: If this field is left blank, an object of the object class defined in the *Class of Desk Object* will be entered automatically.

- *Default Dispatcher*
  Defines the used dispatcher.
- *Fixed Theme*
  Defines the used theme.
- *Portal*
  Defines the used portal.
- *Personalized Portal Parts*
  If users add their own portal pages to their user environment (e.g. static URLs), these will be listed here.
- *Number of Objects in Kernel Cache*
  For internal use.
- *Events*
  The object *User Events* stores a list of all events having taken place and information on the form in which the events are to be communicated to the user.
- *Favorites*
  Defines the favorites used in the create dialog.
- *Template Collections*
  The user can use templates from the template collection when creating objects.
- *Print Configuration*
  - *Printers*
    Defines the used printers.
  - *Papers*
    Defines the used papers.
- *Display Support Button*
  Defines whether a feedback button is displayed next to the "Help" symbol.
- *Enable Fabasoft app.telemetry on Clients*
  Defines whether the support button is available.
- *Enable Folio Network Drive*
  Defines whether the Fabasoft Folio Client provides the “Open Folio Network Drive” command.
- *Enable Open Calendar Menu*
  Defines whether the “Open Calendar” command is shown.
- *Enable Open Address Book Menu*
  Defines whether the “Open Address Book” command is shown.
- *Synchronization Mode*
  Defines how the “Folio Folder” can be used.
- *Settings for Test Dispatch*
  Defines the recipients for a test dispatch.
- *Attachments from User*
  For internal use.

“Apps” tab

The “Apps” tab provide information about licensed apps.

“Commonly Used” tab

The “Commonly Used” tab contains all objects that the user used last.

“SAP” tab

- *SAP Connection Information*
  Defines the SAP connection information.

“Contact Synchronization” tab

- *Contact Folder*
  Defines the contact folder for contact synchronization.

“User Environment (Advanced)” tab

- *Current Document to Store*
  For internal use.
- *Show Error Message in Status Bar*
  Defines whether an error message is displayed in the status bar.
- *Available Departments*
  Defines the available departments.
- *Stamps*
  Defines PDF stamps.
- *Default Color for Annotations*
  Defines the default color for PDF annotations.
- *Settings for Branches*
  Defines whether specific branches such as buttons and links should be displayed in specific application views.
- *Show Menu Bar*
  Defines whether the menu bar is shown.
- *Office Documents*
  Defines how Microsoft Office documents are opened (on the device in Office Online).
- *Recovery Information*
  In this field all objects, which could not be recovered automatically, are listed. In the *Reason* column, the cause is displayed (the appropriate object could be locked by another user). Click "Show File in Folder" to execute further steps, for example importing the file manually.
- *Cached Contents*
  For internal use.
- *Creation Mode Contracts*
  Defines how contracts are generated.
- *Announcements*
  Shows announcements of the user.

3.3 Managing User Substitutions

To enable a user substitution for a user, add a *User Substitution* object on the “User Substitutions” tab of the user object.

User substitutions are managed here: *Domain Administration* > *User Objects* > *User Substitutions*.

Creating a User Substitution

When creating a *User Substitution* enter at least a name.

Editing a User Substitution's Metadata

Use the “Properties” context menu command to edit the user substitution’s metadata.

“User Substitution” tab

- *User*
  Shows the user for whom this user substitution was created. Assign the user by entering the user substitution in the user object on the “User Substitutions” tab.
- *Roles With Substitute*
  Defines the substitution.
  - *Position*
    Defines the position of the role.
  - *Group*
    Defines the group of the role.
  - *Substitute*
    Defines the user’s substitute of the role.
  - *Start of Substitution*
    Defines the date when the substitution is to start for this role.
  - *End of Substitution*
    Defines the date when the substitution is to end for this role.
  - *Personal Substitution*
    Defines whether the substitute can also process activities assigned personally to the substituted user (e.g. those not assigned to the user by his role).
  - *Remark*
    Defines a remark.
  - *Denied For Substitute*
    Defines the access types not permitted to the substitute for this role.
- *Common Substitute*
  Defines the common substitution settings. The settings are taken over to all rows in the *Roles With Substitute field.*
  - *Substitute*
    Defines the user’s substitute
  - *Start of Substitution*
    Defines the date when the substitution is to start.
  - *End of Substitution*
    Defines the date when the substitution is to end.
  - *Personal Substitution*
    Defines whether the substitute can also process activities assigned personally to the substituted user (e.g. those not assigned to the user by a role).
  - *Remark*
    Defines a remark.

- *Denied For Substitute*
  Defines the access types not permitted to the substitute.

If you only specify a start date and no end date, the substitution will start on the date specified and end when the user ends the substitution.

If you only enter an end date and no start date, the substitution will start immediately and expire on the end date specified.

If you do not specify any of these dates, the substitution will start immediately and can only be ended when the substituted user explicitly ends the substitution.

3.4 Managing Groups

Users can be members of groups. Groups are useful, for example, to provide access rights to all members of a group.

Groups are managed here: *Domain Administration* > *User Objects* > *Groups*.

Creating a Group

When creating a *Group* enter at least a short name. The group name will be generated automatically and has the format `<short name> (<name>)`.

Editing a Group's Metadata

Use the “Properties” context menu command to edit the group’s metadata.

“Group” tab

- *Active*
  The group's default status is active.
  Note: Deactivate any group object no longer required. Deleting a group object can cause inconsistencies to arise in the system.
- *Short Name*
  Defines an abbreviation for the group.
- *Description*
  Defines the group's name.
- *External ID*
  Defines the external ID.
- *Organizational Unit Type*
  Defines *Organizational Unit Types* to which the group belongs.
- *Subgroups*
  Defines groups subordinate to this group.
  Note: This field automatically contains those *Groups* subordinate to the current group.
- *Supergroups*
  Defines groups superordinate to this group.
  Note: This field automatically contains those *Groups* superordinate to the current group.
- *Members*
  Defines all members of the group.

“Address” tab

- *Addresses*
  Defines addresses of the group.
- *Telephone Numbers*
  Defines telephone numbers of the group.
- *E-Mail Addresses*
  Defines e-mail addresses of the group.

"User Settings" tab

- *Login Name*
  Defines the name of the group.
- *Object Class for Created Users*
  Defines the object class (e.g. *User*) to be used as a template for the automatic creation of a user object.
- *Send Teamroom Invitations to Members*
  Defines whether Tearoom invitations are sent if the group is added to a Teamroom.
- *Default Position Template*
  Defines the position that will serve as a template
- *Default Environment Template*
  Defines the user environment that will serve as a template, if a user environment is to be automatically created for a group member.
  Note: If a user environment also exists in the *Default Environment Template* field of the user object, two environments will be created. The environment created from the user object template will automatically be the default environment.
- *Environment Template per Tenant*
  Defines tenant-specific user environment templates.

"Substitutions" tab

- *Group Substitutions*
  Defines substitutions of the whole group. During the substitution period, members of the substitution group become members of the substituted group. The rights of the substitution group can be restricted.
- *Substituted Groups*
  Shows the substituted groups.

"Apps" tab

Shows information about licensed apps.

"Workflow" tab

Defines workflow settings.

"Access Permissions" tab

- *Default ACL for New Objects*
  Defines the ACL for new objects.
- *ACL Objects*
  If ACL objects are entered in this field, no other predefined ACL objects from the "Security" tab will be available for selection for objects of this group.

- *ACL Assignments*
  Defines which group will be entered as an object's *Group* in the case of an ACL change to the object. The choice of group is governed by the assignment of groups to ACLs.

"Advanced" tab

- *Restricted Role Administration*
  If, for example, an administrator group is not superordinate to the entire organization, but only to a certain department, members of this administrator group will only be able to assign roles to users of the respective department or its subordinate departments.
- *Distinguished Name (dn)*
  Defines the distinguished name of the group.
- *Default Column Settings*
  Defines the default column settings.
- *Template Collections*
  The user can use templates from the template collection when creating objects.
- *Available Departments*
  Defines the available departments.
- *Synchronize Mode*
  Defines how the "Folio Folder" can be used.
- *Print Configuration*
  - *Printers*
    Defines the used printers.
  - *Papers*
    Defines the used papers.
- *Portal*
  Defines a *Portal* for use when a *User Environment* is automatically created for a group member and no *Portal* has been created in the template for the user environment. It makes no difference whether the template is created in the *Group* or in the *User*.

3.5 Managing Positions

Typical positions are "Staff Member" or "Manager". A position within an organizational unit type defines a role.

Positions are managed here: *Domain Administration* > *User Objects* > *Positions*.

Creating a Position

When creating a *Position* enter at least a reference and a software component.

Editing a Position's Metadata

"Position" tab

- *Multilingual Name*
  Defines the name.
- *Template Collections*
  Defines templates for the position. Users with this position can use templates from the template collection when creating objects.

- *Activities Assigned to Position*
  Shows the assigned activities.
- *Activities with Long Term Deadlines Assigned to Position*
  Shows the assigned activities with long-term deadlines.
- *Signature Configurations*
  Defines the signature configurations of the position.

3.6 Managing Organizational Unit Types

Typical organizational unit types are “Department” or “Team”. A position within an organizational unit type defines a role.

Organizational unit types are managed here: *Domain Administration* > *User Objects* > *Organizational Unit Types*.

Creating an Organizational Unit Type

When creating an *Organizational Unit Type* enter at least a reference and a software component.

Editing an Organizational Unit Type's Metadata

“Organizational Unit Type” tab

- *Multilingual Name*
  Defines the name.
- *Positions*
  Defines the assigned positions.
- *Level*
  Defines the hierarchy level.
- *Activities Assigned to Organizational Unit Type*
  Shows the assigned activities.
- *Activities with Long Term Deadlines Assigned to Organizational Unit Type*
  Shows the assigned activities with long-term deadlines.

4 Advanced Administration Tasks

In general, advanced administration tasks should only be carried out by Fabasoft Folio specialists.

4.1 Managing Devices

Device objects are automatically created for each device used (identified by the MAC address). Device objects can be used to define authorized devices for users (*User* object > “Authentication” tab > *Authorized Devices* field).

Devices are managed here: *Domain Administration* > *User Objects* > *Devices*.

Creating a Device

When creating a *Device* enter at least a name.

Editing a Device's Metadata

Use the “Properties” context menu command to edit the device’s metadata.

“Device” tab

- *Name*
  The name of the device.
- *Device Type*
  The device's system environment.
- *Device Identification*
  The device's MAC address.
- *Frontend Server*
  Specifies whether the device acts as a frontend server (Fabasoft Folio Webserver).
- *Number of CPUs*
  The number of available processors on the device.
- *Number of CPU Cores*
  The number of CPU cores on the device.
- *Used Products*
  The Fabasoft software products used on the device.
- *Cached Device Licenses*
  Cached Software Product Licenses are used for a cross-domain license verification.

4.2 Managing ACLs

An ACL (Access Control List) is a list of Access Control Entries (ACE - access authorization). ACLs define access rights to Fabasoft Folio objects for domains, organizational units, groups and positions or users. An ACL is assigned to every Fabasoft Folio object (even ACL objects).

ACLs are managed here: *Domain Administration* > *User Objects* > *ACLs*.

Creating an ACL

When creating an *ACL* enter at least a reference and a software component.

Editing an ACL's Metadata

“ACL” tab

- *Multilingual Name*
  Defines the name.
- *ACL Entries*
  Defines the access control entries. An entry is only taken into account, if the following conditions are met:
  - *Domains*
    Defines the domain condition (e.g. “Owner Domain” or “All Domains”).
  - *Organizational Unit Types/Groups*
    Defines the organizational unit type/group condition (e.g. “All”). The symbol has following meaning:
    - ◇
      No additional condition.

-  (and superordinate groups)
  The ACL entry applies to the group or organizational unit type selected and its superordinate groups or organizational unit types.
-  (and subordinate groups)
  The ACL entry applies to the group or organizational unit selected and its subordinate groups or organizational unit types.
-  (if superordinate to object group)
  The ACL entry applies to the superordinate group or organizational unit type of the selected group or organizational unit type.
-  (if subordinate to object group)
  The ACL entry applies to the subordinate group or organizational unit type of the selected group or organizational unit type.

- *Positions/Users*
  Defines the position/user condition (e.g. “All”).
- *Access Types*
  If all conditions are met, the defined access types (e.g. Create object) are granted. If the conditions of several ACEs are met, the access types are merged.

4.3 Managing Domains

Make settings for the entire Fabasoft Folio domain in the *Current Domain*. You can also configure individual software components in the active Fabasoft Folio domain, by referencing configuration objects for the respective software component in the active Fabasoft Folio domain.

If you require domain-specific or client-specific settings, create a separate configuration object for each individual Fabasoft Folio domain or client.

Domains and tenants are managed here: *Domain Administration* > *Domain Objects* > *Domains*.

Editing a Current Domain's Metadata

“System Configuration” tab

- *Test System*
  Defines whether it is a test system.
- *Enable 2-Phase Commit*
  Set whether 2-phase commit is to be used for distributed transactions.
- *Enable Full Text Queries*
  Defines whether full text searches are to be supported. Full text queries require an index server to run on the server on which the Fabasoft Folio MMC Services are running. To enable full text queries, set this field to “Yes”. If no index server is available, the field must be set to “No”, otherwise any full text search will cause an error.
- *Enable Optimized Full Text Queries*
  The default setting is for a full text search to first search the Fabasoft Folio database. It will subsequently also search index server catalogues where this database's objects are stored, but not the other index server catalogues. Define whether the search engine is to search the index server before the Fabasoft Folio database, in the case of a pure full text search (without any further restrictions on other properties).

- *Queries are Case-Insensitive*
  Defines here whether upper/lower case is significant for local searches and server searches.
  Note: "Yes" means that the search will ignore upper/lower case.
- *Enable Queries With Starting '*'*
  Defines whether queries with a starting '*' are supported.
- *Display Owner if Read Access Is Denied*
  Defines whether the name of the owner is shown, if the access rights of a user are not sufficient.
- *Enable Quotas*
  Defines whether storage quotas can be used.
- *Archived Versions Remain in Fabasoft Folio*
  Defines whether the archive versions of archived objects and their contents will not be deleted from the Fabasoft Folio domain after archiving.
- *When Archiving Convert Content to Final Form*
  Defines whether the object's content is converted to a final format (e.g. PDF) before being archived.
- *When Versioning Convert Content to Final Form*
  "Yes" means that, on creating a new version of an object, the current version's content will be converted to a final format (e.g. PDF).
- *Format for Final Form*
  Defines the file format for finalized contents.
  Example: PDF, TIF
- *Date/Time to Generate a Status Object per Device*
  If a user logs in for the first time to a Fabasoft Folio domain after this date and no *Status Information* object exists for the user, an object of the object class *Status Information* will be created automatically. This object stores information on the user's device and environment (registry entries, operating system version etc).
- *Used Holidays and Time Intervals*
  Defines non-working days, holidays and time periods.
- *Privileged Users*
  Reference *Users* in the context of accessing Fabasoft Folio Services. The login name of these privileged users is specified during Fabasoft Folio Services installation.
- *Anonymous User*
  Defines the user for anonymous access.
- *Used Licenses*
  Lists the Fabasoft software product licenses currently in use.
- *Cache- and UDP-Multicast-Protocol*
  The cache and UDP multicast protocol optimizes the Fabasoft Folio kernel cache behavior. The goal is to reduce the RPCs (Remote Procedure Call) necessary between server and client during read access to objects, if the object already exists in the client cache. To achieve this, Fabasoft Folio COO Service sends UDP Multicast packets to inform clients of any object modifications. Configure in this field whether or not the Cache and UDP Multicast Protocol is to be used. Also make settings for the Multicast address and the Multicast port.
  - *Enabled*
    Enable and disable the protocol.

    - *Multicast Address*
      Defines the multicast address.
    - *Port*
      Defines the multicast port.
- *Custom Task Panes, Toolbars and Menus*
  Defines a User Interface Scoping Rule. A Fabasoft DUCX Expression for calculating the applicable validity area is provided in this field (*Expression To Compute The Current Scope* field). This validity area enables menus, task interfaces and toolbars to be configured, e.g. as role-specific.
- *Global Wastebasket*
  Deleted objects are stored in the global wastebasket. As default setting, the *Global Wastebasket* is managed by administrators. The global wastebasket contains a separate object for each user and date, where the destroyed objects are located. The administrators are authorized to restore or permanently destroy objects in the global wastebasket.
- *Template Collections*
  Defines templates (objects in *Template Collections*) that are available across domains. Templates are available for creating objects in the *Template Categories* area.
- *Print Configuration*
    - *Printers*
      Defines the used printers.
    - *Papers*
      Defines the used papers.

"Components Configuration" tab

Defines the used configurations.

"Placement" tab

Defines in which COO store the objects of the defined object classes are created. Object classes not configured are distributed to all COO stores. If a COO store that belongs to another tenant should be used, the *Allow All Tenants* option must be selected.

Note: Placement configuration changes and new COO stores work immediately without the need to restart all Fabasoft Folio Kernel instances (neither the current nor others).

"Upgrade" tab

Defines information for updating a domain.

"Domain" tab

Note: The Domain ID identifies the domain. The Domain ID is a unique label for each individual Fabasoft Folio domain. It comprises two numbers, the Major and Minor Domain ID, separated by a dot (e.g. 10.110).

- *Major Domain ID*
  Shows the Major Domain ID of the Fabasoft Folio domain. It is fixed by the software product license on installation of the Fabasoft Folio domain.
- *Minor Domain ID*
  Shows the Minor Domain ID of the Fabasoft Folio domain. It is fixed by the software product license on installation of the Fabasoft Folio domain.

4.4 Managing Licenses

A *Software Product License* contains information on Fabasoft *Software Product Suites* and Fabasoft *Software Products* that may be used with this license. License objects cannot be edited.

Licenses are managed here: *Domain Administration* > *Domain Objects* > *Licenses*.

Editing a License's Metadata

"Software Product License" tab

- *Name*
  Shows the license name.
- *Software Product Suites*
  Shows the licensed software product suites.
- *Software Products*
  Shows the licensed software products.
- *Contract ID*
  Shows the number of the contract on which this software product license is based.
- *Customer Name*
  Shows the name of the organization for which this software product license was issued.
- *Person Responsible*
  Shows the name of the customer employee responsible for Fabasoft software products.
- *Major Domain ID*
  Shows the Major Domain ID assigned to the customer by the software product license.
- *Minor Domain ID*
  Shows the Minor Domain ID assigned to the customer by the software product license.
- *Additional Domain IDs*
  Shows further Domain IDs that the customer can use with this software product license.
- *Minimum Minor Domain ID for Tenant*
  Shows the lowest number for the Minor Domain ID that can be used for additional Fabasoft Folio clients.
- *Maximum Minor Domain ID for Tenant*
  Shows the highest number for the Minor Domain ID that can be used for additional Fabasoft Folio clients.
- *Type of Contract*
  Shows the type of contract on which the software product license is based.
- *Date of Contract*
  Shows the date on which the contract was signed.
- *Key Type*
  Shows the type of license key.
  - "Test key"
    This key is only used for test installations. Test keys have an expiry date and a restricted number of MAC addresses that can be used.
  - "Production key"
    This key is used for productive systems. These keys have no expiry date and no restriction on the number of MAC addresses.

- *Expiration Date*
  Shows the date when the software license expires.
- *Server Info*
  Shows descriptions of the servers that can be used within this Fabasoft Folio domain.
- *Concurrent User License Timeout (Minutes)*
  Shows the timeout for reusing the license.
- *Key*
  Shows the registration key for this software product license.

4.5 Managing Stores

There are two sorts of Fabasoft Folio Stores:

- Fabasoft Folio COO Store:
  Fabasoft Folio Component Object Stores (COO Stores) store metadata for objects. A Fabasoft Folio COO Store uses a Fabasoft Folio COO Service to store data.
- Fabasoft Folio MMC Store:
  Fabasoft Folio Multimedia Content Stores (MMC Stores) store contents for objects (documents, spreadsheets, images etc.). A Fabasoft Folio MMC Store uses a Fabasoft Folio MMC Service to store data.

Besides Fabasoft Folio Stores, Fabasoft Folio Outbound Gateways can also be managed. You can only create these in Fabasoft Folio Server Management.

Use a Fabasoft Folio Outbound Gateway

- to access objects of other Fabasoft Folio domains (through communication with the Fabasoft Folio Inbound Gateways of the foreign Fabasoft Folio domains) and
- to store component objects not created in the local Fabasoft Folio domain. Component objects are objects created when installing software components.

The Fabasoft Folio Outbound Gateways are also divided into Fabasoft Folio COO Outbound Gateways and Fabasoft Folio MMC Outbound Gateways. Fabasoft Folio COO Outbound Gateways support communication with other Fabasoft Folio domains. The contents of the replicated component objects are stored in Fabasoft Folio MMC Stores.

Stores are managed here: *Domain Administration* > *Domain Objects* > *Stores*.

4.5.1 Managing COO Stores

Fabasoft Folio Component Object Stores (COO Stores) store metadata for objects.

Creating a COO Store

When creating a *COO Store* enter at least a name.

Editing a COO Store's Metadata

"COO Store" tab

- *COO Store Active*
  Defines the state of the Fabasoft Folio COO Store. Do not delete any *COO Store* object.

- *COO Store Number*
  This number is automatically assigned to the store being created. The first Fabasoft Folio COO Store is assigned the number "1". The number cannot be changed.
- *COO Service*
  Each Fabasoft Folio COO Store requires a Fabasoft Folio COO Service. Fabasoft Folio COO Services are generated when setting up a Fabasoft Folio domain or in the Fabasoft Folio Server Management. Manage Fabasoft Folio COO Services in the Fabasoft Folio Server Management.
- *Indexing Service*
  Defines any indexation service configured. The Fabasoft Folio COO Store is indexed according to the settings of the indexation service.
- *Default MMC Store*
  Assigns a Fabasoft Folio MMC Service to the Fabasoft Folio COO Store.
- *Maximum Number of Versions Kept*
  Defines the largest number of versions that an object can have. The oldest version will be deleted if this number is exceeded.
- *Days After Which Older Versions Are Automatically Destroyed*
  Defines how long an older version is to be stored. On this time expiring, the version will be deleted.

4.5.2 Managing MMC Stores

Fabasoft Folio Multimedia Content Stores (MMC Stores) store contents.

Creating a MMC Store

When creating a *MMC Store* enter at least a name.

Editing a MMC Store's Metadata

"MMC Store" tab

- *MMC Store active*
  Defines the state of the Fabasoft Folio MMC Store. Do not delete any *MMC Store* object.
- *MMC Store Number*
  This number is automatically assigned to the Fabasoft Folio MMC Store being created. The first Fabasoft Folio MMC Store is assigned the number "1". The number cannot be changed.
- *MMC Service*
  Fabasoft Folio MMC Services are created when setting up a Fabasoft Folio domain or in the Fabasoft Folio Server Management. Manage Fabasoft Folio MMC Services in the Fabasoft Folio Server Management.
- *Archive Configuration*
  - *Object Class*
    Defines the object classes the objects of which are to be archived.
  - *Archive Store*
    Defines in which *Archive Store* the objects are to be archived.
  - *Alternative Archive Store*
    Defines in which *Archive Store* objects are to be archived if the main archive store is not available.

- *Archive Store for Versions*
  Defines in which *Archive Store* older version of objects are to be archived.
- *Alternate Archive Store for Versions*
  Defines in which *Archive Store* older versions of objects are to be archived if the main archive store for object versions is not available.
- *Retention Period in Years*
  Defines the minimum length of time for which the objects are to remain archived.
- *Retention Period for Versions in Years*
  Defines the minimum length of time for which the older versions of objects are to remain archived.

4.5.3 Managing COO Outbound Gateways

Fabasoft Folio COO Outbound Gateways support communication with other Fabasoft Folio domains.

Editing a COO Outbound Gateway's Metadata

"COO Outbound Gateway" tab

- *COO Outbound Gateway Active*
  Defines the state of the Fabasoft Folio COO Outbound Gateway. Do not delete any *COO Outbound Gateway* objects.
- *COO Outbound Gateway Number*
  This number is automatically assigned to the Fabasoft Folio COO Outbound Gateway being created. The first Fabasoft Folio COO Outbound Gateway is assigned the number "1".
- *COO Service*
  The value in this field cannot be changed. A Fabasoft Folio COO Outbound Gateway is assigned to a Fabasoft Folio COO Service. The assignment takes place when installing the Fabasoft Folio domain or creating a Fabasoft Folio COO Service in the Fabasoft Folio Server Management.
- *Default Gateway*
  Defines whether to use the Fabasoft Folio COO Outbound Gateway as default gateway. To use this Fabasoft Folio COO Outbound Gateway as default gateway set this field to "Yes".
- *Domains*
  In this object list you can assign the Fabasoft Folio COO Outbound Gateway to different Fabasoft Folio domains.
- *Default MMC Store*
  Assigns a COO Outbound Gateway to the Fabasoft Folio MMC Service.
- *Maximum Number of Versions Kept*
  Defines the largest number of versions that an object can have. The oldest version will be deleted if this number is exceeded.
- *Days After Which Older Versions Are Automatically Destroyed*
  Defines how long an older version is to be stored. On this time expiring, the version will be deleted.

4.6 Managing Services

It is not possible to create objects of object classes *COO Service* or *MMC Service* at the desktop. These object classes are abstract. Service objects are created automatically when using the Fabasoft Folio Server Management to create a Fabasoft Folio Service.

Services are managed here: *Domain Administration* > *Domain Objects* > *Services*.

4.6.1 Managing COO Services

Fabasoft Folio COO Service are used to store metadata.

Editing a COO Service's Metadata

"COO Service" tab

- *Primary COO Service*
  The Fabasoft Folio COO Service with instance number "1" is always the primary Fabasoft Folio COO Service. You can view the instance number in the Fabasoft Folio Server Management. This field cannot be edited. The network addresses of the other Fabasoft Folio Services are obtained from the primary Fabasoft Folio COO Service.
  Note: All Fabasoft Folio Service objects are stored in the primary Fabasoft Folio COO Service.
- *Database Name*
  Defines the identifier the Fabasoft Folio COO Service uses to identify the database.
  - Microsoft SQL Server: Name of database
  - Oracle: Oracle SID / TNS Name
- *Database is Read Only*
  Defines whether the Fabasoft Folio COO Service can write to the database. For Fabasoft Folio COO Service to write to the database, set this field to "No".
- *Object Cache Parameter*
  - *Maximum Cache Size (MB)*
    Defines the maximum cache size in megabyte. The default value is 5000.
- *Additional Database Parameters*
  Defines additional parameters for the Fabasoft Folio COO Service. These parameters can be used, for example, for debugging purposes.
  - *Parameter Name*
  - *Parameter Value*

"Information" tab

- *COO Service Information*
  - *Number of Objects*
    Contains the number of objects stored in the Fabasoft Folio COO Stores that are served by this Fabasoft Folio COO Service.
  - *Number of Object Versions*
    Contains the number of all object versions stored in the Fabasoft Folio COO Stores that are served by this Fabasoft Folio COO Service.
  - *Object IDs in COO Stores*
    This list contains all Fabasoft Folio COO Stores that use this Fabasoft Folio COO Service.

        - *COO Store*
          Contains the Fabasoft Folio COO Store.
        - *Next Object ID*
          Shows which object address will be assigned to the next Fabasoft Folio object created in this Fabasoft Folio COO Store.
        - *Available Object IDs*
          Shows how many object addresses are available in this Fabasoft Folio COO Store. IDs of destroyed Fabasoft Folio objects may not be reused.
    - *Used in Bootstrap Query*
      Shows whether component objects are queried again on starting the Fabasoft Folio COO Service. The value in this field is defined by the Fabasoft Folio Kernel and cannot be changed by users.
    - *Stored Object Classes*
      Shows the object classes of the objects stored in Fabasoft Folio COO Stores and served by this Fabasoft Folio COO Service.
- *Object Cache Information*
    - *Maximum Cache Size in MB*
      Shows the maximum cache size for the Fabasoft Folio COO Service.
    - *Current Cache Size in MB*
      Shows the current cache size for the Fabasoft Folio COO Service.
    - *Number of Currently Cached Objects*
      Shows how many Fabasoft Folio objects are currently cached in this Fabasoft Folio COO Service.
    - *Number of Hits*
      Shows how often Fabasoft Folio objects have been found in and loaded from the cache.
    - *Number of Misses*
      Shows how often Fabasoft Folio objects were not found in the cache.
    - *Object Refreshes*
      Shows how often Fabasoft Folio objects in the cache were refreshed. The figure is calculated from the point-in-time of the last emptying of the cache and its subsequent default stocking. The cache is refreshed each time that Fabasoft Folio COO Services is started.
- *Database Information*
    - *Size of Database in KB*
      Shows the kilobytes of memory taken up by the database assigned to this Fabasoft Folio COO Service. This value can be larger than the memory actually needed by the Fabasoft Folio objects, since the size is defined when the database is created.
    - *Reserved Space in Database in KB*
      Shows in kilobytes how much memory is reserved in the database assigned to this Fabasoft Folio COO Service. This reserved memory is not used for database entries.
    - *Used Space in Database in KB*
      Shows in kilobytes how much memory in the database assigned to this Fabasoft Folio COO Service is actually used by database entries.
    - *Last Update of Statistics*
      Shows when the database statistics were last updated. If the dates of last updates differ from table to table within the database, this field will display the earliest date.

- *Last Backup*
  Shows when the last database backup was made.
- *Database Version*
  Shows the database version currently in use.
- *Database Server Instance*
  Shows the name of the server on which the database is located.
- *Database Name*
  Shows the name of the database assigned to the Fabasoft Folio COO Service.

- *Locking Information*
  - *Locked Objects*
    Shows how many Fabasoft Folio objects of this Fabasoft Folio COO Service are currently blocked. For example, an object is blocked when its metadata is being edited. The block will be lifted when
    - metadata editing is finished, or the attribute editor is closed.
    - 8 minutes have passed since blocking, if the attribute editor has not been correctly closed.
    - an administrator removes the entry from the database.
  - *Maximum Number of Locked Objects so Far*
    Shows the maximum number of Fabasoft Folio objects that have been blocked at any one time.
- *Pending Configuration Updates*
  Shows whether synchronization of information on object classes and communication settings for Fabasoft Folio Gateways between Fabasoft Folio Services is pending. By default, synchronization takes place immediately on starting the Fabasoft Folio COO Service. Subsequently synchronization is conducted every 60 minutes.
  Creating new object classes (e.g. when installing Fabasoft software components) also requires synchronization. Installing a Fabasoft software component creates a number of identical synchronization tasks. A timeout of 60 seconds is defined to avoid processing identical tasks simultaneously. This timeout will be restarted each time that a pre-existing task is created. This means that synchronization is implemented 60 seconds subsequent to the last task being created.

"Tables" tab

- *Service Definitions*
  Assigns *Service Table Definitions* to the Fabasoft Folio COO Service. A *Service Table Definition* defines which properties of an object class or a composite property are to be stored in a separate table in this Fabasoft Folio COO Service.
- *Database Table Definitions*
  Contains a list of relationalized tables for this Fabasoft Folio COO Service. These tables were created during installation of the Fabasoft Folio domain. No *Service Table Definition* exists for these tables.
  - *Compound Type/Object Class*
    Shows the composite type (or the object class) for which a separate table exists.
  - *Table Name*
    Suggest a name for the table. The character string contained in this field will be automatically modified where necessary.

        - Table names have the prefix “fsc”.
        - The name will be checked for conformity with the syntactic rules of the relevant database system. Special characters are substituted by “_”.
        - No SQL key words can be used.
    - *Defined Table State*
      Shows whether the table is activated and therefore in use.
    - *Table Properties*
      Shows which properties are stored in the table.
    - *Used by Properties*
      Composite types can be used by different properties. This list references properties with data to be stored in the table of the composite type. If no composite properties are listed here, all properties using the composite type will be added automatically on storing.
- *Class Hierarchy as Known by the Service*
  Shows all object classes together with the respective basis class from which objects in this Fabasoft Folio COO Service can be instanced and stored.

“Service” tab

- *Service Active*
  Defines the state of the Fabasoft Folio COO Service. Do not destroy any Fabasoft Folio COO Service.
- *Instance Number*
  Shows the instance number of the Fabasoft Folio COO Service. The value is inserted automatically. The Fabasoft Folio COO Service with instance number “1” is always the primary Fabasoft Folio COO Service.
- *Version Number*
  Shows the version number of the Fabasoft Folio backend services.
- *Number of Server Threads*
  Defines in how many server threads the Fabasoft Folio COO Service is to run. The larger the number of server threads, the more powerful the Fabasoft Folio COO Service. Too many server threads, however, will overload the server, thus reducing performance.
- *Server Name*
  Defines the name of the server on which the Fabasoft Folio COO Service is to run.
- *Listening Port of Service*
  Defines the port number of the Fabasoft Folio backend server where the Fabasoft Folio Kernel is to establish a connection to this Fabasoft Folio COO Service.
- *Network Addresses*
  Defines the server name or IP addresses and transmission protocols where the Fabasoft Folio Kernel or the Fabasoft Folio COO Service can be reached.
- *Last Recovery of Service*
  Shows when the Fabasoft Folio COO Service was last recreated.
- *Service Information*
  Shows information on the current connection and operating status of the Fabasoft Folio COO Service.
    - *Mode*
      Shows the operating status of the Fabasoft Folio COO Service. “Normal operation” signifies that the Fabasoft Folio COO Service is available for queries. “COO Service <Name of Service>

is not available" signifies that the Fabasoft Folio COO Service is not available (e.g. when the service is stopped). This value will be displayed in each field that takes its value directly from the Fabasoft Folio COO Service (e.g. fields on the "Information" tab).

- *Server Information*
  - *CPU*
    Shows the processor architecture of the server on which this Fabasoft Folio COO Service runs.
  - *Operating System*
    Shows the operating system of the server on which this Fabasoft Folio COO Service runs.
  - *Database System*
    Shows the database system of the server on which this Fabasoft Folio COO Service runs.
  - *Hardware ID of Server*
    Shows the MAC address of the network card for the server on which this Fabasoft Folio COO Service runs.
  - *Number of CPUs*
    Shows how many CPUs are on the server that runs this Fabasoft Folio COO Service.
  - *Physical Memory in KB*
    Shows (in kilobytes) the total physical main memory of the server on which this Fabasoft Folio COO Service runs.
  - *Physical Memory Used in KB*
    Shows (in kilobytes) the total physical main memory of the server on which this Fabasoft Folio COO Service runs.
  - *Virtual Memory of Process in KB*
    Shows (in kilobytes) the virtual memory available on the Fabasoft Folio COO Service.
- *Additional Parameters*
  Defines additional parameters for the Fabasoft Folio COO Service. These parameters can be used, for example, for debugging purposes.

4.6.2 Managing MMC Services

Fabasoft Folio MMC Service are used to store contents.

Editing a MMC Service's Metadata

"MMC Service" tab

- *Transaction Log*
  Defines the directory for storing the log files for the 2-Phase Commit. If you change the value in this field, you must restart the Fabasoft Folio MMC Service.
- *MMC Service Areas*
  Defines in the MMC areas the physical memory for the contents of multimedia objects. There must always be at least one writable Fabasoft Folio MMC area. Define in the *Change Mode* field whether a Fabasoft Folio MMC area is writable.
  - *Name*
    Defines the name of the Fabasoft Folio MMC area. You are free to choose any name.
  - *Start Point (UTC)*
    Defines the date from which the Fabasoft Folio MMC area can be used.

- *End Point (UTC)*
  Defines the date up to which the Fabasoft Folio MMC area can be used.
  Note: There must be no gap between the *End Point* of the "old" Fabasoft Folio MMC area and the *Start Point* of a "new" Fabasoft Folio MMC area.
- *Type*
  Defines how the folder structure should be constructed. The following options are available:
  - "One directory per day (Change date)"
    Storage follows the change date of the object, i.e. the file is stored in the directory where it was last changed. This storage structure is suitable for logging Fabasoft Folio MMC areas. Files are located in the directory <Directory on the server>\<Domain ID of MMC Store>\<Change date>. A separate directory is created for the year and the combination of month and day. Furthermore, a daily directory of this type contains up to 256 subdirectories.
    Note: This type is only permitted when logging mode is used.
  - "Content Addressed Storage (CAS)"
    Selecting this type of storage means that identical *Contents* (`COOSYSTEM@1.1:contcontent`) of different Fabasoft Folio objects will be only stored once in the Fabasoft Folio MMC area. This saves memory. CAS calculates a hash value for each content and then checks whether it already exists in the system. Only if this hash value does not exist will the content be stored. If the contents of an existing Fabasoft Folio object stored using CAS changes, CAS will calculate and store a new hash value for the changed content. The previous content will however not be destroyed. To delete contents no longer in use, run "Clean up areas with logging" for the respective *MMC Service*. All contents with hash values no longer assigned to a Fabasoft Folio objects will be deleted from the file system.
    The types "One directory per day (version date)", "Compound files per day" and "One directory per version of object" are no longer supported for new Fabasoft Folio MMC areas, but remain available for existing Fabasoft Folio MMC areas.
    Note: You cannot change the type once files have already been stored in this Fabasoft Folio MMC area.
- *Change Mode*
  Defines which actions can be run in this Fabasoft Folio MMC area. Choose from the following options:
  - "All operations"
    Select this option for active Fabasoft Folio MMC areas. Directory contents can then be read, written and deleted.
  - "Read only"
    Only select this option for Fabasoft Folio MMC areas with contents that may no longer be changed.
  - "Read and delete only"
    Only select this option for Fabasoft Folio MMC areas that should no longer be written to (e.g. because there is insufficient memory). It will then only be possible to read and delete contents.

    Note: Fabasoft Folio MMC areas which can or should no longer be written to, must be set to "Read only" or "Read and delete only" in *Change Mode*. This change must take place one day after the *End Point (UTC)* so that no inconsistencies arise when switching to a new Fabasoft Folio MMC area.

- *Log Changes*
  Defines whether all changes to files are to be logged.
- *Directory on Server*
  Defines the directory in which multimedia contents are to be stored.
- *Backup Directory on Server*
  Defines the directory in which backups for multimedia contents are to be stored. We recommend that you do not create the backup directory on the same physical data carrier.
  Note: Creating or changing a backup directory requires subsequent synchronization of registry entries using the Fabasoft Folio Server Management.
- *Path to Directory on Server*
  Defines the UNC path to the directory of the Fabasoft Folio MMC area. When cleaning up the Fabasoft Folio MMC area the Fabasoft Folio Kernel uses this specified path instead of the local path.
- *Path to Backup Directory on Server*
  Defines the UNC path to the backup directory of the Fabasoft Folio MMC area.
- *Backup Date (UTC)*
  Shows the date of the last backup. This date is referred to when cleaning up Fabasoft Folio MMC areas. File versions (not object versions!) older than the date in this field will be removed from the Fabasoft Folio MMC area when cleaning up. Only the most recent version of the file will be kept. If a backup directory has been defined and areas are to be backed up using the Fabasoft product functionality, the *Backup Date (UTC)* will be set automatically to the corresponding date each time "Create backup for areas" is called. You can also enter the value in this field manually.
- *Cleanup Date (UTC)*
  Shows the date the last time the Fabasoft Folio MMC area was cleaned up. If logging mode is activated, each time a multimedia content is stored by an application (word processing program, table calculation etc.) a new file will be created in the Fabasoft Folio MMC area. Cleaning up removes the logged contents from the Fabasoft Folio MMC area. Only the most recent versions of the multimedia content are retained.
  Note: The object versions will not be deleted.
- *Cleaned Up Until (UTC)*
  Shows the date up to which the Fabasoft Folio MMC area has been cleaned up. In the case of automatic clean-up, the date refers to the *Backup Date (UTC)*. The most recent versions of multimedia content are not affected.

- *Information about MMC Service Areas*
  Shows statistical information on the Fabasoft Folio MMC areas.
  - *Name*
    Shows the name of the Fabasoft Folio MMC area.
  - *Device*
    Shows the drive where the Fabasoft Folio MMC area is located.
  - *Device Capacity in KB*
    Shows (in kilobytes) the overall memory of this Fabasoft Folio MMC area.
  - *Free Space in KB*
    Shows (in kilobytes) the memory available on the drive where the Fabasoft Folio MMC area is located.

- *% Free Space*
  Shows what percentage of the memory can still be used.

- *Service Active*
  Defines the state of the Fabasoft Folio MMC Service. Do not destroy any Fabasoft Folio MMC Service.
- *Instance Number*
  Shows the instance number of the Fabasoft Folio MMC Service. The value is inserted automatically.
- *Version Number*
  Shows the version number of the Fabasoft Folio backend services.
- *Number of Server Threads*
  Defines in how many server threads the Fabasoft Folio MMC Service is to run. The larger the number of server threads, the more powerful the Fabasoft Folio MMC Service. Too many server threads, however, will overload the server, thus reducing performance.
- *Server Name*
  Defines the name of the server on which the Fabasoft Folio MMC Service is to run.
- *Listening Port of Service*
  Defines the port number of the Fabasoft Folio backend server where the Fabasoft Folio Kernel is to establish a connection to this Fabasoft Folio MMC Service.
- *Network Addresses*
  Defines the server name or IP addresses and transmission protocols where the Fabasoft Folio Kernel or the Fabasoft Folio MMC Service can be reached.
- *Last Recovery of Service*
  Shows when the Fabasoft Folio MMC Service was last recreated. If a Fabasoft Folio Service has been deleted, recreate it in the Fabasoft Folio Server Management by clicking “All Tasks” > “Recreate Service” in the Fabasoft Folio domain.
- *Service Information*
  Shows information on the current connection and operating status of the Fabasoft Folio MMC Service.
  - *Mode*
    Shows the operating status of the Fabasoft Folio MMC Service. “Normal operation” signifies that the Fabasoft Folio MMC Service is available for queries. “MMC Service <Name of Service> is not available” signifies that the Fabasoft Folio MMC Service is not available (e.g. when the service is stopped). This value will be displayed in each field that takes its value directly from the Fabasoft Folio MMC Service (e.g. *Information about MMC Service Areas*).
- *Server Information*
  - *CPU*
    Shows the processor architecture of the server on which this Fabasoft Folio MMC Service runs.
  - *Operating System*
    Shows the operating system of the server on which this Fabasoft Folio MMC Service runs.
  - *Database System*
    Shows the database system of the server on which this Fabasoft Folio MMC Service runs.

- *Hardware ID of Server*
  Shows the MAC address of the network card for the server on which this Fabasoft Folio MMC Service runs.
- *Number of CPUs*
  Shows how many CPUs are on the server that runs this Fabasoft Folio MMC Service.
- *Physical Memory in KB*
  Shows (in kilobytes) the total physical main memory of the server on which this Fabasoft Folio MMC Service runs.
- *Physical Memory Used in KB*
  Shows (in kilobytes) the total physical main memory of the server on which this Fabasoft Folio MMC Service runs.
- *Virtual Memory of Process in KB*
  Shows (in kilobytes) the virtual memory available on the Fabasoft Folio MMC Service.

- *Additional Parameters*
  Defines additional parameters for the Fabasoft Folio MMC Service. These parameters can be used, for example, for debugging purposes.

4.7 Managing Service Definitions

Definitions for services (e.g. flattening tables) define access to external data sources (e.g. MAPI integration). Flattening causes certain object properties to be written to a separate table. This functionality shortens search times for properties that are searched for on a frequent basis. Content properties cannot be flattened.

Definitions for services are objects of the following object classes:

- *Stored Procedure for Named Transactions*
- *Service External Data Definition*
- *Service Data Source*
- *Query Procedure*
- *Service Table Definition*

When installing definitions, new tables are created in the Fabasoft Folio databases for storing flattened properties or keys for external data sources, for example.

Service definitions are managed here: *Domain Administration* > *Domain Objects* > *Service Definitions*.

4.7.1 Stored Procedure for Named Transactions

"Named Transactions" are transactions which have been assigned a name. When using transaction tags, this name will be explicitly noted in the database server transaction log.

The Microsoft SQL server uses named transactions to restore multiple databases to a consistent status at the same point-in-time.

Editing a Stored Procedure for Named Transactions' Metadata

- *Multilingual Name*
  Defines the name.

- *Install Immediately During Load*
  Defines whether to install this definition when loading the software components with Fabasoft Folio Server Management.
- *Installed Statements*
  Defines an instruction for creating "Stored Procedures". In the *Database System* field enter the database system to which the code in the *Statement* field applies. The instruction will not be written directly to the Fabasoft Folio domain, but imported as a file (e.g. text file).
- *Executed Statements*
  Defines the instruction Fabasoft Folio uses to run the query procedure. In the *Database System* field enter the database system to which the code in the *Statement* field applies. The instruction will not be written directly to the Fabasoft Folio domain, but imported as a file (e.g. text file).
- *Statements to Uninstall*
  Defines an instruction on uninstalling "Stored Procedures". In the *Database System* field enter the database system to which the code in the *Statement* field applies. The instruction will not be written directly to the Fabasoft Folio domain, but imported as a file (e.g. text file).

4.7.2 Service Data Source

Service Data Sources are used by Fabasoft Folio COO Services to determine how a Fabasoft Folio COO Service will connect with an external data source.

Editing a Service Data Source's Metadata

- *Multilingual Name*
  Defines the name.
- *Parameters*
  Defines how a Fabasoft Folio COO Service connects with the external data source.
  - *Parameter Type*
    Defines the type of parameter.
    Example: Driver
  - *Parameter Value*
    Defines the parameter value.
    Example: ODBC
  - *Software Component*
    A defined software component prevents your configuration being overwritten by an update.

4.7.3 Query Procedure

Query procedures store configuration data that enable "Stored Procedures" to be used for searches. A "Stored Procedure" consists of a series of SQL commands, stored in compiled form in the database system. These can be used in the Fabasoft Folio domain to search for objects (e.g. if the users often run the same search query).

Editing a Query Procedure's Metadata

"Query Procedure" tab

- *Usable for Object Class*
  Defines the object class to which the query procedure applies.

- *Parameters*
  Defines the properties valid for the query procedure.
- *Multilingual Name*
  Defines the name.
- *Install Immediately During Load*
  Defines whether to install this definition when loading the software components with Fabasoft Folio Server Management.
- *Installed Statements*
  Defines an instruction for creating "Stored Procedures". In the *Database System* field enter the database system to which the code in the *Statement* field applies. The instruction will not be written directly to the Fabasoft Folio domain, but imported as a file (e.g. text file).
- *Executed Statements*
  Defines the instruction Fabasoft Folio uses to run the query procedure. In the *Database System* field enter the database system to which the code in the *Statement* field applies. The instruction will not be written directly to the Fabasoft Folio domain, but imported as a file (e.g. text file).
- *Statements to Uninstall*
  Defines an instruction on uninstalling "Stored Procedures". In the *Database System* field enter the database system to which the code in the *Statement* field applies. The instruction will not be written directly to the Fabasoft Folio domain, but imported as a file (e.g. text file).

4.7.4 Service Table Definition

Editing a Service Table's Metadata

- *Multilingual Name*
  Defines the name.
- *Install Immediately During Load*
  Defines whether to install this definition when loading the software components with Fabasoft Folio Server Management.
- *Requested Table State*
  Defines the table state.
  - "Inactive"
  - "Create on the COO Service next starting"
  - "Active"
  - "Corrupt"
  - "Delete on the COO Service next starting"
  - "Change columns on the COO Service next starting"
- *Internal Tables*
  - *Compound Type/Object Class*
    Shows the compound type (or the object class) for which a separate table exists.
  - *Table Name*
    Suggest a name for the table. The character string contained in this field will be automatically modified where necessary.
    - Table names have the prefix "fsc".

        - The name will be checked for conformity with the syntactic rules of the relevant database system. Special characters are substituted by "_".
        - No SQL key words can be used.
    - *Defined Table State*
      Shows whether the table is activated and therefore in use.
    - *Table Properties*
      Shows which properties are stored in the table.
    - *Used by Properties*
      Compound types can be used by different properties. This list references properties with data to be stored in the table of the composite type. If no composite properties are listed here, all properties using the composite type will be added automatically on storing.

4.8 Archives

You can find more information on the Fabasoft iArchive in the White Paper "Installation and Configuration of Fabasoft iArchive".

Archives are managed here: *Domain Administration* > *Domain Objects* > *Archives*.

4.9 Software Components

Software products consist of individual software components.

Note: The software components are located as COO files of the respective software products in the corresponding folder on the installation media (`Setup/<software product>/Package`).

Software components can be installed and updated not only via a software product, but also individually. To do so, use the Fabasoft Folio Server Management or Fabasoft Folio Web Management.

Software components are managed here: *Domain Administration* > *Domain Objects* > *Software Components*.

Editing the Software Component's Metadata

"Software Components" tab

The "Software Components" tab provides the following fields:

- *Reference*
  Defines the reference of the software component. The reference of software components is in upper case.
- *Version Number*
  Defines the software component's version number.
- *Name*
  Defines the full name of the software component.
- *State*
  Defines the state of the software component:
    - "Installed"
      Defines that the software component is intended for productive use.

- "In Development"
  Defines that the software component is in development.
- "Deleted"
  Defines that the software component is deleted and no longer usable. The name of the software component has the suffix "(Deleted)"

- *Copyright*
  Defines a copyright note.
- *Translations*
  Defines the copyright note for different languages.
- *Prerequisite Components*
  Shows software components that are required by this software component. This information is important for the software component to be extracted correctly.
- *Extended Components*
  Shows software components that are extended by this software component.
  Note: Extensions are not checked during extraction. If you forget to specify an extended software component, the extensions will not be added to the Fabasoft Folio domain when installing the software component.
- *References of Optionally Extended Components*
  Shows software components that will only be extended if they are installed in the Fabasoft Folio domain (such as Help components). The absence of these software components will have no negative effects on the function of the developed software components. This makes "for" instructions possible for optional software components.
  Note: You cannot enter a software component that is required (*Prerequisite Components*) or has been extended (*Extended Components*) as an optionally extended component.
- *Contents*
  Defines files that this software component requires.
  - *Base Name*
    Defines the file name.
  - *Language*
    Defines the language if a different file is to be used for each language. Use this option for Help files, for instance.
  - *Device Type*
    Defines a value for the file to be transferred only to devices of the corresponding device class.
  - *Content Type*
    Defines how and when a file is transferred or loaded.
    Choose from the following options:
    - "Normal"
      The file will be loaded at setup by cooprep.exe or on first use of the software component.
    - "Delayed"
      The file will not be loaded by cooprep.exe but instead loaded at first use of the software component. Only use this option for software components deployed on a few devices.
    - "Developer Help File"
      Defines the developer help file of the software component.

        - "Web Server Specific File"
          The file will only be loaded if Fabasoft Folio is run in the context of the Microsoft Internet information service. These files will load when the Fabasoft Folio Kernel is started. Use this file type for controls in the Fabasoft web browser client, for instance. These controls will be registered on the server as COM objects.
    - *Content*
      Defines the file.
- *Additional Development Domains*
  Defines additional Fabasoft Folio domains where this software component can be developed. You can change the status of the Fabasoft Folio domain from "Installed" to "Under development".
- *Trace*
  Enables trace output oft he software component.

"Installation" tab

- *Compatible With Version Number*
  Defines a compatibility version number.
- *Action to Install Component*
  Defines the Fabasoft Folio action to be used for installing the software component. The action will be run at the device on starting setup. This Fabasoft Folio action can be called several times from the device. This makes it necessary for the Fabasoft Folio action to know whether or not it is being called for the first time.
  Note: The action will run in the context of the user currently logged on.
- *Action to Uninstall Component*
  Defines the Fabasoft Folio action to be used for uninstalling the software component from the device.
  Note: The action will run in the context of the user currently logged on.
- *Component Objects to Be Prepared*
  When installing or first calling the software component, the *Action* `COOSYSTEM@1.1:ComponentObjectPrepare` will be run for all objects entered in this list.
  Use this field for your software component's TypeLib-Object or for data source objects of the *Fabasoft Folio/Data* ODBC driver. The *ComponentObjectPrepare* method of a TypeLib-Object checks whether the TypeLib has already been registered on the local computer. If not, the registry will be updated. This check usually generates only slight overhead.
- *Upgrade Actions per Version*
  This list is used to have object classes automatically adapted to a specific Fabasoft Folio version on upgrading.

4.10 Software Products

A Fabasoft Folio installation consists of software products, each providing functionality for specific areas. For example, the software product *Fabasoft Folio/Base* provides or enables basic functionality, *Fabasoft Folio/AT* does the same for automatic tasks.

Note: Software products are located on the installation media in the setup directory in a separate folder (`Setup/<software product>/Package`).

Software products can be installed or updated with the Fabasoft Folio Server Management or Fabasoft Folio Web Management.

Software products are managed here: *Domain Administration* > *Domain Objects* > *Software Products*.

Editing the Software Product's Metadata

"Software Product" tab

- *Reference*
  Defines the reference of the software product. `Component`, `Edition` and `Solution` are common prefixes for the reference of software products.
  Example: `ComponentBase`
- *Name*
  Defines the full name of the software product.
  Example: Fabasoft Folio/Base
- *Version Number*
  Defines the software product's version number.
- *State*
  Defines the state of the software product.
  - "In Development"
    Is used for development.
  - "Installed"
    Is used for productive use.
- *Copyright*
  Defines the copyright.
- *Copyright Bitmap*
  Defines the copyright as BMP file.

"Components" tab

- *Product Components*
  Defines all software components of the product.
- *References of Product Components*
  Defines all references of the product components.
- *Demo Components*
  Defines all demo components of the product. When installing a domain, you can define whether demo components should be installed.
- *References of Demo Components*
  Defines all references of the demo components.
- *Unselected Components*
  Defines components that are not installed by default.
- *References of Unselected Components*
  Defines all references of the unselected components.
- *References of Excluded Components and Products*
  Defines all references of components, which are not included in the product.

4.11 Languages

English and German are the default languages available for multilingual fields. You can install additional languages using the software product *Fabasoft Folio/LDK*.

Note: The installation only enables multiple language selection for multilingual fields. The user interface will not be available in these languages.

Languages are managed here: *Domain Administration* > *Domain Objects* > *Languages*.

Creating a System Language

When creating a *System Language* enter at least a name.

Editing a System Language's Metadata

"System Language" tab

- *Reference*
  Defines the reference of the language. `LANG_` and `SUBLANG_` are prefixes for the reference of language objects.
  Example: `LANG_GERMAN`, `SUBLANG_ENGLISH_UK`
- *Language Name*
  Defines the name of the language.
- *Default Language*
  Defines whether this language is used as default in the domain.
- *Part of Language*
  Defines a superordinate language, if the object represents a sublanguage.
- *Language IDs on Devices*
  Defines the language identifications of the operating systems used.
  Example: `de`

4.12 Query Scopes

Query Scopes can be assigned to object classes. This limits or expands searches for objects of this object class to specific Fabasoft Folio Domains, Fabasoft Folio COO Stores and Fabasoft Folio COO Services.

Query scopes are managed here: *Domain Administration* > *Domain Objects* > *Query Scopes*.

Creating a Query Scope

When creating a *Query Scope* enter at least a reference and a software component.

Editing a Query Scope's Metadata

"Query Scope" tab

- *Multilingual Name*
  Defines the name of the query scope.
- *Search All Known Domains*
  Defines whether the search covers all accessible domains.

- *Search Default COO Services for Moved Objects*
  In domains or tenants default COO services for moved objects can be defined. These COO services can be included in the search.
- *Domains Searched*
  Defines the domains to be searched.
- *COO Stores Searched*
  Defines the COO stores to be searched.
- *COO Services Searched*
  Defines the COO services to be searched.

5 Configuration Tasks

The following chapters describe common configuration tasks in Fabasoft Folio.

5.1 Automatic Configuration

The Fabasoft Folio configuration can be done using a single configuration file which is applied to the installation.

```
fsceval -eval "coodomain.Configure('/path/to/settings.expr')"
```

The settings file follows the Kernel Expression syntax assuming the compound type `COOSYSTEM@1.1:ConfigurationSettings` (which can be used as reference for the available settings) as local scope and parameters as global scope. The process environment can be accessed using the `::env` key. Please note that the trailing semicolon after each statement is mandatory.

A simple settings file may look like this:

```
domainfinalformat = "pdf";
domaindelegation = {
  delegationactivated = true;
  delegationkeypath = "/path/to/key.p12";
  delegationkeypassword = ::env.DELEGATIONKEYPASSWORD;
};
FSCSMTP@1.1001:ciserver = "smtp.example.com";
FSCSMTP@1.1001:ciport = 25;
FSCSMTP@1.1001:cisecure = true;
```

The environment variables can be either specified by changing the process environment or by using the `fsceval` parameter `env`:

```
fsceval -eval "coodomain.Configure('/path/to/settings.expr')" -env settings.env
```

The local configuration file can be generated using the following command line:

```
fsceval -eval "coodomain.WriteConfigurationSettings('/path/')"
```

A template configuration file can be generated using the parameter `CSM_TEMPLATE`:

```
fsceval -eval "coodomain.WriteConfigurationSettings('/path/', CSM_TEMPLATE)"
```

In case of an update legacy configurations might have been made by modifying the built-in configurations. A merged configuration file can be generated using the parameter `CSM_MERGE`:

```
fsceval -eval "coodomain.WriteConfigurationSettings('/path/', CSM_MERGE)"
```

Please refer to the enumeration type `ConfigurationSettingsMode` for further generation options.

5.2 Configuration Evaluation Order

A distinction is made between built-in configurations supplied with Fabasoft Folio and local configurations used to configure settings based on the local environment and the customer needs. All built-in configurations can be found in the domain administration in the "Configuration Objects" list. The local configurations are stored in the current domain or tenant ("Components Configuration" tab).

Configuration Hierarchies

Configurations of the same type can be specified as configuration hierarchy. This is done using the *Based on* field ("Configuration" tab) of a configuration.

If the *Based on* field is empty, the configurations from the current domain and the domain type (or the default configuration, if the domain type hierarchy does not specify a configuration) are added automatically to the evaluation hierarchy.

In most cases, the effective configuration value is a result of the hierarchy. For example, if no value is specified in the topmost configuration the value is looked up in the next configuration in the hierarchy and so on.

Which Configuration Is Used?

Following mechanism is performed to determine the used configuration:

1. Local configurations
   1.1. If a configuration is specified in the current tenant ("Components Configuration" tab), this configuration inclusive all "based on" configurations are used.
   Note: If no "based on" configuration is defined, the configuration in the current domain will be used, too. If no configuration is found in the current domain or the configuration also defines no "based on" configuration, the firstly found built-in configuration will be used, too.
   1.2. If a configuration is specified in the current domain ("Components Configuration" tab), this configuration inclusive all "based on" configurations are used.
   Note: If no "based on" configuration is defined, the firstly found built-in configuration will be used, too.
2. Built-in configurations
   2.1. If no configuration was found, the configuration of the *Domain Type* (and its hierarchy) of the current tenant ("Components Configuration" tab) inclusive all "based on" configurations are used.
   2.2. If no configuration was found, the configuration of the *Domain Type* (and its hierarchy) of the current domain ("Components Configuration" tab) inclusive all "based on" configurations are used.
   2.3. If no configuration was found, the default configuration inclusive all "based on" configurations (but typically a default configuration has no base) are used.

5.3 Announcements

Announcements are shown on the welcome screen or notifications and can be used to inform users about events like a maintenance downtime. Announcements are defined in the *Announcement Configuration*.

To define an announcement, proceed as follows:

1. Edit the desired *Announcement Configuration*.
2. On the "Announcement Configuration" tab, create a new announcement.
3. Define what, when and to whom the announcement should be displayed.

Note: If you test your settings of an announcement, duplicate it to ensure that you see the changes of the announcement in the welcome screen, because the old setting may be cached for the test user.

Following properties are available for announcements:

- *Title*
  The title of the announcement is displayed in bold.
- *Always Show in Welcome Screen*
  Defines whether, the announcement is shown in the welcome screen, even if the user has deactivated the welcome screen and even if welcome screens are generally disabled in the virtual application configuration ("GUI" tab > *Show Welcome Screen*).
- *Symbol for Announcement in Progress*
  The symbol that is shown while steps of the announcement are not processed by the user.
  Note: If the announcement has no steps to be processed, this symbol will never be shown.
- *Description for Announcement in Progress*
  The description that is shown while steps of the announcement are not processed by the user.
  Note: If the announcement has no steps to be processed, this text will never be shown.
- *Elements of Announcement*
  An announcement can consist of several steps that have to be processed by the user. A description of such steps can be found below.
  Note: If all steps are completed, they will be hided.
- *Expression to Be Done When All Steps Have Been Completed*
  The defined expression is executed after all steps have been completed.
- *Description for Completed Announcement*
  The description that is shown when all steps have been completed or when no steps have been defined.
- *Symbol for Completed Announcement*
  The symbol that is shown when all steps have been completed or when no steps have been defined.
- *Valid from*
  The announcement is only shown from this date. If left empty, the announcement is shown until the *Valid to* date is reached.
- *Valid to*
  The announcement is only shown until this date is reached. If left empty, the announcement is shown form the *Valid from* date.
- *Time Span to Complete (in Days)*
  If steps are defined, the user can carry out all steps within the defined time span. The first executed step starts the time span calculation and may override the *Valid to* date for this user.
- *Show Announcement for Following Users/Groups*
  The users and groups who should see the announcement.

- *Also Show Announcement for Users If*
  An expression that defines users who should see the announcement, too.
  Note: The user has to be defined in at least one of the two properties *Show Announcement for Following Users/Groups* and *Also Show Announcement for Users If* to be able to see the announcement.
- *Show Announcement for All Concerned Objects*
  Defines whether the announcement should be displayed multiple times, based on the objects returned by the *Concerned Objects* expression.
- *Concerned Objects*
  An expression that defines the concerned objects.

Following properties are available for announcement steps:

- *Title*
  The title of the announcement is displayed in bold.
- *Hyperlink*
  If the hyperlink is clicked by the user, the step is marked as done. If the *Expression to Validate* always returns false, the step is only marked as done when the user clicks the hyperlink.
- *Description*
  The description of the step.
- *Symbol for Announcement Element in Progress*
  As long as a defined hyperlink is not clicked and the expression to validate is false this symbol is displayed.
- *Expression to Validate*
  If true, the step is marked as completed.
- *Symbol for Completed Announcement Element*
  If a defined hyperlink is clicked or the expression to validate is true, this symbol is displayed.

5.4 Transfer/Publish a Teamroom

To allow users to transfer or publish Teamrooms from Fabasoft Folio to the Fabasoft Cloud, the following configuration settings are needed.

Note: The functionality is only available in the Fabasoft Private Cloud, Fabasoft Cloud Enterprise and Superior.

Configuration in the Fabasoft Cloud

To authorize users to transfer or publish a Teamroom, proceed as follows:

1. Navigate in the cloud organization, in the “Advanced Settings”, in the “OAuth Clients” area.
2. Create an *OAuth Client*.
3. Edit the properties of the OAuth client. The data in the fields *Client ID* and *Client Secret* are needed for the configuration in Fabasoft Folio. In the *Purpose* field, you must define the following three web service definitions: “Transfer Teamroom”, “Retract Teamroom” and “Recover Teamroom”.
4. All users who have at least read access in the Teamroom are entitled to transfer or publish Teamrooms. Users in the Fabasoft Cloud and Fabasoft Folio are identified by the e-mail address.

Configuration in Fabasoft Folio

To determine which Fabasoft Cloud domains should be allowed as target domains, proceed as follows:

1. Define in the current domain on the "System Configuration" tab in the *Target Domain* field the desired Fabasoft Cloud domains. You can either create a new *Linked Domain* or edit a linked domain provided by the product.
2. In the *Base URL* field, type the URL of the Fabasoft Cloud domain.
3. Set the domain active (*Active* field) and available (*Availability* field).
4. Enter the *Client ID* and *Client Secret*, generated by the OAuth client in the Fabasoft Cloud.
5. Click "Next".

After a successful configuration, the "Transfer Teamroom" command is available in the context menu of Teamrooms.

Note: For automatically continuing stalled Teamroom transfers an *Automated Task* executing the `FSCTRANSFER@1.1001:CheckStalledTransfers` action can be created.

5.5 SMTP Configuration

Use cases that send e-mails on the server-side require a third-party SMTP server and a corresponding configuration in Fabasoft Folio.

The SMTP server can be configured in an instance of `FSCSMTP@1.1001:Configuration`.

Following properties can be defined:

- *Server*
  Defines the hostname or IP address of the SMTP server.
- *Port*
  Defines the port of the SMTP server that should be used for sending an e-mail.
- *Sender E-Mail Address (on Behalf of)*
  Defines the default sender. This means that the e-mail will be delivered "on behalf of".
  Example: `Office <office@fabasoft.com>`
  The e-mail (sent by user David Porter) will contain:
  `Office <office@fabasoft.com> on behalf of David Porter`
- *Expression for Computing the Sender Name*
  A Fabasoft app.ducx Expression to calculate the sender name that is displayed in an e-mail client.
  Example:
  ```
  STRING @sendername;
  @sendername = coouser.userfirstname + " " + coouser.usersurname;
  return @sendername;
  ```
  An e-mail client would show:
  `David Porter <david.porter@fabasoft.com>`
- *Additional Info*
  Additional information for the SMTP server can be configured as key/value pairs. Currently the implementation considers the following keys:
  - `"username"`
    Used for authentication at the SMTP server.

- `"password"`
  Used for authentication at the SMTP server.
- `"timeout"`
  Timeout threshold for the SMTP server session.
- `"content-charset"`
  A specific content character set for the e-mail transfer (Default: `UTF-8`).

- *Well Known Mail Recipients*
  In this field e-mail addresses can be stored. For each e-mail address a context has to be defined that is used as key (normally a string object). To get the e-mail for a specific context the action `FSCSMTP@1.1001:GetWellKnownMailRecipients` is provided.

5.6 Scheduling

Scheduling is used for several use cases (e.g. follow-ups) that carry out tasks repeatedly in defined time intervals.

To enable scheduling, proceed as follows:

1. Navigate to the *User* object of the user who runs the Fabasoft Folio AT Service.
2. Edit the user and click the "Advanced" tab.
3. In the *List of Automated Tasks* field select a list of automated tasks or create a new one.
4. Edit the properties of the list of automated tasks.
5. In the *Tasks* field create a new automated task and edit its properties.
6. In the *Start on/at* field, enter a date on which the automatic task should run for the first time. Additionally, define a *Repetition* and *Repetition Type*.
   In the *Action* field select one of the following described actions and save the settings.
7. Repeat step 5 and 6 to define an automated task for each action described in the following.

Automated tasks for all of the following actions are required. The assignment of a background task to an automated check deadline task is based on the *Duration* field of the background task definition.

- `FSCSCHEDULE@1.1001:CheckDeadlines`
  Background tasks instantiated prior to version 2019 can only be handled by `CheckDeadlines`. Newer background tasks will not be handled by `CheckDeadlines`.
- `FSCSCHEDULE@1.1001:CheckVeryLongDeadlines`
  Only executes background tasks with duration "Very Long" (`DD_VERYLONG`).
- `FSCSCHEDULE@1.1001:CheckLongDeadlines`
  Only executes background tasks with duration "Long" (`DD_LONG`).
- `FSCSCHEDULE@1.1001:CheckDefaultDeadlines`
  Only executes background tasks with no defined duration or duration "Default" (`DD_DEFAULT`).
- `FSCSCHEDULE@1.1001:CheckDefaultMailDeadlines`
  Only executes background tasks with duration "Default (E-Mail)" (`DD_DEFAULT_MAILS`).
- `FSCSCHEDULE@1.1001:CheckDefaultRuleDeadlines`
  Only executes background tasks with duration "Default (Rule Tasks)" (`DD_DEFAULT_RULES`).
- `FSCSCHEDULE@1.1001:CheckShortDeadlines`
  Only executes background tasks with duration "Short" (`DD_SHORT`).
- `FSCSCHEDULE@1.1001:CheckVeryShortDeadlines`
  Only executes background tasks with duration "Very Short" (`DD_VERYSHORT`).

- `FSCSCHEDULE@1.1001:CheckDeadlineDateInitialization`
  This action is used for checking the initialization of deadline dates (e.g. if a follow-up should be based on a base date property that is initially empty).
- `FSCSCHEDULE@1.1001:CheckDeadlineDateRecalculation`
  This action is used for recalculating deadline dates (e.g. if the execution date of a follow-up should be recalculated when the base date changes).

5.7 Notifications

This chapter describes configuration possibilities for notifications and necessary settings for enabling e-mail notifications.

Folio Configuration

On the "History" tab of the used *Folio Configuration,* the following settings for notifications are available:

- *Log History Events*
  Defines whether events are logged for the history.
- *History Limitations*
  Defines the maximum number of logged history entries for certain object classes. By default, 500 entries are logged.
- *Limit for Notification Sources*
  Defines the maximum number of objects to which a user can subscribe. By default, this value is unlimited.
- *Maximum Number of Levels for Get History*
  Defines the maximum number of levels within a folder structure that should be considered for evaluating the history of an object. By default, this value is unlimited.
- *Maximum Number of Objects for Get History*
  Defines the maximum number of objects to be considered for evaluating the history per object. By default, this value is unlimited.
- *Limit for Cached Objects in History Calculation*
  Defines the maximum number of objects allowed to store their events in the cache of an object. The cache of an object is used to speed up the evaluation of the history.
- *Destination for History Cache*
  For each object class an object on which the cache is stored can be defined. By default, all objects of a Teamroom are stored in the Teamroom's cache.
- *Notification Settings*
  For each event type a default display value can be defined. These values can be overwritten by a user.
- *Skipped Event Types*
  It is possible to define a condition under which an event type should not be logged for a certain object class. In the *Expression* field, an expression returning true or false has to be entered.

E-Mail Notifications

To be able to use e-mail notifications, e-mail addresses have to be assigned to the users ("Address" tab, *E-Mail Addresses* field).

Make sure that a working SMTP configuration exists as described in chapter 5.5 "SMTP Configuration". Additionally, make sure that scheduling is configured as described in chapter 5.6

“Scheduling”. In particular the background task definition `FSCFOLIO@1.1001:DeadlineSendUserEventNotifications` (*Duration*: “Default (E-Mail)”) is used to provide the functionality.

5.8 Follow-ups

Follow-ups are sent via e-mail. Therefore, e-mail addresses have to be assigned to the users (“Address” tab, *E-Mail Addresses* field).

Make sure that a working SMTP configuration exists as described in chapter 5.5 “SMTP Configuration”. Additionally, make sure that scheduling is configured as described in chapter 5.6 “Scheduling”. In particular the background task definitions `FSCNOTIFICATION@1.1001:DeadlineSendResubmission` (*Duration*: “Short”) and `FSCNOTIFICATION@1.1001:DeadlineSendRecalcResubmission` (*Duration*: “Short”) are used to provide the functionality.

5.9 Scanning at the Device

When using the “Scan” command on the “Tools” menu it is assumed that the device scanner puts the file path of the scanned document in the clipboard. Either the scanner supports this functionality or the program `copytoclipboard.exe` provided by Fabasoft has to be assigned as post-processing step in the third-party scanner software.

To enable the scan functionality with the `copytoclipboard.exe` tool, proceed as follows:

1. Unzip `Setup\ComponentsBase\Client\copytoclipboard.zip` from the installation media and copy `copytoclipboard.exe` to the client computer.
2. Assign `copytoclipboard.exe` as post-processing step in the scanner software.

5.10 Folio Folder and Folio Network Drive

The Folio Folder can be enabled in the user environment or the group of the user (“Advanced” tab, *Synchronization Mode* field: “No Synchronization”, “Synchronized Folder”, “Synchronized Desktop or Synchronized Folder”).

The Folio network drive can be enabled in the user environment of the user (“Advanced” tab, *Enable Folio Network Drive* field).

Note: By default, the Folio Folder is disabled and the network drive is enabled.

5.11 Long-Term Suspended Activities

Depending on the processing state, activities are displayed on different tabs in the worklist. In order to increase the clarity and improve the performance, the "Long-Term Suspended" tab can be enabled.

For this purpose, a time interval has to be defined in the workflow configuration from when a suspension is considered as long-term. As soon as the submission date falls inside the time interval, the activity will be moved from the "Long-Term Suspended" tab on the "Suspended/Pending" tab (configuration of an automated task is required). This setting does not affect already defined suspensions. If the setting is removed again, the automatic task moves all long-term suspended activities on the "Suspended/Pending" tab.

To define a time interval from when a suspension is considered as long-term, proceed as follows:

1. Edit the *Workflow Configuration*.
   Note: To find the correct configuration see chapter 5.2 “Configuration Evaluation Order”.
2. On the “Worklist” tab, define a *Time Interval From When a Suspension Is Considered as Long-Term*.

To define the automatic task that is used to move the long-term suspended activities, proceed as follows:

1. Navigate to the *User* object of the user who runs the Fabasoft Folio AT Service.
   Note: The user needs search rights for activity instances. By default, users with position *Administration* or *System Administration* have search rights.
2. Edit the user and click the “Advanced” tab.
3. In the *List of Automated Tasks* field select a list of automated tasks or create a new one.
4. Edit the properties of the list of automated tasks.
5. In the *Tasks* field create a new automated task and edit its properties.
6. In the *Start on/at* field, enter a date on which the automatic task should run for the first time.
   In the *Object* field, create a new *List of Background Activities*.
   In the *Action* field, select the `COOWF@1.1:CheckLongTermActivities` action.
   Additionally define the *Repetition* and *Repetition Type* and save the settings.

5.12 Video and Audio Conversion

By default, the video and audio conversion is disabled. To be able to convert videos and audios the third-party product FFmpeg has to be available on the Fabasoft Folio AT server.

To enable the automated task that carries out the conversion, proceed as follows:

1. Navigate to the *User* object of the user who runs the Fabasoft Folio AT Service.
2. Edit the user and click the “Advanced” tab.
3. In the *List of Automated Tasks* field select a list of automated tasks or create a new one.
4. Edit the properties of the list of automated tasks.
5. In the *Tasks* field set the *State* of the “Process Digital Asset Conversions” task to “Activated” and save the settings.

5.13 Google Maps

To be able to use Google Maps within Fabasoft Folio a Google Maps JavaScript API key has to be defined.

To define the API key, proceed as follows:

1. Create a new or edit an existing *Map Configuration*.
   Note: For more information about configuration evaluation, see chapter 5.2 “Configuration Evaluation Order”.
2. On the “Map Configuration” tab, specify the *Host* of Fabasoft Folio (e.g. folio.comp.com) and your own *API Key* obtained from Google.

5.14 Object Class Hierarchy for the Mindbreeze Search

On object classes, the “Generate Thesaurus” context menu command can be used to generate a thesaurus of the whole object class hierarchy. The generated terms contain the file extensions as *Hidden Synonyms* and the object address of the corresponding object class as *Exact Match*.

This thesaurus can be further structured and may serve as the basis for filter options in the Mindbreeze search.