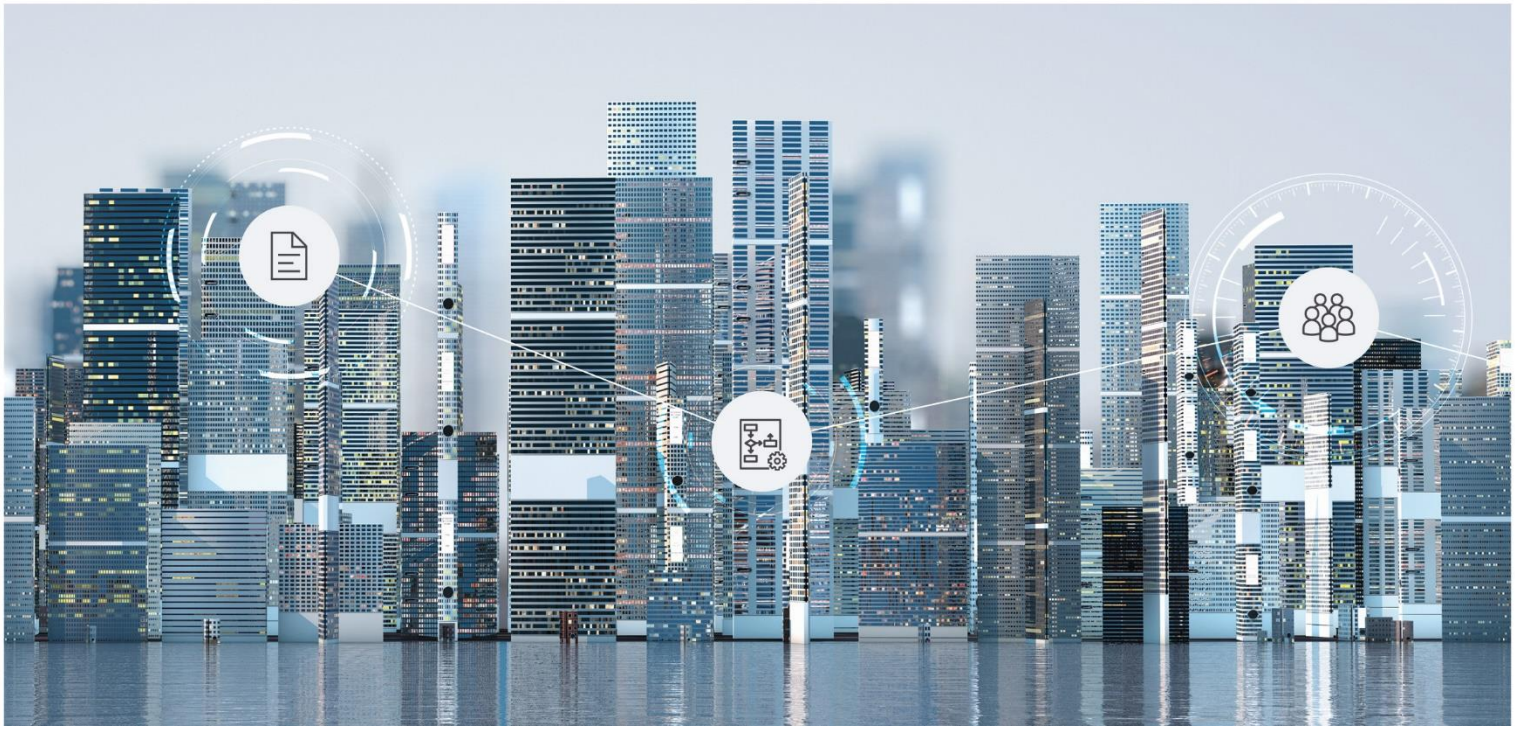
Fabasoft



White Paper

Configuration of Fabasoft Folio Multi-Tenant Installations

Fabasoft Folio 2021 Update Rollup 2

Contents

1 Introduction

This document describes the configuration and usage of tenants. It is separated into two parts. The first part describes to configuration possibilities of a tenant system. The second part describes the installation and configuration of a tenant system.

2 Software Requirements

System environment: All information contained in this document implicitly assumes a Microsoft Windows or Linux environment.

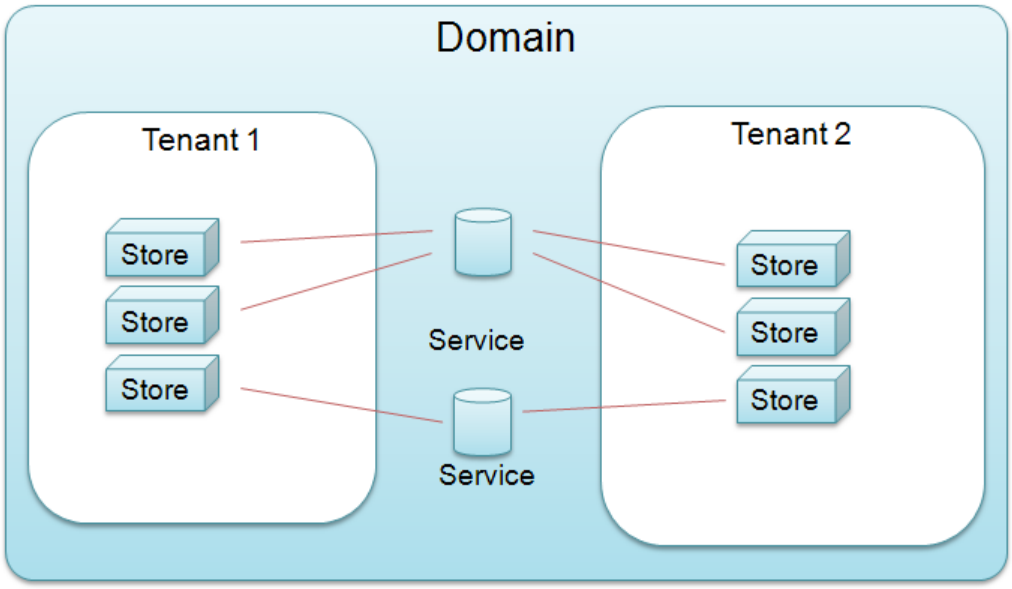
Supported platforms: For detailed information on supported operating systems and software see the software product information on the Fabasoft distribution media.

3 Designing a Multi-Tenant System

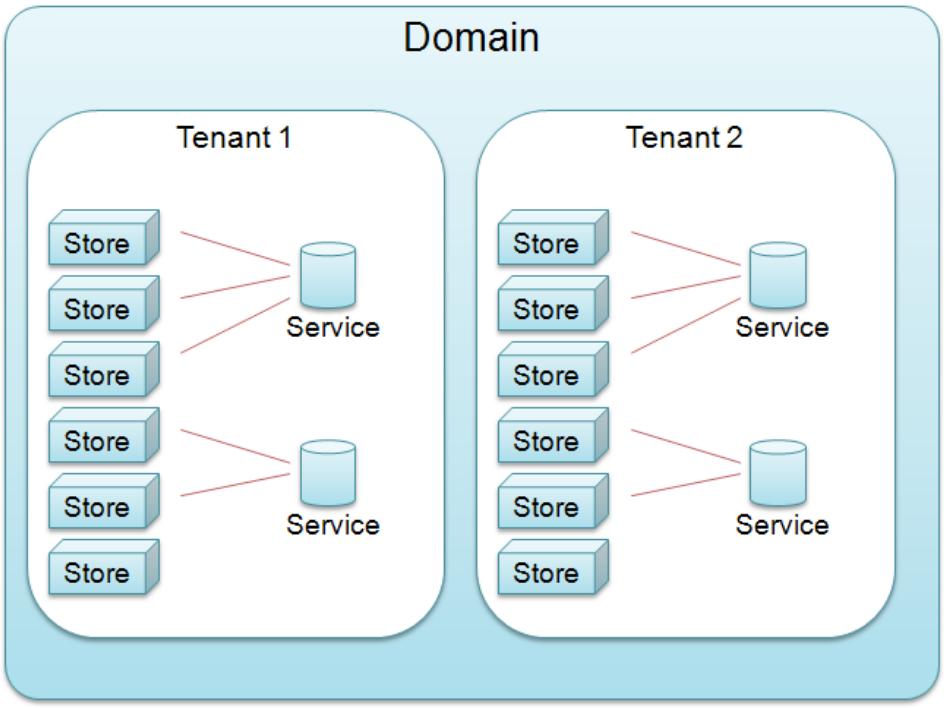
There are different strategies to pursue to create services for tenants.

Approaches:

- **Shared:** The tenants use common Fabasoft Folio COO Services and Fabasoft Folio MMC Services.



- **Connected services:** Each tenant has own Fabasoft Folio COO Services and Fabasoft Folio MMC Services.



Approach	Security Patterns	Extensibility Patterns	Scalability Patterns
Shared	• Access Control Lists	• Tenant Specific Configuration	• Service Based Scale-Out
Connected Services	• Access Control Lists • Service Based Restrictions	• Tenant Specific Configuration • Tenant Specific Persistence Configuration	• Service Based Scale-Out • Tenant Based Scale-Out

3.1 Access Control Lists

In all Fabasoft software products the access to Fabasoft Folio object is controlled by access control lists (ACL). ACLs are defined by *ACL* objects. Each Fabasoft Folio business object has a pointer to an *ACL* object which defined the access to this business object.

3.2 Service Based Restrictions

The service based restrictions are defined by connection configurations of Fabasoft Folio Services. Access to these services can be restricted by client IP address to restrict data access to clients which belong to the given tenant. This restriction can be defined in the *Fabasoft Folio COO Service* object.

Example: It is possible to give a business unit its own Fabasoft Folio Web Service which is only allowed to connect to a specific Fabasoft Folio Backend Service where confidential data is stored.

3.3 Tenant Specific Configuration

The domain configuration and administration may be delegated to the tenants to enable a decentralized administration or can still be handled centrally. The configuration of software components of a domain can be changed on a per tenant basis to customize the software for specific requirements of a tenant, independent from the domain configuration.

3.4 Tenant Specific Persistence Configuration

The Fabasoft Folio objects of each Fabasoft Folio Tenant can be stored in different databases and file systems and can be made accessible via different Fabasoft Folio Services. By using different services per tenant, the data persistence configuration can be customized for specific requirements of a tenant.

Example: For each Fabasoft Folio Tenant own service definitions can be defined.

3.5 Service Based Scale-Out

A Fabasoft Folio Service stores a specific data set in a dedicated database and file system. To handle increasing workloads and data size, Fabasoft Folio Services can be added to an existing installation to provide additional capacity.

3.6 Tenant Based Scale-Out

A new Fabasoft Folio Tenant can be added to an existing installation and hosted by dedicated Fabasoft Folio Services to provide additional capacity for a new environment with an independent configuration.

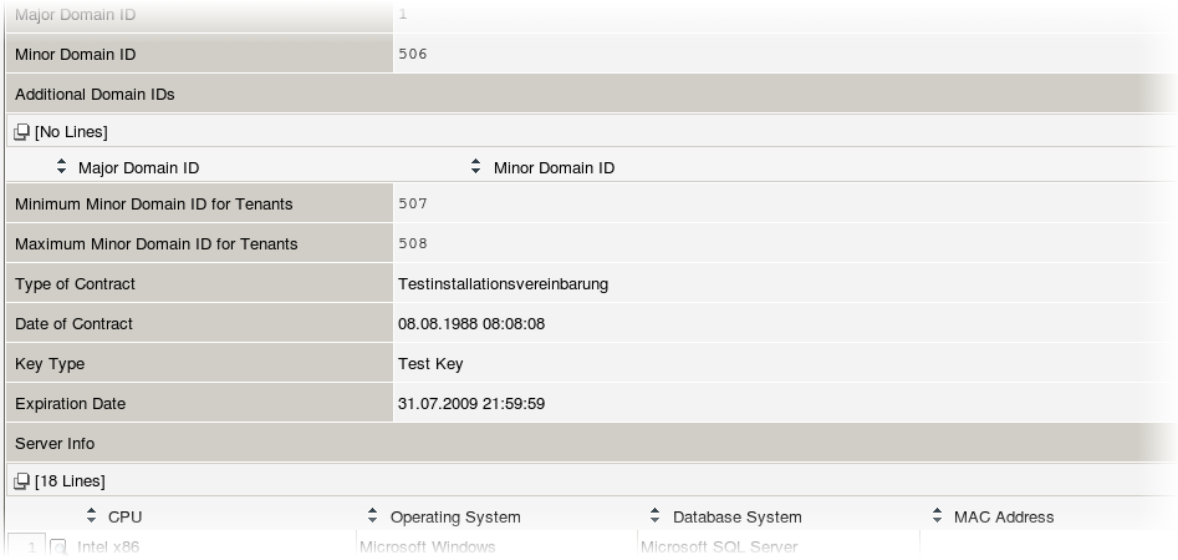
Example: If a new business unit is founded in an organization, the system installation can be extended with a new tenant, which does not affect the existing installation.

4 Installing and Configuring a Multi-Tenant System

4.1 Create a new Tenant

A Fabasoft Folio Tenant is handled like a domain in Fabasoft Folio. Therefore the class for tenants is also derived from the Current Domain class. Every Fabasoft Folio Tenant also has a Domain ID, the Major Domain ID is generally the same for all tenants and primary domains. The tenants of a Fabasoft Folio system indeed distinguish themselves in the Minor Domain ID.

To be able to create Fabasoft Folio Tenants in a Fabasoft Folio Domain, you need a separate product license. The license creates an area of Minor Domain Ids, in addition to the existing Domain ID for the primary domain, which can be used for the creation of tenants.



To create a new tenant in a Fabasoft Folio environment perform following steps:

1. Ensure the license is valid for tenants. If the license is not valid load an appropriate license onto the Fabasoft Folio Domain.
2. Navigate to *Domain Administration* > *Domain Objects*.
3. On the "Domain" tab create a new object of the object class *Tenant*. For the minor domain ID the next free minor domain ID is used.

A Fabasoft Folio Tenants is derived from the Current Domain class. The identification happens via the Domain ID. During creation, the Domain Major ID is created from the primary domain and the first free Domain Minor ID obtained from the product license.

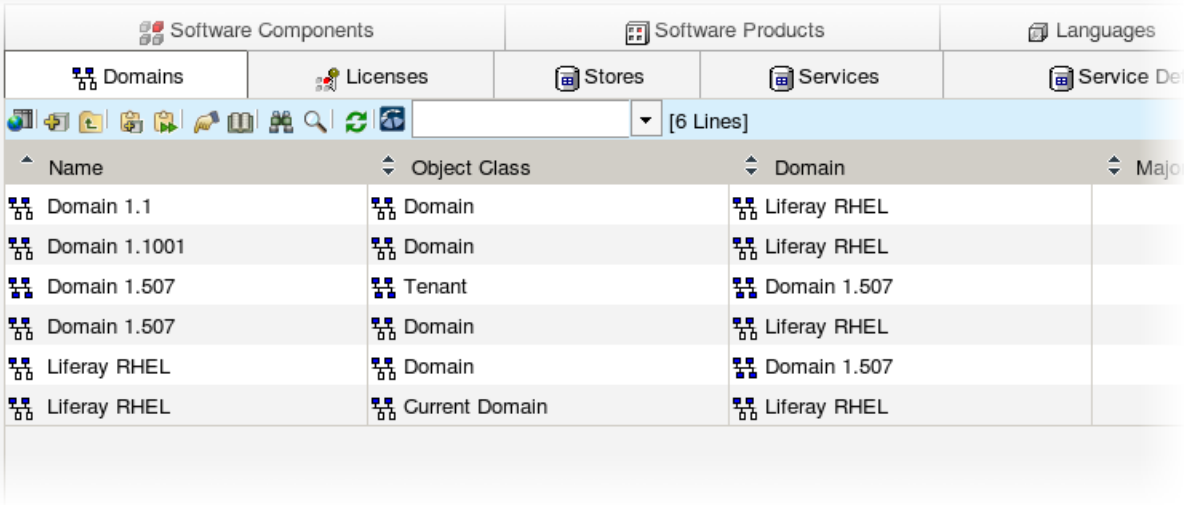
The main domain must be known in the tenant and the tenant must be known in the main domain. Therefore, five objects are generated when a tenant is created.

Tenant:

- the *tenant* object (object class *Tenant*)
- a COO store object
- a MMC store object
- the main domain (object class *Domain*)

Main domain:

- the tenant (object class *Domain*)



Each tenant knows its main domain. This domain is stored in the property *Originating Domain* in the object class *Tenant*.

In a tenant a new *tenant* can also be created. That means that a tenant is the originating domain of another tenant.

4.2 Services and Stores

Tenants have their own COO and MMC stores. The *COO store* of an object defines to which tenant the object belongs.

Each store needs a service, which provides the persistent data management of all objects of the store. All services of a tenant system always belong to the main domain. A service can store object data of one or more stores. The stores can be in one domain or different tenants.

The first COO store of a tenant uses the primary COO services and the first MMC store uses the primary MMC service of the main domain. These stores should only be used for administrative object. So it is necessary to create new COO stores and MMC stores for the tenant. The object placement should also be modified in the object classes.

While creating a tenant the object placement will not be changed. These changes have to be done by the administrator.

If the object placement is not modified all objects of the tenant are balanced to all COO and MMC stores. So also in the first COO and MMC store are user objects which should be avoided.

4.2.1 Create new Services

To create a service only for the tenant, install a service in the main domain, which is not used by any COO or MMC store. Therefore, perform following steps:

1. Start the Fabasoft Server Management
2. Start the Create Services Wizard
3. Create a COO service and an MMC service without any stores.

Now create stores in the Fabasoft Folio desk. To see how to do that see next chapter.

4.2.2 Create new Stores for the Tenant

It is recommended to create COO- and MMC stores for the tenant right after creating one. To create a new store, perform following steps:

1. Switch to the tenant where the store should be created.
2. Create new objects of the object class *COO Store* and *MMC Store*.
3. Assign a *COO Service* and a *MMC Service* to the object.

Note:

- To create stores in a tenant you must be word in the context of the tenant. It is not possible to change the *Domain* of the store afterwards.
- There is a limit of stores per tenant of 254.
- Be careful at the definition of services for stores. The service cannot be modified after the first assignment.

4.3 Object Placement in a Tenant System

The object placement of object classes defines in which *COO Store* and *MMC Store* new objects are created.

The object placement is one of the most important points at the design of a Fabasoft Folio installation. In a single domain installation there is the decision in which store the object of an object class are created. In a tenant system there is an additional dimension. Should the object be created in the tenant or in the main domain?

The COO store defines the assignment of an object to a tenant. An object which is created in the COO store of tenant A is an object of tenant A, no matter which user this object has created or which user the owner of the object is.

The property Object Placement (`COOSYSTEM@1.1:domainplacement`) defines in which COO store the objects of the class are created.

4.3.1 Definition of COO Stores in a Tenant System

If a new object is created in a tenant, the software checks whether there are stores in the property *COO Stores for new Objects* for the object class in the current domain.

- If there is exactly one *COO store* the object is created in this store.
- If there are more than one *COO* store the object is created in the store with the lowest free object id (assumed that this store contains the lowest number of objects).
- If there is no *COO store* of the tenant, the object placement of the base class is used.
- If no configuration is found, the store with the lowest free object id (among all stores of the current domain) is used.

4.3.2 Tenant Comprehensive Object Placement

Fabasoft Folio objects can also be created in Fabasoft Folio COO Stores that are not directly connected to the tenant of the creator.

Example: In the primary domain a Fabasoft Folio Store can be set up in which all objects of a particular application (e.g. the Fabasoft Workflow software component) are saved.

To formulate the Fabasoft Folio Tenants spanning object placement create an *object class* entry on the "Object Placement" tab of the desired Fabasoft Folio tenant with the property *Allow All Tenants* enabled and insert the desired Fabasoft Folio COO Store in the *COO Stores for new Objects* property.

4.4 Domains in a Tenant System

If you consider tenant systems, you can identify various Fabasoft Folio Domains when working with tenants.

4.4.1 Base Domain

When starting the client, a user establishes a connection to one Fabasoft Folio Domain. The domain to which this connection is established is the base domain. The Fabasoft software components are primarily loaded from this domain. In tenant systems this is generally the primary domain. The network address of the primary Fabasoft Folio COO Service is also the address of the base domain.

4.4.2 Home Domain

For every user in Fabasoft Folio a *User* must also be supplied. The user object identifies the user within Fabasoft Folio. The Fabasoft Folio Domain, in which the user object is placed, is marked as the home domain of the user.

4.4.3 Current Domain (Tenant)

In a tenant system every user can be associated with one or more Fabasoft Folio Tenants, in which the user can be active. The user can decide during use, which tenant he wants to work in at that particular moment and change it as needed. The tenant (resp. the domain) in which the user works at any point in time is the current domain.

If no Fabasoft Folio Tenant is being used the current domain is the same as the base domain. The current domain is also delivered with the `CooRuntime::GetCurrentDomain` function.

4.4.4 Object Domain

Every Fabasoft Folio object belongs to a Fabasoft Folio Domain that is defined through the Fabasoft Folio COO Store of the object.

4.5 User in a Tenant System

If you consider a user in the context of a tenant system, you must answer the following questions:

- Where should the *users* be created?
- To which Fabasoft Folio Domain must a *user* connect?
- In which tenants is a *user* allowed to work?

4.5.1 Create User – Home Domain

There are two different ways to create *users*:

- Centrally in the primary domain.
- De-centrally in the Fabasoft Folio Tenants.

If the *user* (and possibly the *groups*) are centrally created in the primary domain, the advantage of simpler management exists. Users do not have to be searched for in various Fabasoft Folio Domains. All users in the domain use the same home domain (that which is identical to the base domain).

If you want to manage the *user* centrally in the primary domain, it is recommendable with the *User class* (and perhaps with other administration classes like *Group*, *User Substitution*) to place the object placement in the Fabasoft Folio COO Store 1 of the primary domain. The placement should be used for all tenants.

With decentralized user management, the user is created in the tenant, in which they are anticipated to work in. This management is equivalent to the locality principal. If the user changes to another tenant (if his field of activity changes), which can happen in many cases, the Base domain, Home domain and current domain are different.

If you want to manage *users* de-centrally, you must change beforehand into the tenant, in which the *users* should be created.

4.5.2 Connection to Domain – Base Domain

The base domain in a tenant system is always the primary domain, no matter which tenant the user works in. Therefore, always use the network address (resp. the Domain ID) of the primary domain to establish the connection to Fabasoft Folio.

4.5.3 Tenants of a User – Current Domain

You define which tenants a user is allowed to work in, in the *Client Domains* property of the *User* object (`COOSYSTEM@1.1:userclientdomains`). If the user should have the right to work in several Fabasoft Folio Tenants, a tenant can be defined as a standard tenant. The standard tenant is automatically used after the user login.

If a *user* has no Fabasoft Folio Tenant associated, then he can only work in the primary domain.

On the "Client Domains" tab in the *Client Domain* property of the *User* class is back linked with the *User* in the *Client Domain* property of the *Current Domain* class. This way you can easily check on a client domain, which user can work in this Fabasoft Folio Domain.

The exchange of the Fabasoft Folio Tenant is achieved on the *Desk* by using the menu item "Tools" > "Settings". On the "Client domain" tab the user can choose the desired *Tenant*. The changing of the tenant is implemented by the `COOSYSTEM@1.1:SetClientDomain` action.

4.6 ACLs in a Tenant System

The scope of a Fabasoft Folio object of a Fabasoft Folio Tenant system is achieved exclusively using the *Access Control List* (ACL) of the *object*. By using the *ACL,* it must be guaranteed that objects of a Fabasoft Folio Tenant are only accessible to users of this tenant.

4.6.1 Check of Domains in Access Control Entries

In *ACLs* the validity of an entry (*Access Control Entry* - ACE) in certain Fabasoft Folio Domains can be restricted. In the ACL editor the Fabasoft Folio Domain of the entry is defined in the first column. This column is generally also used to separate the accesses to objects of various tenants.

If Fabasoft Folio is used as a tenant system, then the domain columns in an *ACL* are compared against the current Fabasoft Folio Tenants.

- If a certain Fabasoft Folio Domain/Tenant is given in the *ACE*, then this entry is only valid if the *user* is currently working in this tenant.
  `coort.GetCurrentDomain() == <domain in ace>`
- If "Object Domain" is used in the *ACE* then this entry is only valid if the *user* is currently working in the Fabasoft Folio Tenant to which the Fabasoft Folio object also belongs.
  `coort.GetCurrentDomain() == theobject.Get_objdomain()`
- If "Owner Domain" is used in the *ACE* then this entry is only valid if the user is currently working in the Fabasoft Folio Tenant to which the *Owner* property referenced Fabasoft Folio object belongs.
  `coort.GetCurrentDomain() == theobject.Get_objowner().Get_objdomain()`

If Fabasoft Folio is no longer used with tenants then the domain column in the *ACL* is compared against the Home Domain of the current *user* (Home Domain = the Fabasoft Folio Domain in which the *user* object of the *current domain* lies).

So instead of `coort.GetCurrentDomain()` the following is compared `coort.GetCurrentUser().Get_objdomain()`.

Note: The comparison of the Fabasoft Folio Domains does not take place by comparing the objects (resp. the addresses). In place of that the similarity of the Domain Major ID and the Domain Minor ID is checked.

4.6.2 Design of ACLs

With the design of *ACLs* you should watch out for the following points (especially with the use of "Object Domain" and "Owner Domain"):

ACLs on Fabasoft Folio component objects are actually not allowed to contain any restrictions on "Object Domain" resp. "Owner Domain" because component objects must be available for all tenants.

- *ACLs* on administration objects should, in the normal case, have no restrictions on "Object Domain" resp. "Owner Domain" – except if this is explicitly desired.
- *ACLs* on normal objects should normally be restricted to "Object Domain" or to explicit domain(s) (in certain circumstances also "Owner Domain"), so that the objects are only available within the tenants.

The mentioned points should be especially noted in connection with the *ACLs* in delivered Fabasoft software components.

If objects are safeguarded between tenants, it must be noted that the combination of "Owner Domain" and "Object Owner" normally makes no sense.

Example:

The following scenario should make clearer the anomaly with the use of "Owner Domain" und "Object Owner":

A user is created in the "HD" primary domain. So a central management of users is implemented. But the user works by default in tenant "B".

An object has, in the *ACL*, an *ACE* with the combination "Owner Domain" and "Object Owner". The mentioned user creates an object with this *ACL*. After the object has been created, the user doesn't have access anymore.

Cause:

When checking whether this *ACE* can be used the “Owner Domain” is compared with the current domain of the *user*.

Owner Domain = "HD"

Current tenant (resp. Current Domain) = "B"

`"HD" <> "B"` ==> *ACE* is not valid for the current user anymore!

If you wish that the Fabasoft Folio objects of a Fabasoft Folio Domain can only be used by those users that work in this domain, then you are better off using the “Object Domain” entry. This entry can of course also be combined with the entry for “Object Owner” in an *ACE*.

Solution:

The Fabasoft Folio object is created by the *user* in the domain "B". In the *ACE* the combination of “Object Domain” and “Object Owner” is used.

Domain of the object = "B"

Current domain = "B"

`"B" = "B" AND cooobj.objowner = coort.CurrentUser()` ==> *ACE* is valid!

4.6.3 Standard ACLs

The Standard ACLs of the basis system

- *ACL for objects for development,*
- *ACL for objects for administration,*
- *Standard ACL for general objects, and*
- *Standard ACL*

take into account the aforementioned points. Every user has read access to objects for development and to object for administration.

The *Default ACL for Common Objects* is especially used by object classes. This *ACL* allows all users of the tenant system to instantiate objects of the class. But *Software Product Licenses* also have this *ACL*. This way every user can also find this object.

The *Default ACL* is used for new objects of end users. This *ACL* restricts the access of every user that originates from the user domain. However, note that with central management of users, this *ACL* must also be adapted (here with a central management of the Home Domain – Owner Domain – with the current user domain not matching, see above).

4.6.4 Default ACLS for new Objects

Every Fabasoft Folio object requires an *ACL*. Therefore during creation, an initialization value is detected using the Attribute Constructor Action `COOSYSTEM@1.1:AttrObjACLConstructor` of the property *ACL* (`COOSYSTEM@1.1:objacl`).

The *Default ACL* for new objects is defined through the following steps:

1. The current domain is detected. In a tenant system, that is the current tenant of the user aside from the base domain to which the user had logged on (return value of the function `CooRuntime::GetCurrentDomain`).

2. The *ACL* is detected from the *Default ACL for New Objects* property (`COOSYSTEM@1.1:classdefaultacl`) of the object class of the object
   2.1. Firstly an *ACL* of the current domain is searched for in the *Default ACL for New Objects* property. If an *ACL* is found, it is used as the initialization value.
   2.2. If no *ACL* for the current domain is found, then the first one in the above mentioned list is taken.
   2.3. If no *ACL* is defined in the list, it is continued on with the next criterion.
3. The *ACL* is detected from *the Default ACL for New Objects* property (`COOSYSTEM@1.1:grdefaultacl`) of the object group.
   3.1. Firstly an *ACL* of the current domain is searched for in the *Default ACL for New Objects* property. If an *ACL* is found, it is used as the initialization value.
   3.2. If no *ACL* for the current domain is found, then the first *ACL* in the above mentioned list is taken.
   3.3. If no *ACL* is defined in the list, it is continued on with the next criterion.
4. The *ACL* is detected from the *ACL Objects* property (`COOSYSTEM@1.1:graclobjects`) of the object group.
   4.1. Firstly an *ACL* of the current domain is searched for in the *ACL Objects* property. If an *ACL* is found, this is taken as the initialization value.
   4.2. If no *ACL* of the current domain is found then the first *ACL* in the above mentioned list is used.
   4.3. If no *ACL* is defined in the list, it is continued on with the next criterion.
5. *Standard ACLs* of the System (`COOSYSTEM@1.1`) software component are assigned. For certain object classes the *Standard ACL* from the system is predefined.
   5.1. *Development Objects* (*Properties*, *Types*, *Actions* etc. – but not object classes) generally get the *ACL for Developer Objects* (`COOSYSTEM@1.1:DefaultDeveloperACL`).
   5.2. *Administration Objects* (*User*, *Groups*, *Substitutions*, etc.) generally get the *ACL for Administration Objects* (`COOSYSTEM@1.1:DefaultAdministratorACL`).
   5.3. Object classes generally get the *Default ACL for Common Objects* (`COOSYSTEM@1.1:DefaultGlobalACL`).
   5.4. Normal Objects of the end user generally get the *Default ACL* (`COOSYSTEM@1.1:DefaultACL`).

4.7 Object Search in a Tenant System

In the domain clause of a search request one can define the search area of a Fabasoft Folio query.

- No Domain Clause
  The search happens in all Fabasoft Folio COO Services to which the Fabasoft Folio Kernel (hosted by the Fabasoft Folio Web Service) is connected. That means that it will also be searched for in the data of other Fabasoft Folio Tenants.
  Example:
  `SELECT objname FROM COOStore`
  finds all Fabasoft Folio COO Stores of the primary domain and all tenants.
- LOCAL
  The search only happens in the current Fabasoft Folio Tenant and in the primary domain of the tenant. So no objects from other tenants are returned from the search.

Example:
`LOCAL SELECT objname FROM COOStore`
finds all Fabasoft Folio COO Stores of the primary domain and current client domains

- DOMAINS ('Domain ID')
  The search only happens in the specified Fabasoft Folio Domain, e.g. in the local tenant.
  Example:
  `DOMAINS ('10.113') SELECT objname FROM COOStore`
  finds all Fabasoft Folio COO Stores of the tenants with the Domain ID 10.113

4.7.1 Local Search

The local search in the (`CooRuntime::SearchLocalObjects`) cache generally finds all Fabasoft Folio objects that are located in the user cache.

If objects of object class `COOSYSTEM@1.1:Domain` are searched for, then only Fabasoft Folio Domain objects of the current domain are returned.

With the Boolean transaction variable `COOSYSTEM@1.1:TV_SEARCHCURRENTDOMAIN (ID=5)` the local search can also be restricted to other objects classes in the current domain. Set this variable to "*True*" if this behavior is desired from `CooRuntime::SearchLocalObjects`.

4.7.2 Domains in the Search Dialog

In the search dialog the user has the option to restrict the search to particular Fabasoft Folio Domains. The domains in the list of the search dialog are detected with a local search. Therefore, by default, only two domains are entered if one works in a tenant – the tenant and the primary domain.

If the user of tenant A should also be able to search in tenant B then you must perform following steps:

1. Change into tenant A.
2. Place a new domain object for tenant B into this tenant.
   2.1. Enter the Major Domain ID and Minor Domain ID of tenant B.
   2.2. Set the Directly Connected Domain property to "*True*".
3. Change into tenant B.
4. Place a new domain object for tenant A into this tenant.
   4.1. Enter the Major Domain ID and Minor Domain ID of tenant A.
   4.2. Set the Directly Connected Domain property to "*True*".

4.8 Development of Software Components

With Fabasoft software component development in tenant systems you must be especially careful that the Fabasoft software components are only installed once in the entire tenant system. This means that all tenants use the same component objects (however it isn't an isolated domain configuration).

It must be possible, however, for component object settings to be individually definable. The Fabasoft software component system takes this into consideration, for example, with the object placement and with the choice of the *Default ACL* for new objects.

4.8.1 Configuration Objects for Tenants

For global settings Software Components in Fabasoft often make available *Configuration Objects*.

This way, for example, the functionality of Fabasoft Folio software products can be easily adapted to individual customer requirements in many areas by using the Configuration.

Note: Configuration *Objects* are delivered as *Component Objects*. For this reason these *Configuration Objects* can be delivered with default settings along with the Fabasoft software components. Additionally, through the fixed object address of the object in the methods, the Fabasoft software component can be directly referenced and therefore quickly found.

As already described above, Fabasoft software components are only installed once in a tenant system (however it isn't an isolated domain configuration). Therefore, by default, also only one *Configuration Object* exists – although the settings must be different among individual tenants (especially in ASP Operation).

The possibility of defining a *Configuration Object* for every Fabasoft Folio Tenant must be provided by the Fabasoft software component. The *Configuration Object* that is delivered with the Fabasoft software component saves the general settings that are valid for all Fabasoft Folio Tenant. If required a *Configuration Object* can also be created for each Fabasoft Folio Tenant in which the settings for this tenant are placed.

Consequentially it follows that:

- *Configuration Objects* should be creatable in production domains. This means that the *Configuration Object* is not allowed to be abstractly defined.
- *Configuration Objects* must be found quickly. Therefore a pointer that points to the *Configuration Object* valid for this Fabasoft Folio Domain should be available in the *current domain* object. This means that the *Current Domain* class (`COOSYSTEM@1.1:CurrentDomain`) must be extended to include a pointer property of that type.
- The Fabasoft software component should, using an action, detect the C*onfiguration Object* that should be used for the current Fabasoft Folio Domain. Only if no separate *Configuration Object* is defined for the current domain, should the *Configuration Object* of the Fabasoft software component be accessed via the object address of the *Configuration Object*.

Example:

The Fabasoft Folio software products deliver the *FSCOWS@1.1001:Web Service Configuration* object along with the FSCOWS software component. In this object various settings for Friendly URLs and WebDAV Integration are stored.

The class *Current Domain* (`COOSYSTEM@1.1:CurrentDomain`) would be extended with the property *FSCOWS@1.1001:Web Service Configuration*. In this property every tenant can now lay down a different *Web Service Configuration*.

4.8.2 ACL for Object Classes

ACLs for object classes and other Fabasoft component objects with the domain definition "Owner Domain" must be checked for their plausibility. In tenant systems, it cannot be assumed that the owner domain is also the current user domain. The owner of Fabasoft component objects is generally the *system administrator* of the primary domain. However, a user works in a Fabasoft Folio Tenant. This means that the domains are different. Consequently, these objects cannot be used by the user in the tenant any more.