flexera

Spider 6.4

Spider Importer

Legal Information

Book Name: Spider 6.4 Importer

Part Number: SPI-0001-IM06

Product Release Date: November 2019

Copyright Notice

Intellectual Property

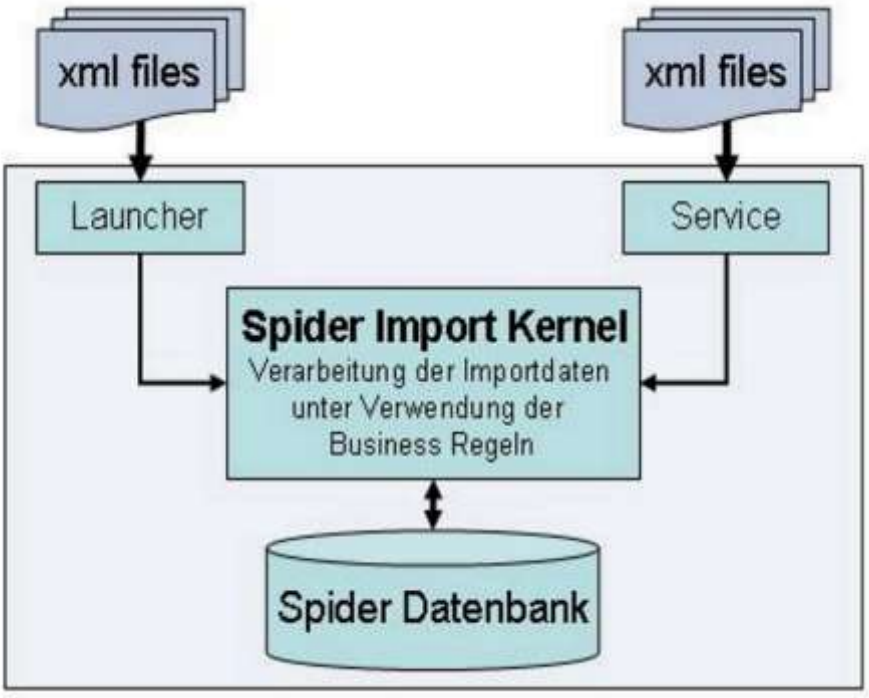
Restricted Rights Legend

Content

Introduction

The Spider Importer represents the interface to the Spider products. Use the Spider Importer to transfer data and objects into the Spider databases by applying the implemented business rules.

The import is carried out using special XML files, which contain the data to be updated. Apart from processing via the *Spider Import Launcher*, you can implement continuous interfaces by using the Spider services. The following figure shows schematically the processing of the XML import files via the Spider Import Kernel.



The processing of the import files can be carried out in two ways. On the one hand, you can load the XML file from the Spider Import Launcher for processing. On the other hand, you can save the file in a customer-specific folder, which is monitored by the Import service. The files in this folder will be processed automatically if the service is activated. The import will create new Spider objects and update or delete existing objects.

The scope and structure of Spider import objects is a central subject in this document. In chapter 1, the import of objects as well as the fundamental XML structure of the import objects will be explained. Refer to the detailed description of these objects in the following chapters.

CHAPTER 1

Basics

In this chapter

1.1 Design of the XML structure

The Windows application *Spider Import Launcher* and the import service each process identical XML files. The scope of the supported import objects is described in this document. When using the objects, make sure to use the correct spelling.

The design of the XML structure depends on the used Spider import objects. In principle, the structure is similar for all import objects:

```
<?xml version="1.0" encoding="UTF" ?>
<ImportData>
<ImportConfig>
        <Mode>IMPORT</Mode>
        <Culture>de-DE</Culture>
</ImportConfig>
        <SpiderObjectName>
                <Attribut1>...</Attribut1>
                <Attribut2>...</Attribut2>
                <Attribut3>...</Attribut3>
                <Parameter1>...</Parameter1>
                <Parameter2>...</Parameter2>
                <Parameter3>...</Parameter3>
                ...
        </SpiderObjectName>
        <SpiderObjectName>
        ...
        </SpiderObjectName>
</ImportData>
```

1.1.1 Header

The XML structure starts with a header.

Specify the attribute encoding to determine the character set to be used. Make sure to use Unicode (UTF) for all imports described in this document:

```
<?xml version="1.0" encoding="UTF" ?>
```

Note If the attribute encoding is not specified, **UTF** (Unicode) will be adopted as standard.

1.1.2 Special characters

Special characters for element names have to be encoded in Unicode in the entire XML file. If the Unicode coding is incomplete, errors will occur during opening and importing. The special characters are:

- preceded by the Escape character "_" (Underscore),
- followed by an "x" as identifier for a hexadecimal, numerical value,
- complemented by a four-digit value (NNNN) and
- and terminated by the Escape character "_" (Underscore) again, which ends the so-called escape sequence.

This will result in the following format for special characters: "_xNNNN_". For the four-digit value (Unicode), please refer to the character map, which can be found under "Start/Accessories/System Utilities/Character Map". The Unicode is displayed in the format "U+NNNN" and has to be specified as "_xNNNN_" for the importer.

Below find some examples for converted special characters:

- Spaces to _x0020_
- & to _x0026_
- < to _x003C_
- > to _x003E_
- _ to _x005F_
- / to _x002F_

1.1.3 Date values

The format of date values is defined by specifying the culture in the configuration block <ImportConfig> . Typically used culture values for the import are: "de-DE", "en-GB" or "en-US".

Date values can also be entered in the format of an ISO 8601 standard. This is valid across cultures and does not only enable the input of a date, but also a time. For example, the time value "30.12.1978 13:45" according to **ISO standard ISO 8601** is written as **"1978-12-30T13:45"** or, using the time zone notation, as "1978-12-30T12:45 +1:00".

Example:

```
<ObjectName>
        <Field1>...</Field1>
        <Field2>...</Field2>
        <StartDate>2009-03-19T00:00:00</StartDate>
</ObjectName>
```

1.1.4 Decimal values

The format of decimal numbers must correspond to the culture specification from the configuration block *<ImportConfig>*.

1.1.5 Empty values / set NULL

To empty a field, you have to enter NULL.

Example:

```
<ObjectName>
        <Field1>...</Field1>
        <Field2>...</Field2>
        <PurchaseDate>NULL</PurchaseDate>
</ObjectName>
```

Note	Empty XML elements are ignored, so no import action takes place.

1.2 Importing objects

1.2.1 ImportConfig

Each import file can contain a configuration block which can be used for setting an alternative configuration.

An import configuration block has the following structure:

```
<ImportConfig>
        <Mode>IMPORT</Mode>
        <Culture>de-DE</Culture>
</ImportConfig>
```

Key	Meaning	Default
Mode	Mode sets the ImportMode for the whole file. However, this value can be overridden on each data **record** (see "**ImportMode**" on page 11).	IMPORT
Culture	Abbreviation for the usage of formats in region-dependent number formats and date formats.	de-DE

Note ImportConfig is optional. If the file to be imported does not contain a configuration block, *Culture* is taken from the application configuration (Application\Culture), and the import is carried out according to the *ImportConfiguration*.

1.2.2 ImportMode

The import mode determines the behavior for the import of each data record. By default, the import mode *Import* is used.

Modus	Meaning
Insert	Only Insert, no Update
Update	Only Update, no Insert
Import	Insert and Update (Standard)
Delete	Only Delete
Ignore	Data record is ignored

For each data record, the ImportMode can be set defiantly to the standard.

Example for Update:

```
<ImportData>
<Asset>
        <SCobjectName>Laptop</SCobjectName>
        <AssetTemplateName>Laptop-7</AssetTemplateName>
        <MandatorName>Standard</MandatorName>
        <AssetNo>12744c36-dd99-4f4b-b606-2178107e5d9e</AssetNo>
        <AssetStatusName>active</AssetStatusName>
        <ImportMode>Update</ImportMode>
</Asset>
</ImportData>
```

For the following data record, the default value or the value from ImportConfig is used again.

1.2.3 Attribute fields

A set of attribute fields is determined for each Spider object. In addition to the primary keys, alternative key fields, external key fields and other maintained fields are possible. An overview of the attribute fields of each object is not described in this document. Use the **Spider Admin-Tool** to find them in the corresponding **ObjectField definition**. Make sure to use the correct spelling when carrying out an import.

1.2.4 Parameter fields

The Spider objects configuration model enables the definition of parameter fields, which can be set for each object. In order to define an object parameter (object definition), a field name has to be set, which differs from the attribute names and which is unique for the object. This name has to used for the import. The correct spelling must be observed and can be taken from the **Spider Admin-Tool**.

1.2.5 Overwriting automatic values when creating

Automatic values can be configured per Spider object for an attribute. A new unique value is created based on a stored automatism when creating a new parameter.

In the configuration (Config) of Spider Importer the configuration key *AutoIdentifier*. *Enforce* can be used to determine, whether automatic values which have been set during the creation may be overwritten with the settings from the import XML file (Enforce=False). Otherwise (Enforce=True), the set automatic values are generated and possible settings from the import XML file are ignored.

1.2.6 Primärschlüssel

Ein Primärschlüssel ist eine Spalte (=einfacher Schlüssel) oder eine Gruppe von Spalten (=zusammengesetzter Schlüssel) und dient der eindeutigen Definierung eines Datensatzes/Zeile in einer Tabelle.

Es ist nicht möglich, die als Primärschlüssel gekennzeichneten Felder mit dem Importer zu verändern (aktualisieren). Das liegt daran, dass diese Felder für die Identifizierung des Objektes verwendet werden.

1.2.7 Alternative key

An alternative key is a field or a group of fields which can be used to resolve the primary key of an object/data record. Since the primary key of an object is mostly unknown, the alternative keys can be used to identify an object. The alternative keys are defined for each import object. The resolution of an object is only done if an object can be uniquely identified by combining the specified alternative keys. If several alternative key fields are available, not all must be used for the identification. Often, these are alternatives or fields for a further limitation in order to reach uniqueness. The meaning of the alternative keys is described for each object in this document.

Note	Alternative key fields can be modified with the importer only if the identification of the import objects is done via the primary key!

1.2.8 Mandatory fields

- Mandatory fields have to be observed for the import.
- Mandatory fields can be defined in the scope of the object definition. This is also true for parameters.
- The content of the mandatory fields can be seen using the Spider Admin-Tool.

1.2.9 System fields

Spider system fields usually start with the prefix "sys". These fields cannot be overwritten because they are filled in by the business logic.

Note	Calculated fields cannot be overwritten, either.

1.2.10 Resolving objects

To resolve object references which are saved in attribute fields, their alternative keys can be entered. The target is to determine the primary key of the referenced object (mostly ID) in order to save it as external key.

We distinguish between *Lookup* and *Preceding Import*:

- For the **Lookup**, the primary key of an already available object is determined. If no object is found, the import is canceled and a corresponding error message is displayed.
- For the **preceding import**, however, a new object can be created beforehand. To do this, all required values must be resolved for the preceding import.

If an object shall be precedingly imported or referenced, the alternative key fields are specified with preceding object names.

Example:

```
<ObjectNameID>1</ObjectNameID>
```

To be resolved by the alternative field name Identifier:

```
<ObjectNameIdentifier>DN00001</ObjectNameIdentifier>
```

Note	The Importer runs with more performance if the primary key is used instead of alternative key fields.

1.2.11 Abbreviations and rules

The following chapters show the Spider objects and their fields, which can be imported. The name of an import object refers to the corresponding table name in the database. There can be other identifiers in the web front end. For the description of the fields, the following abbreviations are used in this document:

Abbreviation	Name	Meaning
PI	PreceedingImport	Preceding Import of objects. These are carried out before the actual import.
LU	LookUp	Lookup objects resolve primary keys of objects by using alternative key fields. They are useful for import objects to make it unnecessary to enter external keys such as IDs or GUIDs. Lookup objects are only designed for the resolution of the primary keys and cannot be changed. They have no functionality for the import of new data or the update of already available data.
PK	PrimaryKeyNames	Primary key
AK	AlternativeKeyNames	Alternative key

Note

In this documentation, only the primary keys, the alternative keys, and the preceding import objects are displayed. More detailed Information about the (attributes and parameters) of the import objects can be seen using the Spider **Admin-Tool** from the **Object definition**.

CHAPTER 2

Spider Core

In this chapter

2.1 Spider Core import objects

2.1.1 CostCentre

The object **CostCentre** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>CostCentre</Name>
<Public>true</Public>
<TableName>CostCentre</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,ProductArea,AccountingArea,FullName,
MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName>
        <LookUp>false</LookUp>
        <Insert>true</Insert>
        <Update>true</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name **ProductArea** **AccountingArea**	In principle, these three fields are required for the identification of a CostCentre object. If the fields in *ProductArea* and *AccountingArea* are not used in Spider, it is sufficient to identify the CostCentre object via name.
FullName	This field is a combination of the following fields: *<Name> <ProductArea> <AccountingArea>*
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
EmployeeID	**Employee**	DomainLogin StaffNo ExternalReference FullName

Example for a CostCentre import file:

```
<ImportData>
<CostCentre>
        <MandatorID>0</MandatorID>
        <Name>Productizing</Name>
        <AccountingArea>400</AccountingArea>
        <Comment>Testdata</Comment>
</CostCentre>
<CostCentre>
        <MandatorID>0</MandatorID>
        <Name>Sales &amp;amp; Marketing</Name>
        <AccountingArea>307</AccountingArea>
        <Comment>Testdata</Comment>
</CostCentre>
</ImportData>
```

2.1.2 Document

Purpose: Import of files as document (attachment) to a Spider object. To do this, the Spider object has to be resolved via *GenericReference*. Depending on the setting, the documents are either loaded into the database or saved into a folder in the file system. The setting can be done in the application configuration.

Section: Spider.Objects.**Kernel.Document**

Key: **UseStorage**

Value: **TRUE / FALSE**

The recommended setting here is the database storage (value = TRUE).

Documents are administered in a document file. This document file contains all documents and is structured in folders.

The object **Document** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Document</Name>
<Public>true</Public>
<TableName>Document</TableName>
<AlternativeKeyNames>Name,SCObjectID,ObjectID,MandatorID</AlternativeKeyNames>
<ImportType>Document</ImportType>
<PreceedingImport>
        <DependantObjectName>GenericReference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Folder</DependantObjectName>
        <LookUp>false</LookUp>
        <Insert>true</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The identifier of the document (Document) can be resolved via the *Name* field.
SCObjectID **ObjectID**	SCObjectID and ObjectID resolve the Spider object, in which the document is to be imported. For resolving a Spider object, a special import object is **available** (see "**GenericReference**" on page 31). SCObjectID is resolved via the object name of the Spider object. ObjectID is resolved in combination with the ObjectIdentifier. You always have to specify a combination.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolve by
GenericReferenceID	**GenericReference**	ObjectName Identifier MandatorName
FolderID	**Folder**	FolderPath

Example of a document import:

```
<ImportData>
<!-- The example employee has to be imported prior to this import -->
<Document>
        <MandatorName>Standard</MandatorName>
        <ObjectName>Employee</ObjectName>
        <ObjectIdentifier>Smith, John (JohSm)</ObjectIdentifier>
        <Name>Office</Name>
        <Filename>Copy-In.doc</Filename>
        <FolderPath>eDoc/Copies</FolderPath>
</Document>
</ImportData>
```

Description:

- Documents may be attached to all Spider objects, for which the ObjectProperty "CanHaveDocuments" is set to active.
- Objectname and ObjectIdentifier are used to resolve the Spider object for the document (context)
- The FileName refers to the file, which is to be saved as document in Spider. The file path can be specified in absolute or relative form (to the import folder).
- Document folders (FolderPath) are created.
- If a document is moved to another folder, the old folder will be kept.
- The maximum upload size is determined in the web configuration.
- Version management: A new version will only be created if there is another file or the content of the existing file has been changed. If, for example, only the file name has been changed, no new version will be created. In this case, only the file name will be updated, and no further document will be created.

2.1.3 Employee

The object **Employee** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Employee</Name>
<Public>true</Public>
<TableName>Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,DomainLogin,FullName,MandatorID,ExternalReference</Alternati
veKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>EmployeeStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>CostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Currency</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
ExternalReference	In Spider Core, the employee can be resolved via the field *ExternalReference* , as long as this is kept **up-to-date** (see "**EmployeeExternalReference**" on page 20).

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
EmployeeStatusID	**EmployeeStatus**	EmployeeStatusName
CostCentreID	**CostCentre**	CostCentreName CostCentreAccountigArea CostCentreProductArea or CostCentreFullName
CurrencyID	**Currency**	CurrencyName

Example of an EmployeeExternalReference import files:

```
<ImportData>
<Employee>
        <MandatorID>0</MandatorID>
        <ExternalReference>ExtRefSmith</ExternalReference>
        <DomainLogin>JohSm</DomainLogin>
        <Lastname>Smith</Lastname>
        <Firstname>John</Firstname>
        <Title>COO</Title>
        <EmployeeStatusID>1</EmployeeStatusID>
        <EmploymentTypeID>1</EmploymentTypeID>
        <StaffNo>E1000</StaffNo>
        <EmailAddress>John.Smith@domain.corp</EmailAddress>
        <PhoneNo>+41 41 748 22 xx</PhoneNo>
        <MobilePhoneNo>+41 79 xxx xx xx</MobilePhoneNo>
        <DateOfEntry>2010-02-01T00:00:00</DateOfEntry>
        <Comment>Testdata</Comment>
</Employee>
</ImportData>
```

2.1.4 EmployeeExternalReference

Purpose: The object **EmployeeExternalReference** supports the matching of the employee master data with a leading third party system. The field ExternalReference saves the ID or the key of the employee data record in the source system. In contrast to the employee import object, the employee data record is identified exclusively via the field ExternalReference. All other fields can be updated; this applies also to the alternative key fields which are defined in the employee import object, e.g. StaffNo or DomainLogin.

The object **EmployeeExternalReference** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>EmployeeExternalReference</Name>
<Public>true</Public>
<TableName>Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>ExternalReference,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>EmployeeStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>CostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Currency</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
ExternalReference	Employee can be resolved via the field *ExternalReference*.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
EmployeeStatusID	**EmployeeStatus**	EmployeeStatusName
CostCentreID	**CostCentre**	CostCentreName CostCentreAccountigArea CostCentreProductArea or CostCentreFullName
CurrencyID	**Currency**	CurrencyName

Example of an EmployeeExternalReference import files:

```
<ImportData>
<EmployeeExternalReference>
        <MandatorID>0</MandatorID>
        <Firstname>Johnathan</Firstname>
        <EmployeeExternalReference>ExtRefSmith</EmployeeExternalReference>
</EmployeeExternalReference>
</ImportData>
```

2.1.5 EmployeeLogin

The object **EmployeeLogin** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>EmployeeLogin</Name>
<Public>true</Public>
<TableName>EmployeeLogin</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>LoginName</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Currency</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
LoginName	An employee login (EmployeeLogin) can be resolved via the field *LoginName*, as long as this is kept up-to-date. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
EmployeeID	**Employee**	EmployeeDomainLogin EmployeeStaffNo EmployeeExternalReference EmployeeFullName
CurrencyID	**Currency**	CurrencyName

Example of an EmployeeLogin import files:

```
<ImportData>
<EmployeeLogin>
        <EmployeeDomainLogin>JohSm</EmployeeDomainLogin>
        <LoginName>VPN00012</LoginName>
        <Description>Testdata</Description>
</EmployeeLogin>
</ImportData>
```

2.1.6 LegalEntity

The object **LegalEntity** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LegalEntity</Name>
<Public>true</Public>
<TableName>LegalEntity_base_XL</TableName>
<PrimaryKeyNames>MandatorID,ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,Name,ParentID,Path</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Parent</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The identification via the name does not have to be unique, since usually there are more than one legal entity by the same name. In order to identify them, the superordinated path (ParentID) must be specified.
ParentID	The ID of a superordinated object (ParentID) can be resolved via the ParentPath. This is used in combination with the name in order to create a legal entity and also for identification.
Path	The path can be used only for identification, but not for creating a new legal entity! The path is a combination of superordinated path (ParentID) and the name. Each component is separated by a "/" sign.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ParentID	**Parent**	ParentPath

Example of a LegalEntity import file:

```
<ImportData>
<LegalEntity>
        <MandatorName>Standard</MandatorName>
        <Name>Brainwaregroup</Name>
        <Description>Root legal entity</Description>
</LegalEntity>
<LegalEntity>
        <MandatorName>Standard</MandatorName>
        <ParentPath>Brainwaregroup</ParentPath>
        <Name>Spider LCM GmbH</Name>
        <Description>Spider LCM</Description>
</LegalEntity>
<LegalEntity>
        <MandatorName>Standard</MandatorName>
        <ParentPath>Brainwaregroup/Spider LCM GmbH</ParentPath>
```

```
        <Name>Germany</Name>
        <Description>Spider LCM</Description>
</LegalEntity>
<LegalEntity>
        <MandatorName>Standard</MandatorName>
        <ParentPath>Brainwaregroup</ParentPath>
        <Name>Brainware</Name>
        <Description>Brainware</Description>
</LegalEntity>
</ImportData>
```

2.1.7 LegalEntityAuthorisation

Purpose: Import of permissions to a LegalEntity, either for a user or for a role. The permission is granted on the level of the selected legal entity and is valid for all subordinated legal entities.

The removal of a permission is done via the ImportMode **Delete** (see chapter "**ImportMode**" , page 11).

The object **LegalEntityAuthorisation** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LegalEntityAuthorisation</Name>
<Public>true</Public>
<TableName>LegalEntityAuthorisation_base_M</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,LegalEntityID,UserID,RoleID</AlternativeKeyNames>
<SharedColumns>ApplicationGUID</SharedColumns>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>LegalEntity</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Application</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>User</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Role</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
LegalEntityID	The field *LegalEntityID* is used to resolve the legal entity.
UserID*	The field *UserID* is used to resolve the user for which the permission is to be set at the legal entity or removed from it.
RoleID*	The field *RoleID* is used to resolve the role for which the permission is to be set at the legal entity or removed from it. In case of identical role names in the applications, for the sake of uniqueness, the ApplicationGUID has to be specified or resolved.

* Either the combination of LegalEntityID and UserID or LegalEntityID and RoleID are supported.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
LegalEntityID	**LegalEntity**	LegalEntityPath
UserID	**User**	UserLogin
RoleID	**Role**	RoleName ApplicationGUID*
ApplicationGUID*	**Application**	ApplicationCode or ApplicationName

* ApplicationGUID can be resolved via ApplicationCode or ApplicationName. The spelling must be observed.

Example of a LegalEntityAuthorisation import file:

```
<ImportData>
<!-- This import file requires the objects User and Usermandator to be imported first -->
<LegalEntityAuthorisation>
        <LegalEntityPath>Brainwaregroup</LegalEntityPath>
        <UserLogin>SPJohSm</UserLogin>
        <MandantorName>Standard</MandantorName>
</LegalEntityAuthorisation>
</ImportData>
```

2.2 Spider Core system objects

2.2.1 Currency

The object **Currency** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Currency</Name>
<Public>true</Public>
<TableName>Currency</TableName>
<PrimaryKeyNames>Code</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The currency can be resolved via the *Name* field.

Example of a Currency import file:

```
<ImportData>
<Currency>
        <Code>USD</Code>
        <Name>US Dollar</Name>
        <ExchangeRate>0.5</ExchangeRate>
</Currency>
</ImportData>
```

2.2.2 ParaValueDef

The object **ParaValueDef** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ParaValueDef</Name>
<Public>true</Public>
<TableName>ParaValueDef</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Key,Value</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Key	Use these two fields to resolve an entry from an individual value list (ParaValueDef).
Value	

Example of a ParaValueDef import file:

```
<ImportData>
<ParaValueDef>
        <Key>Financing type</Key>
        <Value>Leasing</Value>
        <Text>Leasing</Text>
        <Initial>False</Initial>
        <Active>True</Active>
        <Rank>5</Rank>
</ParaValueDef>
</ImportData>
```

2.2.3 User

The object **User** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>User</Name>
<Public>true</Public>
<TableName>User</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Login</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Menu</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Login	The user can be resolved via the field *Login* if this is kept up-to-date.

Field	DependantObjectName	AK / Resolution by
MenuID	**Menu**	MenuName

Example of a User import file:

```
<ImportData>
<!-- Caution: This import adds an user to your system -->
<User>
        <Login>SPJohSm</Login>
        <Password>securepassword1234</Password>
        <RealName>John Smith</RealName>
        <MenuName>Reference menu</MenuName>
        <EmployeeExternalReference>ExtRefSmith</EmployeeExternalReference>
        <MaintenanceUser>False</MaintenanceUser>
</User>
</ImportData>
```

2.2.4 UserEmployee

The object **UserEmployee** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>UserEmployee</Name>
<Public>true</Public>
<TableName>UserEmployee</TableName>
<PrimaryKeyNames>UserID,MandatorID</PrimaryKeyNames>
<AlternativeKeyNames></AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>User</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
UserID	**User**	UserLogin
EmployeeID	**Employee**	EmployeeDomainLogin EmployeeStaffNo EmployeeExternalReference EmployeeFullName

Example of a UserEmployee import file:

```
<ImportData>
<UserEmployee>
        <MandatorName>Standard</MandatorName>
        <UserLogin>SPJohSm</UserLogin>
        <EmployeeFullname>Smith, John</EmployeeFullName>
</UserEmployee>
</ImportData>
```

2.2.5 UserMandator

The object **UserMandator** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>UserMandator</Name>
<Public>true</Public>
<TableName>UserMandator</TableName>
<PrimaryKeyNames>UserID,MandatorID</PrimaryKeyNames>
<AlternativeKeyNames></AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>User</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
UserID	**User**	UserLogin

Example of a UserMandator import file:

```
<ImportData>
<UserMandator>
        <MandatorName>Standard</MandatorName>
        <UserLogin>SPJohSm</UserLogin>
</UserMandator>
</ImportData>
```

2.2.6 UserRole

The object **UserRole** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>UserRole</Name>
<Public>true</Public>
<TableName>UserRole</TableName>
<PrimaryKeyNames>UserLogin,RoleName,ApplicationGUID</PrimaryKeyNames>
<AlternativeKeyNames></AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Application</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
ApplicationID	**Application**	ApplicationName ApplicationCode

Example of a UserRole import file:

```
<ImportData>
<UserRole>
        <UserLogin>SPJohSm</UserLogin>
        <RoleName>Basic</RoleName>
        <ApplicationName>Spider Core</ApplicationName>
</UserRole>
</ImportData>
```

2.3 Spider Core LookUp objects

2.3.1 Application

The Lookup Object **Application** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Application</Name>
<Public>false</Public>
<TableName>Application</TableName>
<PrimaryKeyNames>GUID</PrimaryKeyNames>
<AlternativeKeyNames>Name,Code</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The *ApplicationName* or the *ApplicationCode* can be used to resolve the field.
Code	

2.3.2 EmployeeStatus

The Lookup object **EmployeeStatus** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>EmployeeStatus</Name>
<Public>false</Public>
<TableName>EmployeeStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The status of an employee can be resolved via the *Name*

2.3.3 Folder

Zweck: Zugriff auf Verzeichnisse innerhalb einer Dokumentenakte eines Spider Objektes. Folder werden beim Importobjekt *Document* benötigt.

Das Lookup Objekt **Folder** wird nach folgenden Regeln aufgelöst:

Excerpt from the ImportConfiguration

```
Name>Folder</Name>
<Public>false</Public>
<TableName>Folder</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Path</AlternativeKeyNames>
<ImportType>Folder</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Beschreibung
Path	Ein Import wird über den Pfad identifiziert. Die einzelnen Bestandteile werden dabei mit einem „/" getrennt

2.3.4 GenericReference

Purpose: A Spider object is resolved via **GenericReference**. To do this, **Objectname** and **ObjectIdentifier** have to be specified. The **Objectname** is the name of the object class, e.g.: Employee, Contract, etc. The **ObjectIdentifier** is the identifier for resolving the object instance and depends on the object class.

The Lookup object **GenericReference** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>GenericReference</Name>
<Public>false</Public>
<TableName>sovObject</TableName>
<AlternativeKeyNames>ObjectName,Identifier,MandatorID</AlternativeKeyNames>
<ImportType>GenericReference</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
ObjectName	A general reference can be resolved using the field *Objectname*.
Identifier	A general reference can be resolved using the field *Identifier*.
MandatorID	The field filters the objects based on one mandator.

2.3.5 Mandator

The Lookup object **Mandator** is resolved according to the following rules:

```
<Name>Mandator</Name>
<Public>true</Public>
<TableName>Mandator</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The mandator is resolved via the *Name* field.

2.3.6 Menu

The Lookup object **Menu** is resolved according to the following rules:

```
<Name>Menu</Name>
<Public>false</Public>
<TableName>Menu</TableName>
<PrimaryKeyNames>GUID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The menu can be resolved via the *Name* field.

2.3.7 Parent

The Lookup object **Parent** resolves a superordinated LegalEntity.

Excerpt from the ImportConfiguration

```
<Name>Parent</Name>
<Public>false</Public>
<TableName>LegalEntity_base_XL</TableName>
<PrimaryKeyNames>MandatorID,ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,Path</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
MandatorID	A superordinated object (parent) can be resolved via the *MandatorID* or the field *Path* field.
Path	

2.3.8 Role

The Lookup object **Role** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Role</Name>
<Public>false</Public>
<TableName>Role</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,ApplicationGUID</AlternativeKeyNames>
<SharedColumns>ApplicationGUID</SharedColumns>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Application</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	Role can be resolved via the field *Name* in combination with the ApplicationGUID. ApplicationGUID is required if identical role names are used in the applications.
ApplicationGUID	

Field	DependantObjectName	AK / Resolution by
ApplicationID	**Application**	ApplicationName or ApplicationCode

CHAPTER 3

Spider Asset

In this chapter

3.1 Spider Asset import objects

3.1.1 Asset

The object **Asset** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Asset</Name>
<Public>true</Public>
<TableName>Asset</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>AssetNo,MandatorID,SCObjectID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AssetType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>LegalEntity</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AssetStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AssetTemplate</DependantObjectName>
        <LookUp>true</LookUp>
```

```
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>DeliveryNote</DependantObjectName>
        <LookUp>false</LookUp>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Location</DependantObjectName>
        <LookUp>false</LookUp>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AccountingApproach</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ServiceLevel</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>CostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>PurchaseCostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
AssetNo	An asset is resolved via the *AssetNo* (Asset Number) field.
SCObjectID	The field filters the assets based on one type (asset class).

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SCObjectID*	**AssetType**	AssetType
SCObjectID*	**SCObject**	SCObjectName
LegalEntityID	**LegalEntity**	LegalEntityPath
AssetStatusID	**AssetStatus**	AssetStatusName
AssetTemplateID	**AssetTemplate**	AssetTemplateName
DeliveryNoteID	**DeliveryNote**	DeliveryNoteIdentifier
LocationID	**Location**	LocationID
FunctionUnitID	**FunctionUnit**	FunctionUnitIdentifier FunctionUnitSCObjectName
AccountingApproachID	**AccountingApproach**	AccountingApproachName
ServiceLevelID	**ServiceLevel**	ServiceLevelName
CostCentreID	**CostCentre**	CostCentreName CostCentreAccountigArea CostCentreProductArea or CostCentreFullName
PurchaseCostCentreID	**PurchaseCostCentre**	PurchaseCostCentreName PurchaseCostCentreAccountigArea PurchaseCostCentreProductArea or PurchaseCostCentreFullName

* The SCObjectID field can be resolved either via *AssetType* or *SCObjectName*.

Example of an Asset import file:

```
<ImportData>
<Asset>
        <MandatorID>0</MandatorID>
        <AssetNo>DP000049</AssetNo>
        <AssetType>Desktop</AssetType>
        <AssetStatusName>Active</AssetStatusName>
        <HostName>npc0005</HostName>
        <MACAddress>01:02:03:04:05:06</MACAddress>
        <SerialNo>JJ4G02J</SerialNo>
        <InventoryNo>41453</InventoryNo>
        <PurchaseDate>2005-12-21T00:00:00</PurchaseDate>
        <BusinessValue>1400</BusinessValue>
        <WarrantyExpires>2008-12-21T00:00:00</WarrantyExpires>
        <DeliveryNoteIdentifier>LIEF0049</DeliveryNoteIdentifier>
        <DeliveryNoteOrderNo>BN08-030</DeliveryNoteOrderNo>
        <OrderDate>2002-06-03T00:00:00</OrderDate>
        <Comment>Testdata</Comment>
</Asset>
</ImportData>
```

3.1.2 DeliveryNote

The object **DeliveryNote** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>DeliveryNote</Name>
<Public>true</Public>
<TableName>DeliveryNote</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The DeliveryNote is resolved via the *Identifier* field.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

Example of a DeliveryNote import file:

```
<ImportData>
<DeliveryNote>
        <MandatorID>0</MandatorID>
        <Identifier>LIEF0001</Identifier>
        <Date>2004-06-18T00:00:00</Date>
        <Supplier>ARP</Supplier>
        <OrderDate>2004-06-03T00:00:00</OrderDate>
        <OrderNo>BN07-001</OrderNo>
</DeliveryNote>
</ImportData>
```

3.1.3 Document

Purpose: Import of files as document (attachment) to a Spider object. To do this, the Spider object has to be resolved via *GenericReference*. Depending on the setting, the documents are either loaded into the database or saved into a folder in the file system. The setting can be done in the application configuration.

Section: Spider.Objects.**Kernel.Document**

Key: **UseStorage**

Value: **TRUE / FALSE**

The recommended setting here is the database storage (value = TRUE).

Documents are administered in a document file. This document file contains all documents and is structured in folders.

The object **Document** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Document</Name>
<Public>true</Public>
<TableName>Document</TableName>
<AlternativeKeyNames>Name,SCObjectID,ObjectID,MandatorID</AlternativeKeyNames>
<ImportType>Document</ImportType>
<PreceedingImport>
        <DependantObjectName>GenericReference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Folder</DependantObjectName>
        <LookUp>false</LookUp>
        <Insert>true</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The identifier of the document (Document) can be resolved via the *Name* field.
SCObjectID **ObjectID**	SCObjectID and ObjectID resolve the Spider object, in which the document is to be imported. For resolving a Spider object, a special import object is **available** (see "**GenericReference**" on page 31). SCObjectID is resolved via the object name of the Spider object. ObjectID is resolved in combination with the ObjectIdentifier. You always have to specify a combination.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolve by
GenericReferenceID	**GenericReference**	ObjectName Identifier MandatorName
FolderID	**Folder**	FolderPath

Example of a document import:

```
<ImportData>
<!-- The imported document needs to be named Copy-In.doc and placed in the same Folder as this
import file. -->
<Document>
        <MandatorName>Standard</MandatorName>
        <ObjectName>DeliveryNote</ObjectName>
        <ObjectIdentifier>LIEF0001</ObjectIdentifier>
        <Name>Office</Name>
        <Filename>Copy-In.doc</Filename>
        <FolderPath>eDoc/Copies</FolderPath>
</Document>
</ImportData>
```

Description:

- Documents may be attached to all Spider objects, for which the ObjectProperty "CanHaveDocuments" is set to active.
- Objectname and ObjectIdentifier are used to resolve the Spider object for the document (context)
- The FileName refers to the file, which is to be saved as document in Spider. The file path can be specified in absolute or relative form (to the import folder).
- Document folders (FolderPath) are created.
- If a document is moved to another folder, the old folder will be kept.
- The maximum upload size is determined in the web configuration.
- Version management: A new version will only be created if there is another file or the content of the existing file has been changed. If, for example, only the file name has been changed, no new version will be created. In this case, only the file name will be updated, and no further document will be created.

3.1.4 FunctionUnit

The object **FunctionUnit** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>FunctionUnit</Name>
<Public>true</Public>
<TableName>FunctionUnit</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,MandatorID,SCObjectID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnitType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>LegalEntity</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>CostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnitStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AccountingApproach</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ServiceLevel</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Location</DependantObjectName>
        <LookUp>false</LookUp>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>OrganisationalUnit</DependantObjectName>
<LookUp>false</LookUp>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The FunctionUnit can be identified via the *Identifier* field. Usually, the field is defined as unique.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
SCObjectID	The field filters the FunctionUnits based on one type.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SCObjectID*	**SCObject**	SCObjectName
SCObjectID*	**FunctionUnitType**	FunctionUnitType
LegalEntityID	**LegalEntity**	LegalEntityPath
CostCentreID	**CostCentre**	CostCentreName CostCentreAccountigArea CostCentreProductArea or CostCentreFullName
FunctionUnitStatusID	**FunctionUnitStatus**	FunctionUnitStatusName
AccountingApproachID	**AccountingApproach**	AccountingApproachName
ServiceLevelID	**ServiceLevel**	ServiceLevelName
LocationID	**Location**	LocationID
OrganisationalUnitID	**OrganisationalUnit**	OrganisationalUnitID

* You can resolve the SCObjectID field either via the *FunctionUnitType* field or the *SCObject-Name* field.

Example of a FunctionUnit import file:

```
<ImportData>
<!-- To import this data correctly, ensure that the configuration autoidentifier.enforce is set to false -->
<FunctionUnit>
        <MandatorID>0</MandatorID>
        <Identifier>WP000001</Identifier>
        <FunctionUnitType>Client System</FunctionUnitType>
        <FunctionUnitStatusName>Active</FunctionUnitStatusName>
        <Label>Testdata</Label>
        <Comment>Testdata</Comment>
</FunctionUnit>
<FunctionUnit>
        <MandatorName>Standard</MandatorName>
        <Identifier>WP000002</Identifier>
        <FunctionUnitType>Client System</FunctionUnitType>
        <FunctionUnitStatusName>Active</FunctionUnitStatusName>
        <Label>Testdata</Label>
        <Comment>Testdata</Comment>
</FunctionUnit>
</ImportData>
```

3.1.5 FunctionUnitEmployee

Purpose: The object **FunctionUnitEmployee** is used to assign an employee to a function unit or to remove it

The object **FunctionUnitEmployee** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>FunctionUnitEmployee</Name>
<Public>true</Public>
<TableName>FunctionUnitEmployee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>FunctionUnitID,EmployeeID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
FunctionUnitID **EmployeeID**	These two fields are required for the identification of a FunctionUnitEmployee object. In order to resolve FunctionUnitID and EmployeeID, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
FunctionUnitID	**FunctionUnit**	FunctionUnitIdentifier FunctionUnitSCObjectName
EmployeeID	**Employee**	FunctionUnitEmployeeDomainLogin FunctionUnitEmployeeStaffNo FunctionUnitEmployeeFullName

Example of a FunctionUnitEmployee import file:

```
<ImportData>
<!-- This import file requires the Core.Employee import file to be imported first. -->
<FunctionUnitEmployee>
        <MandatorID>0</MandatorID>
        <FunctionUnitIdentifier>WP000001</FunctionUnitIdentifier>
        <EmployeeDomainLogin>JohSm</EmployeeDomainLogin>
        <Comment>Testdata</Comment>
</FunctionUnitEmployee>
</ImportData>
```

3.1.6 FunctionUnitRelationship

The object **FunctionUnitRelationship** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>FunctionUnitRelationship</Name>
<Public>true</Public>
<TableName>FunctionUnitRelationship</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,PrimaryFunctionUnitID,SecondaryFunctionUnitID,MandatorID,SCObje
ctID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnitRelationshipType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>PrimaryFunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SecondaryFunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	A FunctionUnitRelationship is resolved via the *Name* field.
PrimaryFunctionUnitID **SecondaryFunction-UnitID**	These two fields are required for the identification of a FunctionUnitRelationship object. For resolving primary and secondary function units, between which a relationship is to be established, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
SCObjectID	The field limits the number of FunctionUnitRelationships to the (FunctionUnitRelationships) of one type.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SCObjectID*	**SCObject**	SCObjectName
SCObjectID*	**FunctionUnitRelationship Type**	FunctionUnitRelationshipType
PrimaryFunctionUnitID	**PrimaryFunctionUnit**	PrimaryFunctionUnitIdentifier PrimaryFunctionUnitSCObjectName
SecondaryFunctionUnitID	**SecondaryFunctionUnit**	SecondaryFunctionUnitIdentifier SecondaryFunctionUnitSCObject Name

* You can resolve the SCObjectID field either via the *FunctionUnitRelationshipType* field or the *SCObjectName* field.

Example of a FunctionUnitRelationship import file:

```
<ImportData>
<FunctionUnitRelationship>
        <MandatorName>Standard</MandatorName>
        <SCObjectName>Depends on</SCObjectName>
        <Name>Depends000001</Name>
        <PrimaryFunctionUnitIdentifier>WP000001</PrimaryFunctionUnitIdentifier>
        <SecondaryFunctionUnitIdentifier>WP000002</SecondaryFunctionUnitIdentifier>
        <PrimaryFunctionUnitDependentDelete>1</PrimaryFunctionUnitDependentDelete>
        <SecondaryFunctionUnitDependentDelete>1</SecondaryFunctionUnitDependentDelete>
        <Comment>Testdata</Comment>
</FunctionUnitRelationship>
</ImportData>
```

3.1.7 FunctionUnitSoftwarePacket

Purpose: The object **FunctionUnitSoftwarePacket** is used to assign a software package to a function unit or to remove it.

The object **FunctionUnitSoftwarePacket** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>FunctionUnitSoftwarePacket</Name>
<Public>true</Public>
<TableName>FunctionUnitSoftwarePacket</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>FunctionUnitID,SoftwarePacketID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SoftwarePacket</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
FunctionUnitID **SoftwarePacketID**	These two fields are required for the identification of a FunctionUnitSoftwarePacket object. For resolving FunctionUnitID and SoftwarePacketID, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
FunctionUnitID	**FunctionUnit**	FunctionUnitIdentifier FunctionUnitSCObjectName
SoftwarePacketID	**SoftwarePacket**	SoftwarePacketFullName

Example of a FunctionUnitSoftwarePacket import file:

```
<ImportData>
<!-- This import requires the SoftwarePacket import file to be imported first -->
<FunctionUnitSoftwarePacket>
        <MandatorName>Standard</MandatorName>
        <FunctionUnitIdentifier>WP000001</FunctionUnitIdentifier>
        <SoftwarePacketFullName>Standardsoftware 1.0</SoftwarePacketFullName>
</FunctionUnitSoftwarePacket>
</ImportData>
```

3.1.8 NetworkInterface

The object **NetworkInterface** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>NetworkInterface</Name>
<Public>true</Public>
<TableName>NetworkInterface</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<!--    no alternative keys    -->
<AlternativeKeyNames></AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Asset</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
FunctionUnitID*	**FunctionUnit**	FunctionUnitIdentifier FunctionUnitSCObjectName
AssetID*	**Asset**	AssetAssetNo AssetSCObjectName

* You can resolve either the *AssetID* or the *FunctionUnitID* field.

Example of a NetworkInterface import file:

```
<ImportData>
<NetworkInterface>
        <MandatorName>Standard</MandatorName>
        <FunctionUnitIdentifier>WP000001</FunctionUnitIdentifier>
        <DHCP>True</DHCP>
        <IPAddress>192.168.143.7</IPAddress>
        <Domain>Green</Domain>
        <SubnetMask>255.255.255.0</SubnetMask>
        <SubnetName>DNS-Suffix</SubnetName>
        <DefaultGateway>192.168.35.254</DefaultGateway>
        <NetworkType>Ethernet</NetworkType>
        <Hostname>Sprinter458</Hostname>
        <Comment>Testdata</Comment>
</NetworkInterface>
</ImportData>
```

3.1.9 Resubmission

Purpose: Certain Spider objects can send resubmissions. To do this, the Spider object has to be resolved via GenericReference .

The object **Resubmission** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Resubmission</Name>
<Public>true</Public>
<TableName>Resubmission</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<!--    no alternative keys    -->
<AlternativeKeyNames></AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>GenericReference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
<Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SendToEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SendCcEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
SCObjectID ObjectID	**GenericReference**	ObjectName Identifier MandatorName
SendToEmployeeID	**SendToEmployee**	SendToEmployeeStaffNo SendToEmployeeDomainLogin SendToEmployeeFullName
SendCcEmployeeID	**SendCcEmployee**	SendCCEmployeeStaffNo SendCCEmployeeDomainLogin SendCCEmployeeFullName

Example of a Resubmission import file:

```
<ImportData>
<!-- This import file requires the Core.Employee import file to be imported first. -->
<Resubmission>
        <MandatorName>Standard</MandatorName>
        <ObjectName>FunctionUnit</ObjectName>
        <ObjectIdentifier>WP000001</ObjectIdentifier>
        <Date>2020-05-25T00:00:00</Date>
        <Designation>Test Designation 1</Designation>
        <Subject>Test subject 1</Subject>
        <Body>Here goes the text ...</Body>
        <SendToEmployeeStaffNo>E1000</SendToEmployeeStaffNo>
</Resubmission>
</ImportData>
```

3.1.10 Software

The object **Software** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Software</Name>
<Public>true</Public>
<TableName>Software</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,InvName,Version,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AccountingApproach</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	This field is required for the identification of a Software object.
InvName	The *InvName* field is required for resolving the software inventory name.
Version	The *Version* field is required for resolving the software version.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
AccountingApproachID	**AccountingApproach**	AccountingApproachName

Example of a Software import file:

```
<ImportData>
<Software>
        <MandatorID>0</MandatorID>
        <Name>Spider Contract</Name>
        <Version>6.1</Version>
        <Language>DE</Language>
        <Manufacturer>Brainwaregroup</Manufacturer>
        <InvName>SOCONTRACT_61_DE</InvName>
        <Active>True</Active>
        <Released>True</Released>
        <ServerSoftware>True</ServerSoftware>
        <LicenceCostFree>False</LicenceCostFree>
        <SoftwareCategory>No Category</SoftwareCategory>
        <OperatingSystem>Microsoft Windows Server</OperatingSystem>
        <OperatingSystemVersion>2008</OperatingSystemVersion>
        <Description>Testdata</Description>
</Software>
</ImportData>
```

3.1.11 SoftwareAssignment

The object **SoftwareAssignment** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SoftwareAssignment</Name>
<Public>true</Public>
<TableName>SoftwareAssignment</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>SoftwareID,AssetID,FunctionUnitID,MandatorID,EmployeeID,Remote,Remot
eServer</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Software</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Asset</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>LegalEntity</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
SoftwareID	These four fields are required for the identification of a Software-Assignment object. For resolving SoftwareID, AssetID, EmployeeID, and FunctionUnitID, the corresponding alternative keys are available. Software+Asset / Software+Employee / Software+Function unit
AssetID	
FunctionUnitID	
EmployeeID	
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
Remote	0 = local / 1 = remote
RemoteServer	If Remote exists, the service can be specified here.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SoftwareID	**Software**	SoftwareName SoftwareVersion or SoftwareInvName
AssetID*	**Asset**	AssetAssetNo AssetSCObjectName
FunctionUnitID*	**FunctionUnit**	FunctionUnitIdentifier FunctionUnitSCObjectName
EmployeeID*	**Employee**	EmployeeDomainLogin EmployeeStaffNo EmployeeFullName
LegalEntityID	**LegalEntity**	LegalEntityPath

* You can resolve either the *AssetID EmployeeID* or the *FunctionUnitID* field.

Example of a SoftwareAssignment import file:

```
<ImportData>
<SoftwareAssignment>
        <MandatorID>0</MandatorID>
        <SoftwareInvName>SOCONTRACT_61_DE</SoftwareInvName>
        <AssetAssetNo>DP000049</AssetAssetNo>
</SoftwareAssignment>
</ImportData>
```

3.1.12 SoftwarePacket

The object **SoftwarePacket** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SoftwarePacket</Name>
<Public>true</Public>
<TableName>SoftwarePacket</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>FullName,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
FullName	This field is required for the identification of a SoftwarePacket object.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

Example of a SoftwarePacket import file:

```
<ImportData>
<SoftwarePacket>
        <MandatorName>Standard</MandatorName>
        <Name>Standardsoftware</Name>
        <VersionMajor>1</VersionMajor>
        <VersionMinor>0</VersionMinor>
        <Description>Standard Softwarepacket</Description>
        <Active>True</Active>
        <Released>True</Released>
</SoftwarePacket>
</ImportData>
```

3.1.13 SoftwarePacketSoftware

The object **SoftwarePacketSoftware** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SoftwarePacketSoftware</Name>
<Public>true</Public>
<TableName>SoftwarePacketSoftware</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>SoftwareID,SoftwarePacketID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Software</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SoftwarePacket</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
SoftwareID **SoftwarePacketID**	These two fields can be used for the identification of a Software-PacketSoftware object. For resolving SoftwareID and SoftwarePacketID, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SoftwareID	**Software**	SoftwareName SoftwareVersion or SoftwareInvName
SoftwarePacketID	**SoftwarePacket**	SoftwarePacketFullName

Example of a SoftwarePacketSoftware import file:

```
<ImportData>
<SoftwarePacketSoftware>
        <MandatorName>Standard</MandatorName>
        <Name>Spider Contract Pack</Name>
        <SoftwareInvName>SOCONTRACT_61_DE</SoftwareInvName>
        <VersionMajor>1</VersionMajor>
        <VersionMinor>0</VersionMinor>
        <Description>Server Produkt Bundle</Description>
        <Active>TRUE</Active>
        <SoftwarePacketFullName>Standardsoftware 1.0</SoftwarePacketFullName>
</SoftwarePacketSoftware>
</ImportData>
```

3.1.14 AssetTemplate

The object **AssetTemplate** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AssetTemplate</Name>
<Public>true</Public>
<TableName>Asset</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,MandatorID,SCObjectID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AssetType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AccountingApproach</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ServiceLevel</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	An AssetTemplate is resolved via the *Name* field.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
SCObjectID	The field filters the AssetTemplates based on one type (asset class)

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
AssetTypeID*	**AssetType**	AssetType
SCObjectID*	**SCObject**	SCObjectName
AccountingApproachID	**AccountingApproach**	AccountingApproachName
ServiceLevelID	**ServiceLevel**	ServiceLevelName

* The SCObjectID field can be resolved either via *AssetType* or *SCObjectName*.

Example of a AssetTemplate import file:

```
<ImportData>
<AssetTemplate>
        <MandatorID>0</MandatorID>
        <AssetType>Desktop</AssetType>
        <Name>PC  DL 300</Name>
        <RAMInMb>1024</RAMInMb>
        <DiskCount>2</DiskCount>
        <DiskTotalInGb>512</DiskTotalInGb>
        <CPUSpeedInMhz>3000</CPUSpeedInMhz>
        <CPUCount>2</CPUCount>
        <CorePerCPU>2</CorePerCPU>
        <CPUType>X86</CPUType>
</AssetTemplate>
</ImportData>
```

3.2 Spider Asset inventory objects

3.2.1 AssetInventory

The object **AssetInventory** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AssetInventory</Name>

<Public>true</Public>
<TableName>Asset</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>AssetNo,MandatorID</AlternativeKeyNames>
<ImportType>Inventory</ImportType>
<InventoryInsert>false</InventoryInsert>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
AssetNo	You can use the *AssetNo* field for the identification of an Asset object. For resolving an AssetID, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

Example of a AssetInventory import file:

```
<ImportData>
<AssetInventory>
        <MandatorName>Standard</MandatorName>
        <AssetNo>DP000049</AssetNo>
        <invLastScanDate>2014-01-01T00:00:00</invLastScanDate>
        <invMessage>Testdata</invMessage>
</AssetInventory>
</ImportData>
```

3.2.2 AssetInventoryParameter

The object **AssetInventoryParameter** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AssetInventoryParameter</Name>
<Public>true</Public>
<TableName>AssetInventoryParameter</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>AssetID,Name</AlternativeKeyNames>
<ImportType>InventoryParameter</ImportType>
<PreceedingImport>
        <DependantObjectName>Asset</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
AssetID	You can use the *AssetID* field for the identification of an Asset object. For resolving an *AssetID*, the corresponding alternative keys are available.
Name	An AssetInventoryParameter is resolved via the *Name* field.

Field	DependantObjectName	AK / Resolution by
AssetID	**Asset**	AssetAssetNo AssetSCObjectName

Example of an AssetInventoryParameter import file:

```
<ImportData>
<AssetInventoryParameter>
        <AssetAssetNo>DP000049</AssetAssetNo>
        <Name>Para01</Name>
        <Value>000000002</Value>
        <CultureCode>de-DE</CultureCode>
        <InvExists>1</InvExists>
</AssetInventoryParameter>
</ImportData>
```

3.2.3 EmployeeInventory

The object **EmployeeInventory** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>EmployeeInventory</Name>
<Public>true</Public>
<TableName>FunctionUnitEmployee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>FunctionUnitID,EmployeeID</AlternativeKeyNames>
<ImportType>Inventory</ImportType>
<InventoryInsert>true</InventoryInsert>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
FunctionUnitID **EmployeeID**	These two fields are required for the identification of a FunctionUnitEmployee object. In order to resolve FunctionUnitID and EmployeeID, the corresponding alternative keys are available.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
FunctionUnitID	**FunctionUnit**	FunctionUnitIdentifier FunctionUnitSCObjectName
EmployeeID	**Employee**	FunctionUnitEmployeeDomainLogin FunctionUnitEmployeeStaffNo FunctionUnitEmployeeFullName

Example of an EmployeeInventory import file:

```
<ImportData>
<!-- This import file requires the Core.Employee import file to be imported first. -->
<EmployeeInventory>
        <MandatorName>Standard</MandatorName>
        <FunctionUnitIdentifier>WP000001</FunctionUnitIdentifier>
        <EmployeeDomainLogin>JohSm</EmployeeDomainLogin>
        <invLastScanDate>2014-01-01T00:00:00</invLastScanDate>
</EmployeeInventory>
</ImportData>
```

3.2.4 SoftwareInventory

The object **SoftwareInventory** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SoftwareInventory</Name>
<Public>true</Public>
<TableName>SoftwareAssignment</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>SoftwareID,AssetID,FunctionUnitID,MandatorID,EmployeeID,Remote,Remot
eServer</AlternativeKeyNames>
<ImportType>Inventory</ImportType>
<InventoryInsert>true</InventoryInsert>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Software</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Asset</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
SoftwareID	These four fields are required for the identification of a Software-Assignment object. For resolving SoftwareID, AssetID, EmployeeID, and FunctionUnitID, the corresponding alternative keys are available. Software+Asset / Software+Employee / Software+Function unit
AssetID	
FunctionUnitID	
EmployeeID	
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
Remote	0 = local / 1 = remote
RemoteServer	If Remote exists, the service can be specified here.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SoftwareID	**Software**	SoftwareName SoftwareVersion or SoftwareInvName
AssetID*	**Asset**	AssetAssetNo AssetSCObjectName
FunctionUnitID*	**FunctionUnit**	FunctionUnitIdentifier FunctionUnitSCObjectName
EmployeeID*	**Employee**	EmployeeDomainLogin EmployeeStaffNo EmployeeFullName

* You can resolve either the *AssetID, EmployeeID* or the *FunctionUnitID* field.

Example of a SoftwareInventory import file:

```
<ImportData>
<SoftwareInventory>
        <MandatorName>Standard</MandatorName>
        <FunctionUnitIdentifier>WP000001</FunctionUnitIdentifier>
        <SoftwareInvName>SOCONTRACT_61_DE</SoftwareInvName>
        <invLastScanDate>2014-01-01T00:00:00</invLastScanDate>
        <invMessage>Testdata</invMessage>
</SoftwareInventory>
</ImportData>
```

3.3 Spider Asset system objects

3.3.1 AssetStatus

The object **AssetStatus** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AssetStatus</Name>
<Public>true</Public>
<TableName>AssetStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	An AssetStatus is resolved via the *Name* field.

Example of an AssetStatus import file:

```
<ImportData>
<AssetStatus>
        <Name>Ordered</Name>
        <Initial>True</Initial>
        <Unknown>False</Unknown>
        <RequireFuntionUnit>False</RequireFuntionUnit>
        <DetachFuntionUnit>False</DetachFuntionUnit>
        <Active>True</Active>
        <Archive>False</Archive>
        <Description>Testdata</Description>
</AssetStatus>
<AssetStatus>
        <Name>Replacement</Name>
        <Initial>False</Initial>
        <Unknown>False</Unknown>
        <RequireFuntionUnit>False</RequireFuntionUnit>
        <DetachFuntionUnit>False</DetachFuntionUnit>
        <Active>True</Active>
        <Archive>False</Archive>
        <Description>Testdata</Description>
</AssetStatus>
</ImportData>
```

3.3.2 AssetStatusSequence

The object **AssetStatusSequence** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AssetStatusSequence</Name>
<Public>true</Public>
<TableName>AssetStatusSequence</TableName>
<PrimaryKeyNames>AssetStatusID,NextAssetStatusID,SCObjectID</PrimaryKeyNames>
<AlternativeKeyNames></AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>AssetType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AssetStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>NextAssetStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
SCObjectID*	**AssetType**	AssetType
SCObjectID*	**SCObject**	SCObjectName
AssetStatusID	**AssetStatus**	AssetStatusName
NextAssetStatusID	**NextAssetStatus**	NextAssetStatusName

* The asset class can either be resolved via the *AssetType* or the *SCObjectName* field.

Example of an AssetStatusSequence import file:

```
<ImportData>
<AssetStatusSequence>
        <AssetStatusName>Ordered</AssetStatusName>
        <NextAssetStatusName>Replacement</NextAssetStatusName>
        <SCObjectName>Server</SCObjectName>
        <Rank>11</Rank>
</AssetStatusSequence>
</ImportData>
```

3.3.3 Currency

The object **Currency** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Currency</Name>
<Public>true</Public>
<TableName>Currency</TableName>
<PrimaryKeyNames>Code</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The currency can be resolved via the *Name* field.

Example of a Currency import file:

```
<ImportData>
<!-- Caution! This import will delete the currency code EUR from your system! -->
<Currency>
        <ImportMode>Delete</ImportMode>
        <Code>EUR</Code>
        <Name>EURO</Name>
        <ExchangeRate>0.5</ExchangeRate>
</Currency>
</ImportData>
```

3.3.4 AccountingApproach

The object **AccountingApproach** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AccountingApproach</Name>
<Public>true</Public>
<TableName>AccountingApproach</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,SCObjectID</AlternativeKeyNames>
<SharedColumns>SCObjectID</SharedColumns>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	AccountingApproach can be resolved via the *Name* field.
SCObjectID	AccountingApproach is administered depending on the asset class, the function unit type or the software. For resolving the asset class, the function unit type or the software, the special import object SCObject is available.

Field	DependantObjectName	AK / Resolution by
SCObjectID	**SCObject**	SCObjectName

Example of an AccountingApproach import file:

```
<ImportData>
<AccountingApproach>
        <SCObjectName>Server</SCObjectName>
        <Name>AccountingApproach_01</Name>
        <Expired>FALSE</Expired>
        <Description>Testdata</Description>
</AccountingApproach>
</ImportData>
```

3.3.5 FunctionUnitStatus

The object **FunctionUnitStatus** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>FunctionUnitStatus</Name
<Public>true</Public>
<TableName>FunctionUnitStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The FunctionUnitStatus is resolved via the *Name* field.

Example of a FunctionUnitStatus import file:

```
<ImportData>
<FunctionUnitStatus>
        <Name>In planning</Name>
        <Initial>TRUE</Initial>
        <Active>TRUE</Active>
        <Archive>FALSE</Archive>
        <Description>Testdata</Description>
</FunctionUnitStatus>
</ImportData>
```

3.3.6 ParaValueDef

The object **ParaValueDef** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ParaValueDef</Name>
<Public>true</Public>
<TableName>ParaValueDef</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Key,Value</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Key	Use these two fields to resolve an entry from an individual value list (ParaValueDef).
Value	

Example of a ParaValueDef import file:

```
<ImportData>
<ParaValueDef>
        <Key>accounting approach</Key>
        <Value>Leasing</Value>
        <Text>Leasing</Text>
        <Initial>False</Initial>
        <Active>True</Active>
        <Rank>5</Rank>
</ParaValueDef>
</ImportData>
```

3.3.7 ServiceLevel

The object **ServiceLevel** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ServiceLevel</Name>
<Public>true</Public>
<TableName>ServiceLevel</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>SCObjectID,Name</AlternativeKeyNames>
<SharedColumns>SCObjectID</SharedColumns>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The ServiceLevel can be resolved via the *Name* field.
SCObjectID	ServiceLevel is administered depending on the asset class or the function unit type. For resolving the asset class or the function unit type, the special import object SCObject is available.

Field	DependantObjectName	AK / Resolution by
SCObjectID	**SCObject**	SCObjectName

Example of a ServiceLevel import file:

```
<ImportData>
<ServiceLevel>
        <SCObjectName>Desktop</SCObjectName>
        <Name>immediate reaction</Name>
        <Expired>FALSE</Expired>
        <Description>response within 2 hours</Description>
</ServiceLevel>
</ImportData>
```

3.3.8 Location

The object **Location** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Location</Name>
<Public>true</Public>
<TableName>Location</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,HierarchyPath</AlternativeKeyNames>
<ImportType>Hierarchy</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
HierarchyPath	The field filters the *Location* using the hierarchy path.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

Example of a Location import file:

```
<ImportData>
<Location>
        <MandatorID>0</MandatorID>
        <Region>Germany</Region>
        <City>Hamburg</City>
        <Building>Paul-Dessau-Strasse 8</Building>
        <Floor>3. OG</Floor>
        <Room>Consulting</Room>
        <Description>Testdata</Description>
</Location>
</ImportData>
```

3.3.9 OrganisationalUnit

The object **OrganisationalUnit** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>OrganisationalUnit</Name>
<Public>true</Public>
<TableName>OrganisationalUnit</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,HierarchyPath</AlternativeKeyNames>
<ImportType>Hierarchy</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
HierarchyPath	The field filters the OrganisationalUnit using the hierarchy path.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

Example of an OrganisationalUnit import file:

```
<ImportData>
<OrganisationalUnit>
        <MandatorID>0</MandatorID>
        <Level01>BrainwareGroup</Level01>
        <Level02>Spider Lifecycle Managementsysteme GmbH</Level02>
        <Level03></Level03>
        <Level04></Level04>
        <Level05></Level05>
</OrganisationalUnit>
</ImportData>
```

3.4 Spider Asset lookup objects

3.4.1 AssetType

The Lookup object **AssetType** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AssetType</Name>
<Public>false</Public>
<TableName>Asset</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<ImportType>TypeLookUp</ImportType>
```

Note	This Lookup object is used to determine the SCObjectID for an AssetType. An alternative key does not need to be entered.

3.4.2 CostCentre

The Lookup object **CostCentre** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>CostCentre</Name>
<Public>false</Public>
<TableName>Core_CostCentre</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,ProductArea,AccountingArea,FullName,MandatorID</AlternativeKeyN
ames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name **ProductArea** **AccountingArea**	In principle, these three fields are required for the identification of a CostCentre object. If the fields ProductArea and AccountingArea are not used in Spider, it is sufficient to identify the CostCentre object via name.
FullName	This field is a combination of the following fields: *<Name> <ProductArea> <AccountingArea>*
MandatorID	The field filters the objects based on one mandator.

3.4.3 PurchaseCostCentre

The Lookup object **PurchaseCostCentre** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>PurchaseCostCentre</Name>
<Public>false</Public>
<TableName>Core_CostCentre</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,ProductArea,AccountingArea,FullName,MandatorID</AlternativeKeyN
ames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name **ProductArea** **AccountingArea**	In principle, these three fields are required for the identification of a CostCentre object. If the fields *ProductArea and AccountingArea* are not used in Spider, it is sufficient to identify the CostCentre object via *Name.*
FullName	This field is a combination of the following fields: *<Name> <ProductArea> <AccountingArea>*
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

3.4.4 Employee

The Lookup object **Employee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Employee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

3.4.5 SendToEmployee

The Lookup object **SendToEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SendToEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

3.4.6 SendCcEmployee

The Lookup object **SendCcEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SendCcEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

3.4.7 Folder

Zweck: Zugriff auf Verzeichnisse innerhalb einer Dokumentenakte eines Spider Objektes. Folder werden beim Importobjekt *Document* benötigt.

Das Lookup Objekt **Folder** wird nach folgenden Regeln aufgelöst:

Excerpt from the ImportConfiguration

```
Name>Folder</Name>
<Public>false</Public>
<TableName>Folder</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Path</AlternativeKeyNames>
<ImportType>Folder</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Beschreibung
Path	Ein Import wird über den Pfad identifiziert. Die einzelnen Bestandteile werden dabei mit einem „/" getrennt

3.4.8 FunctionUnitType

The Lookup object **FunctionUnitType** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>FunctionUnitType</Name>
<Public>false</Public>
<TableName>FunctionUnitType</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<ImportType>TypeLookUp</ImportType>
```

Note	This Lookup object is used to determine the SCObjectID for a FunctionUnitType. An alternative key does not need to be entered.

3.4.9 FunctionUnitRelationshipType

The Lookup object **FunctionUnitRelationshipType** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>FunctionUnitRelationshipType</Name>
<Public>false</Public>
<TableName>FunctionUnitRelationshipType</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The FunctionUnitRelationshipType can either be resolved via the *SCObjectName* or alternatively via the *FunctionUnitRelationshipType* field.

3.4.10 PrimaryFunctionUnit

The Lookup object **PrimaryFunctionUnit** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>PrimaryFunctionUnit</Name>
<Public>false</Public>
<TableName>FunctionUnit</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,MandatorID,SCObjectID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnitType</DependantObjectName>
        LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The PrimaryFunctionUnit can be identified via the *Identifier* field. Usually, the field is defined as unique.
SCObjectID	The field filters the FunctionUnits based on one type.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
SCObjectID*	**SCObject**	SCObjectName
SCObjectID*	**FunctionUnitType**	FunctionUnitType

* The SCObjectID field can either be resolved via the FunctionUnitType or the SCObjectName.

3.4.11 SecondaryFunctionUnit

The Lookup object **SecondaryFunctionUnit** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SecondaryFunctionUnit</Name>
<Public>false</Public>
<TableName>FunctionUnit</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,MandatorID,SCObjectID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnitType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The SecondaryFunctionUnit can be identified via the *Identifier* field. Usually, the field is defined as unique.
SCObjectID	The field filters the FunctionUnits based on one type.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
SCObjectID*	**SCObject**	SCObjectName
SCObjectID*	**FunctionUnitType**	FunctionUnitType

* The SCObjectID field can either be resolved via the FunctionUnitType or the SCObjectName.

3.4.12 GenericReference

Purpose: A Spider object is resolved via **GenericReference**. To do this, **Objectname** and **ObjectIdentifier** have to be specified. The **Objectname** is the name of the object class, e.g.: Employee, Contract, etc. The **ObjectIdentifier** is the identifier for resolving the object instance and depends on the object class.

The Lookup object **GenericReference** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>GenericReference</Name>
<Public>false</Public>
<TableName>sovObject</TableName>
<AlternativeKeyNames>ObjectName,Identifier,MandatorID</AlternativeKeyNames>
<ImportType>GenericReference</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
ObjectName	A general reference can be resolved using the field *Objectname*.
Identifier	A general reference can be resolved using the field *Identifier*.
MandatorID	The field filters the objects based on one mandator.

3.4.13 LegalEntity

The Lookup object **LegalEntity** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LegalEntity</Name>
<Public>false</Public>
<TableName>Core_LegalEntity_02</TableName>
<PrimaryKeyNames>MandatorID,ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,Path</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
MandatorID	The field filters the objects based on one mandator.
Path	The path can be used only for identification, but not for creating a new legal entity! The path is a combination of superordinated path (ParentID) and the name. Each component is separated by a "/" sign.

3.4.14 Mandator

The Lookup object **Mandator** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Mandator</Name>
<Public>false</Public>
<TableName>Core_Mandator</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The mandator is resolved via the *Name* field.

3.4.15 NextAssetStatus

The Lookup object **NextAssetStatus** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>NextAssetStatus</Name>
<Public>false</Public>
<TableName>AssetStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The NextAssetStatus is resolved via the *Name* field.

3.4.16 SCObject

The Lookup object **SCObject** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SCObject</Name>
<Public>false</Public>
<TableName>SCObject</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The corresponding SCObject can be resolved via the *Name* field.

CHAPTER 4

Spider Licence

In this chapter

4.1 Spider Licence import objects

4.1.1 Article

The object **Article** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Article</Name>
<Public>true</Public>
<TableName>Article</TableName>
<KernelTypeFullName>Spider.Licence.Kernel.Article</KernelTypeFullName>
<KernelTypeAssemblyName>Spider.Licence</KernelTypeAssemblyName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>GUID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>ProductVersion</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
GUID	A unique identifier (Globally Unique Identifier) that identifies exactly this article. If no GUID is specified, it will be generated by the system. Attention: Article numbers (SKU) are not unique! An article import without GUID always leads to the creation of a new article.

LookUp

Feld	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ProductVersionID	**ProductVersion**	ProductVersionName

Example of a Article import file:

```
<ImportData>
<Article>
   <MandatorID>0</MandatorID>
   <GUID>4d7ff34d-337f-4df0-95d0-c4d819a96874</GUID>
   <ProductVersionName>Office 2016 Professional (Device, Win)</ProductVersionName>
   <ArticleNo>123456789</ArticleNo>
   <ArticleText>Sample Article</ArticleText>
   <Quantity>1</Quantity>
   <IsLicence>True</IsLicence>
   <IsMaintenance>True</IsMaintenance>
   <IsSubscription>False</IsSubscription>
   <IsUpdate>True</IsUpdate>
   <DurationType>V</DurationType>
</Article>
</ImportData>
```

4.1.2 Licence

The object **Licence** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Licence</Name>
<Public>true</Public>
<TableName>Licence</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>LicenceType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>LegalEntity</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
```

```
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>LicenceStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ProductVersion</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>UpdateProductVersion</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>DowngradeProductVersion</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>OrganisationalUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>CostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>PurchaseCostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Asset</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ResponsibleEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SPCArticle</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The Licence can be identified via the Identifier field. Usually, the field is defined as unique.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SCObjectID*	**LicenceType**	LicenceType
SCObjectID*	**SCObject**	SCObjectName
LegalEntityID	**LegalEntitys**	LegalEntityPath
LicenceStatusID	**LicenceStatus**	LicenceStatusName
ProductVersionID	**ProductVersion**	ProductVersionName
UpdateProductVersionID	**UpdateProductVersion**	UpdateProductVersionName
DowngradeProduct VersionID	**DowngradeProductVersion**	DowngradeProductVersionName
OrganisationalUnitID	**OrganisationalUnit**	OrganisationalUnitID
CostCentreID	**CostCentre**	CostCentreName CostCentreAccountigArea CostCentreProductArea or CostCentreFullName
PurchaseCostCentreID	**PurchaseCostCentre**	PurchaseCostCentreName PurchaseCostCentreAccountigArea PurchaseCostCentreProductArea or PurchaseCostCentreFullName
AssetID	**Asset**	AssetAssetNo
EmployeeID	**Employee**	EmployeeDomainLogin EmployeeStaffNo EmployeeFullName
ResponsibleEmployeeID	**ResponsibleEmployee**	ResponsibleEmployeeDomainLogin ResponsibleEmployeeStaffNo ResponsibleEmployeeFullName
GUID	**SPCArticle**	SPCArticleItemNumber

* The field SCObjectID can either be resolved via the *LicenceType* or the *SCObjectName*.

Example of a Licence import file:

```
<ImportData>
<!-- This import file requires the Core.LegalEntity import file to be imported first. -->
<Licence>
        <MandatorName>Standard</MandatorName>
        <LicenceType>Product Licence</LicenceType>
        <Identifier>VL_0001</Identifier>
        <LegalEntityPath>Brainwaregroup/Spider LCM GmbH</LegalEntityPath>
        <ProductVersionName>Office 2003 Standard (Device, Win)</ProductVersionName>
        <LicenceStatusName>Active</LicenceStatusName>
        <Quantity>5</Quantity>
        <IsUpdate>False</IsUpdate>
        <Downgradeable>False</Downgradeable>
        <Comment>Testdata</Comment>
</Licence>
</ImportData>
```

4.1.3 LicenceAllocation

The object **LicenceAllocation** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LicenceAllocation</Name>
<Public>true</Public>
<TableName>LicenceAllocation</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>LicenceID,TargetLegalEntityID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Licence</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>TargetLegalEntity</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
LicenceID	This field is used to identify the license which will be used for distribution.
TargetLegalEntityID	This field is used to identify the legal entity which will receive the distributed licenses.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
LicenceID	**Licence**	LicenceIdentifier
TargetLegalEntityID	**TargetLegalEntity**	TargetLegalEntityPath

Example of a LicenceAllocation import file:

```
<ImportData>
<!-- This import file requires the Core.LegalEntity import file to be imported first. -->
<LicenceAllocation>
        <LicenceIdentifier>VL_0001</LicenceIdentifier>
        <TargetLegalEntityPath>Brainwaregroup/Spider LCM
GmbH/Germany</TargetLegalEntityPath>
        <Quantity>2</Quantity>
        <Comment>Testdata</Comment>
</LicenceAllocation>
</ImportData>
```

4.1.4 LicenceKey

The object **LicenceKey** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LicenceKey</Name>
<Public>true</Public>
<TableName>LicenceKey</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>LicenceID,MandatorID,Identifier</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Licence</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Asset</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>FunctionUnit</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
LicenceID	This field is used to identify the license which will be used for distribution.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.
Identifier	The LicenceKey can be identified via the *Identifier* field. Usually, the field is defined as unique.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
LicenceID	**Licence**	LicenceIdentifier
AssetID	**Asset**	AssetAssetNo
FunctionUnitID	**FunctionUnit**	FunctionUnitIdentifier
EmployeeID	**Employee**	EmployeeDomainLogin EmployeeStaffNo EmployeeFullName

Example of a LicenceKey import file:

```
<ImportData>
<LicenceKey>
        <MandatorName>Standard</MandatorName>
        <Identifier>1234-5678-9123-4598-7451</Identifier>
        <LicenceIdentifier>VL_0001</LicenceIdentifier>
        <Active>True</Active>
        <ValidFrom>2014-01-01T12:00:00</ValidFrom>
        <Comment>Testdata</Comment>
</LicenceKey>
</ImportData>
```

4.1.5 Document

Purpose: Import of files as document (attachment) to a Spider object. To do this, the Spider object has to be resolved via *GenericReference*. Depending on the setting, the documents are either loaded into the database or saved into a folder in the file system. The setting can be done in the application configuration.

Section: Spider.Objects.**Kernel.Document**

Key: **UseStorage**

Value: **TRUE / FALSE**

The recommended setting here is the database storage (value = TRUE).

Documents are administered in a document file. This document file contains all documents and is structured in folders.

The object **Document** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Document</Name>
<Public>true</Public>
<TableName>Document</TableName>
<AlternativeKeyNames>Name,SCObjectID,ObjectID,MandatorID</AlternativeKeyNames>
<ImportType>Document</ImportType>
<PreceedingImport>
        <DependantObjectName>GenericReference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Folder</DependantObjectName>
        <LookUp>false</LookUp>
        <Insert>true</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The identifier of the document (Document) can be resolved via the *Name* field.
SCObjectID **ObjectID**	SCObjectID and ObjectID resolve the Spider object, in which the document is to be imported. For resolving a Spider object, a special import object is **available** (see "**GenericReference**" on page 31). SCObjectID is resolved via the object name of the Spider object. ObjectID is resolved in combination with the ObjectIdentifier. You always have to specify a combination.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolve by
GenericReferenceID	**GenericReference**	ObjectName Identifier MandatorName
FolderID	**Folder**	FolderPath

Example of a Document import file:

```
<ImportData>
<!-- The imported document needs to be named Copy-In.doc and placed in the same Folder as this
import file. -->
<Document>
        <MandatorName>Standard</MandatorName>
        <ObjectName>Licence</ObjectName>
        <ObjectIdentifier>VL_0001</ObjectIdentifier>
        <Name>Terms of use</Name>
        <Filename>Copy-In.doc</Filename>
        <FolderPath>eDoc/Terms</FolderPath>
</Document>
</ImportData>
```

Description:

- Documents may be attached to all Spider objects, for which the ObjectProperty "CanHaveDocuments" is set to active.
- Objectname and ObjectIdentifier are used to resolve the Spider object for the document (context)
- The FileName refers to the file, which is to be saved as document in Spider. The file path can be specified in absolute or relative form (to the import folder).
- Document folders (FolderPath) are created.
- If a document is moved to another folder, the old folder will be kept.
- The maximum upload size is determined in the web configuration.
- Version management: A new version will only be created if there is another file or the content of the existing file has been changed. If, for example, only the file name has been changed, no new version will be created. In this case, only the file name will be updated, and no further document will be created.

4.1.6 Product

The object **Product** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Product</Name>
<Public>true</Public>
<TableName>Product</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Manufacturer</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>CommercialEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>TechnicalEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	A Product can be resolved via the *Name* field.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ManufacturerID	**Manufacturer**	ManufacturerCode
CommercialEmployeeID	**CommercialEmployee**	CommercialEmployeeeDomainLogin CommercialEmployeeStaffNo CommercialEmployeeFullName
TechnicalEmployeeID	**TechnicalEmployee**	TechnicalEmployeeDomainLogin TechnicalEmployeeStaffNo TechnicalEmployeeFullName

Example of a Product import file:

```
<ImportData>
<Product>
	<MandatorName>Standard</MandatorName>
	<Name>Spider Core</Name>
	<ManufacturerName>brainwaregroup</ManufacturerName>
	<Comment>Testdata</Comment>
</Product>
<Product>
	<MandatorName>Standard</MandatorName>
	<Name>Spider Contract</Name>
	<ManufacturerName>brainwaregroup</ManufacturerName>
	<Comment>Testdata</Comment>
</Product>
</ImportData>
```

4.1.7 ProductVersion

The object **ProductVersion** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ProductVersion</Name>
<Public>true</Public>
<TableName>ProductVersion</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
	<DependantObjectName>Mandator</DependantObjectName>
	<LookUp>true</LookUp>
	<Insert>false</Insert>
	<Update>false</Update>
</PreceedingImport>
<PreceedingImport>
	<DependantObjectName>ProductVersionType</DependantObjectName>
	<LookUp>true</LookUp>
	<Insert>false</Insert>
	<Update>false</Update>
</PreceedingImport>
<PreceedingImport>
	<DependantObjectName>SCObject</DependantObjectName>
	<LookUp>true</LookUp>
	<Insert>false</Insert>
	<Update>false</Update>
</PreceedingImport>
<PreceedingImport>
	<DependantObjectName>Product</DependantObjectName>
	<LookUp>true</LookUp>
	<Insert>false</Insert>
	<Update>false</Update>
</PreceedingImport>
<PreceedingImport>
	<DependantObjectName>Manufacturer</DependantObjectName>
	<LookUp>true</LookUp>
	<Insert>false</Insert>
	<Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	A ProductVersion can be resolved via the *Name* field.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SCObjectID*	**ProductVersionType**	ProductVersionType
SCObjectID*	**SCObject**	SCObjectName
ProductID	**Product**	ProductName
ManufacturerID	**Manufacturer**	ManufacturerCode

* The field SCObjectID can either be resolved via the *ProductVersionType* field or the *SCObject-Name* field.

Example of a ProductVersion import file:

```
<ImportData>
<ProductVersion>
        <MandatorName>Standard</MandatorName>
        <ProductVersionType>Product Specific based</ProductVersionType>
        <MandatorName>Standard</MandatorName>
        <Name>Spider Contract 6</Name>
        <ProductName>Spider Contract</ProductName>
        <Version>6.1</Version>
        <Comment>Testdata</Comment>
</ProductVersion>
</ImportData>
```

4.1.8 ProductVersionSoftware

The object **ProductVersionSoftware** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ProductVersionSoftware</Name>
<Public>true</Public>
<TableName>ProductVersionSoftware</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>ProductVersionID,SoftwareID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ProductVersion</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Software</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
ProductVersionID **SoftwareID**	These two fields are required for the identification of a ProductVersionSoftware object. For resolving *ProductVersionID* and *SoftwareID*, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ProductVersionID	**ProductVersion**	ProductVersionName
SoftwareID	**Software**	SoftwareName SoftwareVersion or SoftwareInvName

Example of a ProductVersionSoftware import file:

```
<ImportData>
<!-- This import file requires the ProductVersion import file to be imported first. -->
<ProductVersionSoftware>
        <MandatorName>Standard</MandatorName>
        <ProductVersionName>Spider Contract 6</ProductVersionName>
        <SoftwareInvName>SOCONTRACT_61_DE</SoftwareInvName>
        <Comment>Testdata</Comment>
</ProductVersionSoftware>
</ImportData>
```

4.1.9 ProductGrouping

The object **ProductGrouping** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ProductGrouping</Name>
<Public>true</Public>
<TableName>ProductGrouping</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The ProductGrouping can be identified via the Identifier field. Usually, the field is defined as unique.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

Example of a ProductGrouping import file:

```
<ImportData>
<ProductGrouping>
        <MandatorName>Standard</MandatorName>
        <Identifier>Spider Contract</Identifier>
</ProductGrouping>
</ImportData>
```

4.1.10 ProductGroupingProduct

The object **ProductGroupingProduct** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ProductGroupingProduct</Name>
<Public>true</Public>
<TableName>ProductGroupingProduct</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>ProductGroupingID,ProductID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ProductGrouping</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Product</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
ProductGroupingID **ProductID**	These two fields are required for the identification of a Product-GroupingProduct object. For resolving ProductGroupingID and ProductID, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ProductGroupingID	**ProductGrouping**	ProductGroupingIdentifier
ProductID	**Product**	ProductName

Example of a ProductGroupingProduct import file:

```
<ImportData>
<!-- This import file requires the ProductGrouping import file to be imported first. -->
<ProductGroupingProduct>
        <MandatorName>Standard</MandatorName>
        <ProductGroupingIdentifier>Spider Contract</ProductGroupingIdentifier>
        <ProductName>Spider Core</ProductName>
</ProductGroupingProduct>
<ProductGroupingProduct>
        <MandatorName>Standard</MandatorName>
        <ProductGroupingIdentifier>Spider Contract</ProductGroupingIdentifier>
        <ProductName>Spider Contract</ProductName>
</ProductGroupingProduct>
</ImportData>
```

4.1.11 Resubmission

Purpose: Certain Spider objects can send resubmissions. To do this, the Spider object has to be resolved via GenericReference .

The object **Resubmission** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Resubmission</Name>
<Public>true</Public>
<TableName>Resubmission</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<!--    no alternative keys    -->
<AlternativeKeyNames></AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>GenericReference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
<Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SendToEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SendCcEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
SCObjectID ObjectID	**GenericReference**	ObjectName Identifier MandatorName
SendToEmployeeID	**SendToEmployee**	SendToEmployeeStaffNo SendToEmployeeDomainLogin SendToEmployeeFullName
SendCcEmployeeID	**SendCcEmployee**	SendCCEmployeeStaffNo SendCCEmployeeDomainLogin SendCCEmployeeFullName

Example of a Resubmission import file:

```
<ImportData>
<!-- This import file requires the Core.Employee and ProductGrouping import files to be imported
first. -->
<Resubmission>
        <ObjectName>ProductGrouping</ObjectName>
        <ObjectIdentifier>Spider Contract</ObjectIdentifier>
        <Date>2020-01-1T12:00:00</Date>
        <Designation>Test reminder</Designation>
        <Subject>Spider Testmessage</Subject>
        <Body>This is a Testmessage that was importet into your Spider system.</Body>
        <SendToEmployeeStaffNo>E1000</SendToEmployeeStaffNo>
</Resubmission>
</ImportData>
```

4.1.12 Manufacturer

The object **Manufacturer** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Manufacturer</Name>
<Public>true</Public>
<TableName>Manufacturer</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Code,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Code	The manufacturer is resolved via the *Code* field.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

Example of a Manufacturer import file:

```
<ImportData>
<Manufacturer>
        <MandatorName>Standard</MandatorName>
        <Code>BWG</Code>
        <Name>brainwaregroup</Name>
        <Website>http://www.brainwaregroup.com</Website>
</Manufacturer>
</ImportData>
```

4.1.13 Maintenance

The object **Maintenance** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Maintenance</Name>
<Public>true</Public>
<TableName>Maintenance</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>LegalEntity</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>MaintenanceStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SPCArticle</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Manufacturer</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ResponsibleEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>CostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	A Maintenance is identified via the *Identifier* field. Usually, the field is defined as unique.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
LegalEntityID	**LegalEntity**	LegalEntityPath
MaintenanceStatusID	**MaintananceStatus**	MaintananceStatusName
SPCArticleID	**SPCArticle**	SPCArticleItemNumber
ManufacturerID	**Manufacturer**	ManufacturerCode
ResponsibleEmployeeID	**ResponsibleEmployee**	ResponsibleEmployeeDomainLogin ResponsibleEmployeeStaffNo ResponsibleEmployeeFullName
CostCentreID	**CostCentre**	CostCentreName CostCentreAccountigArea CostCentreProductArea or CostCentreFullName

Example of a Maintenance import file:

```
<ImportData>
<Maintenance>
        <MandatorName>Standard</MandatorName>
        <Identifier>Test_MT01</Identifier>
        <MaintenanceStatusName>Active</MaintenanceStatusName>
        <Quantity>10</Quantity>
        <ValidFrom>2014-01-01T00:00:00</ValidFrom>
        <ValidTo>2020-01-01T00:00:00</ValidTo>
        <LicenceIdentifier>VL_0001</LicenceIdentifier>
        <TransactionNumber>TAN-12345</TransactionNumber>
        <SigningDate>2013-12-31T12:00:00</SigningDate>
        <Comment>Testdata</Comment>
</Maintenance>
</ImportData>
```

4.1.14 MaintenanceLicence

The object **MaintenanceLicence** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>MaintenanceLicence</Name>
<Public>true</Public>
<TableName>MaintenanceLicence</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>MaintenanceID,LicenceID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Maintenance</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Licence</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
MaintenanceID	The Maintenance to be used can be identified via this field.
LicenceID	The License, which is to be linked to the Maintenance, can be identified via this field. For resolving the LicenceID, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
MaintenanceID	**Maintenance**	MaintenanceIdentifier
LicenceID	**Licence**	LicenceIdentifier

Example of a MaintenanceLicence import file:

```
<ImportData>
<MaintenanceLicence>
        <MandatorName>Standard</MandatorName>
        <MaintenanceIdentifier>Test_MT01</MaintenanceIdentifier>
        <LicenceIdentifier>VL_0001</LicenceIdentifier>
        <Comment>Testdata</Comment>
</MaintenanceLicence>
</ImportData>
```

4.1.15 MaintenanceProduct

The object **MaintenanceProduct** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>MaintenanceProduct</Name>
<Public>true</Public>
<TableName>MaintenanceProduct</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>MaintenanceID,ProductID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Maintenance</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Product</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
MaintenanceID	The Maintenance to be used can be identified via this field.
ProductID	The Product, which is to be linked to the Maintenance, can be identified via this field. For resolving the ProductID, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
MaintenanceID	**Maintenance**	MaintenanceIdentifier
ProductID	**Product**	ProductName

Example of a MaintenanceProduct import file:

```
<ImportData>
<!-- This import file requires the Maintenance import file to be imported first. -->
<MaintenanceProduct>
        <MandatorName>Standard</MandatorName>
        <MaintenanceIdentifier>Test_MT01</MaintenanceIdentifier>
        <ProductName>Spider Contract</ProductName>
        <Comment>Testdata</Comment>
</MaintenanceProduct>
</ImportData>
```

4.2 Spider Licence system objects

4.2.1 LicenceStatus

The object **LicenceStatus** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LicenceStatus</Name>
<Public>true</Public>
<TableName>LicenceStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	A LicenceStatus can be resolved via the *Name* field.

Example of a LicenceStatus import file:

```
<ImportData>
<LicenceStatus>
        <Name>Pending</Name>
        <Initial>False</Initial>
        <Active>True</Active>
        <Archive>False</Archive>
        <Description>This is an imported Teststatus</Description>
</LicenceStatus>
</ImportData>
```

4.2.2 Currency

The object **Currency** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Currency</Name>
<Public>true</Public>
<TableName>Currency</TableName>
<PrimaryKeyNames>Code</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The currency can be resolved via the *Name* field.

Example of a Currency import file:

```
<ImportData>
<Currency>
        <Code>EUR</Code>
        <Name>EURO</Name>
        <ExchangeRate>0.5</ExchangeRate>
</Currency>
</ImportData>
```

4.2.3 ParaValueDef

The object **ParaValueDef** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ParaValueDef</Name>
<Public>true</Public>
<TableName>ParaValueDef</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Key,Value</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Key	Use these two fields to resolve an entry from an individual value list (ParaValueDef).
Value	

Example of a ParaValueDef import file:

```
<ImportData>
<ParaValueDef>
        <Key>Financing type</Key>
        <Value>Leasing</Value>
        <Text>Leasing</Text>
        <Initial>False</Initial>
        <Active>True</Active>
        <Rank>5</Rank>
</ParaValueDef>
</ImportData>
```

4.3 Spider Licence lookup objects

4.3.1 LicenceType

The Lookup object **LicenceType** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LicenceType</Name>
<Public>false</Public>
<TableName>Licence</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>TypeLookUp</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The LicenceType can either be resolved via the *SCObjectName* field or alternatively via the *LicenceType* field.

4.3.2 CostCentre

The Lookup object **CostCentre** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>CostCentre</Name>
<Public>false</Public>
<TableName>Core_CostCentre</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,ProductArea,AccountingArea,FullName,MandatorID</AlternativeKeyN
ames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name **ProductArea** **AccountingArea**	In principle, these three fields are required for the identification of a CostCentre object. If the fields *ProductArea* and *AccountingArea* are not used in Spider, it is sufficient to identify the CostCentre object via *Name*.
FullName	This field is a combination of the following fields: *<Name> <ProductArea> <AccountingArea>*
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.3 PurchaseCostCentre

The Lookup object **PurchaseCostCentre** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>PurchaseCostCentre</Name>
<Public>false</Public>
<TableName>Core_CostCentre</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,ProductArea,AccountingArea,FullName,MandatorID</AlternativeKeyN
ames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name **ProductArea** **AccountingArea**	In principle, these three fields are required for the identification of a CostCentre object. If the fields *ProductArea and AccountingArea* are not used in Spider, it is sufficient to identify the CostCentre object via *Name.*
FullName	This field is a combination of the following fields: *<Name> <ProductArea> <AccountingArea>*
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.4 Employee

The Lookup object **Employee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Employee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.5 SendToEmployee

The Lookup object **SendToEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SendToEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.6 SendCcEmployee

The Lookup object **SendCcEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SendCcEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.7 ResponsibleEmployee

The Lookup object **ResponsibleEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ResponsibleEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The ResponsibleEmployee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The ResponsibleEmployee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.8 ProductVersionType

The Lookup object **ProductVersionType** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ProductVersionType</Name>
<Public>false</Public>
<TableName>ProductVersion</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>TypeLookUp</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The type of a license metric (ProductVersionType) can either be resolved via the *SCObjectName* field or alternatively via the *ProductVersionType* field.

4.3.9 DowngradeProductVersion

The Lookup object **DowngradeProductVersion** is used to resolve the DowngradeProductVersionID at the license.

Excerpt from the ImportConfiguration

```
<Name>DowngradeProductVersion</Name>
<Public>false</Public>
<TableName>ProductVersion</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The product version, which is used as a basis for a downgrade, can be resolved via the *Name* field.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.10 UpdateProductVersion

The Lookup object **UpdateProductVersion** is used to resolve the UpdateProductVersionID at the license.

Excerpt from the ImportConfiguration

```
<Name>UpdateProductVersion</Name>
<Public>false</Public>
<TableName>ProductVersion</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The product version, which is the basis for an update, can be resolved via the *Name* field.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.11 Folder

Zweck: Zugriff auf Verzeichnisse innerhalb einer Dokumentenakte eines Spider Objektes. Folder werden beim Importobjekt *Document* benötigt.

Das Lookup Objekt **Folder** wird nach folgenden Regeln aufgelöst:

Excerpt from the ImportConfiguration

```
Name>Folder</Name>
<Public>false</Public>
<TableName>Folder</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Path</AlternativeKeyNames>
<ImportType>Folder</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Beschreibung
Path	Ein Import wird über den Pfad identifiziert. Die einzelnen Bestandteile werden dabei mit einem „/" getrennt

4.3.12 Asset

The Lookup object **Asset** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Asset</Name>
<Public>false</Public>
<TableName>Asset_Asset</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>AssetNo,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
AssetNo	An asset is resolved via the *AssetNo* field.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.13 Software

The Lookup object **Software** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Software</Name>
<Public>false</Public>
<TableName>Asset_Software</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,Version,InvName,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The Software object can be resolved via the *Name* field.
Version	The field filters the software based on a version.
InvName	The field filters the objects based on one inventory name (InvName). This value is unique.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.14 FunctionUnit

The Lookup object **FunctionUnit** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>FunctionUnit</Name>
<Public>false</Public>
<TableName>Asset_FunctionUnit</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The FunctionUnit object can be identified via the *Identifier* field.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.15 OrganisationalUnit

The Lookup object **OrganisationalUnit** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>OrganisationalUnit</Name>
<Public>false</Public>
<TableName>Asset_OrganisationalUnit</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.16 MaintenanceStatus

The Lookup object **MaintenanceStatus** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>MaintenanceStatus</Name>
<Public>false</Public>
<TableName>MaintenanceStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The MaintenanceStatus can be resolved via the *Name* field.

4.3.17 GenericReference

Purpose: A Spider object is resolved via **GenericReference**. To do this, **Objectname** and **ObjectIdentifier** have to be specified. The **Objectname** is the name of the object class, e.g.: Employee, Contract, etc. The **ObjectIdentifier** is the identifier for resolving the object instance and depends on the object class.

The Lookup object **GenericReference** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>GenericReference</Name>
<Public>false</Public>
<TableName>sovObject</TableName>
<AlternativeKeyNames>ObjectName,Identifier,MandatorID</AlternativeKeyNames>
<ImportType>GenericReference</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
ObjectName	A general reference can be resolved using the field *ObjectName*.
Identifier	A general reference can be resolved using the field *Identifier*.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.18 LegalEntity

The Lookup object **LegalEntity** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LegalEntity</Name>
<Public>false</Public>
<TableName>Core_LegalEntity_02</TableName>
<PrimaryKeyNames>MandatorID,ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,Path</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
MandatorID	The field filters the objects based on one mandator.
Path	The path can be used only for identification, but not for creating a new legal entity! The path is a combination of superordinated path (ParentID) and the name. Each component is separated by a "/" sign.

4.3.19 TargetLegalEntity

The Lookup object **TargetLegalEntity** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>TargetLegalEntity</Name>
<Public>false</Public>
<TableName>Core_LegalEntity_02</TableName>

<PrimaryKeyNames>MandatorID,ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,Path</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Path	The TargetLegalEntity object can be identified via the path. The path is a combination of superordinated path (ParentID) and the name. Each component is separated by a "/" sign.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

4.3.20 Mandator

The Lookup object **Mandator** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Mandator</Name>
<Public>false</Public>
<TableName>Core_Mandator</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The mandator is resolved via the *Name* field.

4.3.21 SCObject

The Lookup object **SCObject** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SCObject</Name>
<Public>false</Public>
<TableName>SCObject</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The corresponding SCObject can be resolved via the *Name* field.

4.3.22 SPCArticle

The Lookup object SPCArticle is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SPCArticle</Name>
<Public>false</Public>
<TableName>spcArticle</TableName>
<PrimaryKeyNames>GUID</PrimaryKeyNames>
<AlternativeKeyNames>ItemNumber</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
ItemNumber	The SPCArticle can be identified via the *ItemNumber (SKU)*.

CHAPTER 5

Spider Contract

In this chapter

5.1 Spider Contract import objects

5.1.1 Contractor

The object **Contractor** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Contractor</Name>
<Public>true</Public>
<TableName>Contractor</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,Name,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ContractorStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ContractorCategory</DependantObjectName>
        <LookUp>false</LookUp>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The Contractor can be identified via the *Identifier* field. Usually, the field is defined as unique.
Name	Alternatively, the Contractor can be resolved via the *Name* field.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ContractorStatusID	**ContractorStatus**	ContractorStatusName
ContractorCategoryID	**ContractorCategory**	ContractorCategoryName

Example of a Contractor import file:

```
<ImportData>
<Contractor>
        <MandatorName>Standard</MandatorName>
        <Name>Spider LCM GmbH</Name>
        <Identifier>SpiderLCM</Identifier>
        <ContractorCategoryName>Supplier</ContractorCategoryName>
        <ContractorStatusName>Active</ContractorStatusName>
        <Code>EP</Code>
        <Comment>Testdata</Comment>
</Contractor>
<Contractor>
        <MandatorName>Standard</MandatorName>
        <Name>Brainware Holding AG</Name>
        <Identifier>BWG</Identifier>
        <ContractorCategoryName>Manufacturer</ContractorCategoryName>
        <ContractorStatusName>Active</ContractorStatusName>
        <Code>BW</Code>
        <Comment>Testdata</Comment>
</Contractor>
</ImportData>
```

5.1.2 Address

The object **Address** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Address</Name>
<Public>true</Public>
<TableName>Address</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,ContractorID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>

        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contractor</DependantObjectName>

        <LookUp>true</LookUp>

        <Insert>false</Insert>

        <Update>false</Update>

</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier **ContractorID**	These fields are required for the identification of an Address object. For resolving the ContractorID, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ContractorID	**Contractor**	ContractorIdentifier or ContractorName

Note: The configuration key **Spider.Contract.Kernel.Address.Enabled** can be used to determine whether the address data are kept up-to-date via the Address object. If the value is set to FALSE, an import of the address data at the Address object is ignored. If the value is been set to TRUE, then the address data is stored in the address table. A Contract can have more than one address.

Example of an Address import file:

```
<ImportData>
<Address>
        <MandatorName>Standard</MandatorName>
        <ContractorIdentifier>SpiderLCM</ContractorIdentifier>
        <Identifier>SPLCM_Hamburg</Identifier>
        <Name>Spider LCM GmbH (Hamburg)</Name>
        <Street>Paul-Dessau-Strasse 8</Street>
        <PostCode>22761</PostCode>
        <City>Hamburg</City>
        <District>Hamburg</District>
        <Country>Germany</Country>
        <Phone>+49 40 788 999-0</Phone>
        <Fax>+49 40 788 999-9</Fax>
        <Category>Supplier</Category>
</Address>
</ImportData>
```

5.1.3 ContactPerson

The object **ContactPerson** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContactPerson</Name>
<Public>true</Public>
<TableName>ContactPerson</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>LastName,FirstName,ContractorID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contractor</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
LastName* **FirstName*** **ContractorID**	In principle, these three fields are required in combination for the identification of a ContactPerson object. For resolving the *ContractorID*, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

* If a contractor has more than one contacts with identical first and last name, the import has to be carried out via the primary key. A resolution via the alternative key is no longer possible.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ContractorID	**Contractor**	ContractorIdentifier ContractorName

Example of a ContactPerson import file:

```
<ImportData>
<ContactPerson>
        <MandatorName>Standard</MandatorName>
        <ContractorIdentifier>SpiderLCM</ContractorIdentifier>
        <LastName>Smith</LastName>
        <FirstName>John</FirstName>
        <Title>Dr</Title>
        <Phone>+49 40 788 999-0</Phone>
        <Fax>+49 40 788 999-9</Fax>
</ContactPerson>
</ImportData>
```

5.1.4 Contract

The object **Contract** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Contract</Name>
<Public>true</Public>
<TableName>Contract</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,SCObjectID,MandatorID</AlternativeKeyNames>

<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ContractType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>LegalEntity</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ContractStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contractor</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ContactPerson</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>CostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>PaymentMode</DependantObjectName>
        <LookUp>false</LookUp>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ThemesGroup</DependantObjectName>
        <LookUp>false</LookUp>
</PreceedingImport>
```

```
<PreceedingImport>
        <DependantObjectName>RequesterEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ResponsibleEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>PurchaserEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Reference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Parent</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<!--    <SucceedingImport>
        <DependantObjectName>Note</DependantObjectName>
        <Insert>true</Insert>
        <Update>false</Update>
        </SucceedingImport> -->
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	If "unique" is defined for the Identifier field, this is sufficient for the identification of a contract. Otherwise, further alternative keys must be specified.
SCObjectID	The field filters the contracts based on one type.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SCObjectID*	**ContractType**	ContractType
SCObjectID*	**SCObject**	SCObjectName
LegalEntityID	**LegalEntity**	LegalEntityPath
ContractStatusID	**ContractStatus**	ContractStatusName
ContractorID**	**Contractor**	ContractorName ContractorIdentifier
ContactPersonID	**ContactPerson**	ContactPersonName
CostCentreID	**CostCentre**	CostCentreName CostCentreAccountigArea CostCentreProductArea or CostCentreFullName
PaymentModeID	**PaymentMode**	PaymentModeName
ThemesGroupID	**ThemesGroup**	ThemesGroupName
RequesterEmployeeID	**RequesterEmployee**	RequesterEmployeeDomainLogin RequesterEmployeeStaffNo RequesterEmployeeFullName
ResponsibleEmployeeID	**ResponsibleEmployee**	ResponsibleEmployeeDomainLogin ResponsibleEmployeeStaffNo ResponsibleEmployeeFullName
PurchaseEmployeeID	**PurchaseEmployee**	PurchaseEmployeeDomainLogin PurchaseEmployeeStaffNo PurchaseEmployeeFullName
ReferenceID	**Reference**	ReferenceIdentifier ReferenceSCObjectName ReferenceMandatorName
ParentID	**Parent**	ParentIdentifier ParentSCObjectName ParentMandatorName

* The field SCObjectID can either be resolved via the ContractType or the SCObjectName.

** ContractorID for the automatic establishment of the ContractorAssignment.

Example of a Contract import file:

```
<ImportData>
<Contract>
        <MandatorName>Standard</MandatorName>
        <Identifier>C001_0001</Identifier>
        <ContractType>Service Contract</ContractType>
        <ContractStatusName>Active</ContractStatusName>
        <ExternalContractNumber>EX_C00001</ExternalContractNumber>
        <StartDate>2014-01-01T12:00:00</StartDate>
        <CurrencyCode>EUR</CurrencyCode>
        <Duration>60</Duration>
        <Comment>Testdata</Comment>
</Contract>
</ImportData>
```

5.1.5 ContractorAssignment

The object **ContractorAssignment** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContractorAssignment</Name>
<Public>true</Public>
<TableName>ContractorAssignment</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,ContractID,ContractorID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contract</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contractor</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The ContractorAssignment can be resolved via the *Name* field. Since it is usually not unique, the resolution must be carried out via other alternative keys.
ContractID **ContractorID**	Alternatively, the identification of the ContractorAssignment can be carried out via these two fields. For resolving *ContractID* and *ContractorID*, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ContractID	**Contract**	ContractIdentifier ContractSCObjectName
ContractorID	**Contractor**	ContractorIdentifier ContractorName

Example of a ContractorAssignment import file:

```
<ImportData>
<ContractorAssignment>
        <MandatorName>Standard</MandatorName>
        <ContractIdentifier>C001_0001</ContractIdentifier>
        <ContractorIdentifier>SpiderLCM</ContractorIdentifier>
        <Name>Agreement_01</Name>
        <AddressIdentifier>SPLCM_Hamburg</AddressIdentifier>
        <Comment>Testdata</Comment>
</ContractorAssignment>
</ImportData>
```

5.1.6 ContractPayment

The object **ContractPayment** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContractPayment</Name>
<Public>true</Public>
<TableName>ContractPayment</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,ContractID,Purpose</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contract</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
ContractID	The field filters the payment plans (ContractPayment) based on one contract object.
Purpose	The purpose of a contract payment plan can be resolved via the *Purpose* field.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ContractID	**Contract**	ContractIdentifier ContractSCObjectName

Example of a ContractPayment import file:

```
<ImportData>
<ContractPayment>
        <MandatorName>Standard</MandatorName>
        <ContractIdentifier>C001_0001</ContractIdentifier>
        <Purpose>Agreement</Purpose>
        <PaymentDate>2014-01-01T12:00:00</PaymentDate>
        <Amount>9889210</Amount>
        <AmountEUR>9889210</AmountEUR>
        <CurrencyCode>EUR</CurrencyCode>
        <Comment>Testdata</Comment>
</ContractPayment>
</ImportData>
```

5.1.7 Attachment

The object **Attachment** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Attachment</Name>
<Public>true</Public>
<TableName>Attachment</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,ContractID,SCObjectID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AttachmentType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AttachmentStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contract</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>RequesterEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ResponsibleEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>CostCentre</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>PaymentMode</DependantObjectName>
        <LookUp>false</LookUp>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	If "unique" is defined for the *Identifier* field, this is sufficient for the identification of an attachment. Otherwise, further alternative keys must be specified.
ContractID	The field filters the attachments based on one contract object.
SCObjectID	The field filters the attachments based on one type.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SCObjectID*	**AttachmentType**	AttachmentType
SCObjectID*	**SCObject**	SCObjectName
AttachmentStatusID	**AttachmentStatus**	AttachmentStatusName
ContractID	**Contract**	ContractIdentifier ContractSCObjectName
RequesterEmployeeID	**RequesterEmployee**	RequesterEmployeeDomainLogin RequesterEmployeeStaffNo RequesterEmployeeFullName
ResponsibleEmployeeID	**ResponsibleEmployee**	ResponsibleEmployeeDomainLogin ResponsibleEmployeeStaffNo ResponsibleEmployeeFullName
CostCentreID	**CostCentre**	CostCentreName CostCentreAccountigArea CostCentreProductArea or CostCentreFullName
PaymentModeID	**PaymentMode**	PaymentModeName

* The field SCObjectID can either be resolved via the *AttachmentType* or the *SCObjectName*.

Example of an Attachment import file:

```
<ImportData>
<Attachment>
        <MandatorName>Standard</MandatorName>
        <AttachmentType>Subject supplement agreement</AttachmentType>
        <AttachmentStatusName>Active</AttachmentStatusName>
        <Identifier>Att_0001</Identifier>
        <ContractIdentifier>C001_0001</ContractIdentifier>
        <Purpose>Agreement</Purpose>
        <PaymentDate>2014-01-01T12:00:00</PaymentDate>
        <Amount>9889210</Amount>
        <AmountEUR>9889210</AmountEUR>
        <CurrencyCode>EUR</CurrencyCode>
        <Comment>Testdata</Comment>
</Attachment>
</ImportData>
```

5.1.8 LegalSuccession

Purpose: Legal succession of contractors.

The object **LegalSuccession** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LegalSuccession</Name>
<Public>true</Public>
<TableName>LegalSuccession</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,AssignorID,AssigneeID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Assignor</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Assignee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	If "unique" is defined for the *Identifier* field, this is sufficient for the identification of a LegalSuccession. Otherwise, further alternative keys must be specified.
AssignorID	For resolving the AssignorID, the alternative keys of the contractor object are available.
AssigneeID	For resolving the AssigneeID, the alternative keys of the contractor object are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
AssignorID	**Assignor**	AssignorIdentifier
AssignneeID	**Assignee**	AssigneeIdentifier

Example of a LegalSuccession import file:

```
<ImportData>
<LegalSuccession>
        <MandatorName>Standard</MandatorName>
        <AssignorIdentifier>SpiderLCM</AssignorIdentifier>
        <AssigneeIdentifier>BWG</AssigneeIdentifier>
        <Identifier>SPBWmerge</Identifier>
        <Comment>Testdata</Comment>
</LegalSuccession>
</ImportData>
```

5.1.9 Document

Purpose: Import of files as document (attachment) to a Spider object. To do this, the Spider object has to be resolved via *GenericReference*. Depending on the setting, the documents are either loaded into the database or saved into a folder in the file system. The setting can be done in the application configuration.

Section: Spider.Objects.**Kernel.Document**

Key: **UseStorage**

Value: **TRUE / FALSE**

The recommended setting here is the database storage (value = TRUE).

Documents are administered in a document file. This document file contains all documents and is structured in folders.

The object **Document** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Document</Name>
<Public>true</Public>
<TableName>Document</TableName>
<AlternativeKeyNames>Name,SCObjectID,ObjectID,MandatorID</AlternativeKeyNames>
<ImportType>Document</ImportType>
<PreceedingImport>
        <DependantObjectName>GenericReference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Folder</DependantObjectName>
        <LookUp>false</LookUp>
        <Insert>true</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The identifier of the document (Document) can be resolved via the *Name* field.
SCObjectID **ObjectID**	SCObjectID and ObjectID resolve the Spider object, in which the document is to be imported. For resolving a Spider object, a special import object is **available** (see "**GenericReference**" on page 31). SCObjectID is resolved via the object name of the Spider object. ObjectID is resolved in combination with the ObjectIdentifier. You always have to specify a combination.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolve by
GenericReferenceID	**GenericReference**	ObjectName Identifier MandatorName
FolderID	**Folder**	FolderPath

Example of a Document import file:

```
<ImportData>
<!-- The imported document needs to be named Copy-In.doc and placed in the same Folder as this
import file. -->
<Document>
        <MandatorName>Standard</MandatorName>
        <ObjectName>Contract</ObjectName>
        <ObjectIdentifier>C001_0001</ObjectIdentifier>
        <Name>Office</Name>
        <Filename>Copy-In.doc</Filename>
        <FolderPath>eDoc/Copies</FolderPath>
</Document>
</ImportData>
```

Description:

- Documents may be attached to all Spider objects, for which the ObjectProperty "CanHaveDocuments" is set to active.
- Objectname and ObjectIdentifier are used to resolve the Spider object for the document (context)
- The FileName refers to the file, which is to be saved as document in Spider. The file path can be specified in absolute or relative form (to the import folder).
- Document folders (FolderPath) are created.
- If a document is moved to another folder, the old folder will be kept.
- The maximum upload size is determined in the web configuration.
- Version management: A new version will only be created if there is another file or the content of the existing file has been changed. If, for example, only the file name has been changed, no new version will be created. In this case, only the file name will be updated, and no further document will be created.

5.1.10 Invoice

The object **Invoice** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Invoice</Name>
<Public>true</Public>
<TableName>Invoice</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>InvoiceNumber,ContractID,AttachmentID,MandatorID</AlternativeKeyName
s>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contract</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Attachment</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
InvoiceNumber	If "unique" is defined for the *Identifier* field, this is sufficient for the identification of an invoice. Otherwise, further alternative keys must be specified.
ContractID	The field filters the invoices based one contract object.
AttachmentID	The field filters the objects based on one attachment.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	Mandator	MandatorName
ContractID*	Contract	ContractIdentifier ContractSCObjectName
AttachmentID*	Attachment	AttachmentIdentifier AttachmentContractIdentifier AttachmentSCObjectName

* You can resolver either the *ContractID* or the *AttachmentID*.

Example of an Invoice import file:

```
<ImportData>
<Invoice>
        <MandatorName>Standard</MandatorName>
        <ContractIdentifier>C001_0001</ContractIdentifier>
        <InvoiceNumber>InvoiceNumber_001</InvoiceNumber>
        <ERPReferenceNumber>InvoiceERPNumber_001</ERPReferenceNumber>
        <CurrencyCode>EUR</CurrencyCode>
        <Date>2014-01-01T12:00:00+01:00</Date>
        <Comment>Testdata</Comment>
</Invoice>
</ImportData>
```

5.1.11 ReferenceObject

The object **ReferenceObject** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ReferenceObject</Name>
<Public>true</Public>
<TableName>ReferenceObject</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,Category,SCObjectID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ReferenceObjectType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	If "unique" is defined for the *Identifier* field, this is sufficient for the identification of a ReferenceObject. Otherwise, further alternative keys must be specified.
Category	The field filters the objects based on one category.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

Example of a ReferenceObject import file:

```
<ImportData>
<ReferenceObject>
        <MandatorName>Standard</MandatorName>
        <Identifier>REF_001</Identifier>
        <ReferenceObjectType>Reference</ReferenceObjectType>
        <Category>General</Category>
        <Comment>Testdata</Comment>
</ReferenceObject>
</ImportData>
```

5.1.12 ContractReferenceObject

The object **ContractReferenceObject** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContractReferenceObject</Name>
<Public>true</Public>
<TableName>ContractReferenceObject</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>ContractID,ReferenceObjectID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contract</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ReferenceObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
ContractID **ReferenceObjectID**	Both fields are required for the identification of a ContractReferenceObject object. For resolving the *ContractID* and the *ReferenceObjectID*, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ContractID	**Contract**	ContractIdentifier ContractSCObjectName
ReferenceObjectID	**ReferenceObject**	ReferenceObjectIdentifier ReferenceObjectCategory

Example of a ContractReferenceObject import file:

```
<ImportData>
<ContractReferenceObject>
        <MandatorName>Standard</MandatorName>
        <ReferenceObjectIdentifier>REF_001</ReferenceObjectIdentifier>
        <ContractIdentifier>C001_0001</ContractIdentifier>
        <Comment>Testdata</Comment>
</ContractReferenceObject>
</ImportData>
```

5.1.13 AuthorisationGroup

The object **AuthorisationGroup** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AuthorisationGroup</Name>
<Public>true</Public>
<TableName>AuthorisationGroup</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The AuthorisationGroup can be resolved via the *Name* field.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

Example of an AuthorisationGroup import file:

```
<ImportData>
<AuthorisationGroup>
        <MandatorName>Standard</MandatorName>
        <Name>AG_Main</Name>
        <Description>This is the main authorisation group.</Description>
        <Comment>Testdata</Comment>
</AuthorisationGroup>
</ImportData>
```

5.1.14 AuthorisationGroupEmployee

Th object **AuthorisationGroupEmployee** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AuthorisationGroupEmployee</Name>
<Public>true</Public>
<TableName>AuthorisationGroupEmployee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>AuthorisationGroupID,EmployeeID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AuthorisationGroup</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Employee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
AuthorisationGroupID **EmployeeID**	Both fields are required for the identification of an AuthorisationGroupEmployee object. For resolving the EmployeeID, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
AuthorisationGroupID	**AuthorisationGroup**	AuthorisationGroupName
EmployeeID	**Employee**	EmployeeDomainLogin EmployeeStaffNo EmployeeFullName

Example of an AuthorisationGroupEmployee import file:

```
<ImportData>
<!-- This import file requires the Core.Employee import file to be imported first. -->
<AuthorisationGroupEmployee>
        <MandatorName>Standard</MandatorName>
        <AuthorisationGroupName>AG_Main</AuthorisationGroupName>
        <EmployeeDomainLogin>JohSm</EmployeeDomainLogin>
        <Description>Imported Testdata</Description>
        <Comment>Testdata</Comment>
</AuthorisationGroupEmployee>
</ImportData>
```

5.1.15 ContractAuthorisationGroup

The object **ContractAuthorisationGroup** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContractAuthorisationGroup</Name>
<Public>true</Public>
<TableName>ContractAuthorisationGroup</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>ContractID,AuthorisationGroupID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contract</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AuthorisationGroup</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
ContractID **AuthorisationGroupID**	Both fields are required for the identification of a ContractAuthorisationGroup object. For resolving the *ContractID* and *AuthorisationGroupID*, the corresponding alternative keys are available.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ContractID	**Contract**	ContractIdentifier ContractSCObjectName
AuthorisationGroupID	**AuthorisationGroup**	AuthorisationGroupName

Example of a ContractAuthorisationGroup import file:

```
<ImportData>
<ContractAuthorisationGroup>
        <MandatorName>Standard</MandatorName>
        <ContractIdentifier>C001_0001</ContractIdentifier>
        <AuthorisationGroupName>AG_Main</AuthorisationGroupName>
        <Description>Imported Testdata</Description>
        <Comment>Testdata</Comment>
</ContractAuthorisationGroup>
</ImportData>
```

5.1.16 Note

The object **Note** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Note</Name>
<Public>true</Public>
<TableName>Note</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>SCObjectID,ObjectID,Text</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>GenericReference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Path	An import is identified via the path. Each component is separated by a "/" sign

Field	DependantObjectName	AK / Resolution by
GenericReferenceID	**GenericReference**	ObjectName Identifier MandatorName

Example of a Note import file:

```
<ImportData>
<Note>
        <ObjectName>Contract</ObjectName>
        <ObjectIdentifier>C001_0001</ObjectIdentifier>
        <Text>Note - always comes in handy.</Text>
</Note>
</ImportData>
```

5.1.17 Resubmission

Purpose: Certain Spider objects can send resubmissions. To do this, the Spider object has to be resolved via GenericReference .

The object **Resubmission** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Resubmission</Name>
<Public>true</Public>
<TableName>Resubmission</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<!--    no alternative keys    -->
<AlternativeKeyNames></AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>GenericReference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
<Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SendToEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SendCcEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
SCObjectID ObjectID	**GenericReference**	ObjectName Identifier MandatorName
SendToEmployeeID	**SendToEmployee**	SendToEmployeeStaffNo SendToEmployeeDomainLogin SendToEmployeeFullName
SendCcEmployeeID	**SendCcEmployee**	SendCCEmployeeStaffNo SendCCEmployeeDomainLogin SendCCEmployeeFullName

Example of a Resubmission import file:

```
<ImportData>
<!-- This import file requires the Core.Employee import file to be imported first. -->
<Resubmission>
        <ObjectName>Contract</ObjectName>
        <ObjectIdentifier>C001_0001</ObjectIdentifier>
        <Date>2020-01-1T12:00:00</Date>
        <Designation>Test Designation 1</Designation>
        <Subject>Spider Testmessage</Subject>
        <Body>This is a Testmessage that was importet into your Spider system.</Body>
        <SendToEmployeeStaffNo>E1000</SendToEmployeeStaffNo>
</Resubmission>
</ImportData>
```

5.1.18 Task

The object **Task** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Task</Name>
<Public>true</Public>
<TableName>Task</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Subject,OwnerEmployeeID,SCObjectID,ObjectID,MandatorID</AlternativeK
eyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>GenericReference</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>TaskStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>InitiatorEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>OwnerEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Subject	The name of the task can be resolved via the *Subject* field.
OwnerEmployeeID	The owner of the resubmission can be resolved via the alternative keys of the *Employee* object.
SCObjectID	The SCObjectID can be used to resolve the object name, which the task is referring to (usually *Contract* or *Attachment*).
ObjectID	The identification of the object referenced in the task is carried out via the *ObjectID* field.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
SCObjectID ObjectID	**GenericReference**	ObjectName Identifier MandatorName
TaskStatusID	**TaskStatus**	TaskStatusName
InitiatorEmployeeID	**InitiatorEmployee**	InitiatorrEmployeeStaffNo InitiatorEmployeeDomainLogin InitiatorEmployeeFullName
OwnerEmployeeID	**OwnerEmployee**	OwnerEmployeeStaffNo OwnerEmployeeDomainLogin OwnerEmployeeFullName

Example of a Task import file:

```
<ImportData>
<!-- This import file requires the Core.Employee import file to be imported first. -->
< Task>
        <MandatorName>Standard</MandatorName>
        <Subject>important to do</Subject>
        <Description>Delete Imported demo files from your system</Description>
        <InitiatorEmployeeDomainLogin>JohSm</InitiatorEmployeeDomainLogin>
        <OwnerEmployeeDomainLogin>JohSm</OwnerEmployeeDomainLogin>
        <StartDate>2014-01-01T12:00:00+01:00</StartDate>
        <DueDate>2016-01-01T12:00:00+01:00</DueDate>
        <Priority>2</Priority>
        <TaskStatusName>TaskStatus_New</TaskStatusName>
        <Completion>10</Completion>
        <SCObjectName>Contract</SCObjectName>
        <ObjectIdentifier>C001_0001</ObjectIdentifier>
</Task>
</ImportData>
```

5.1.19 Rating

The object **Rating** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Rating</Name>
<Public>true</Public>
<TableName>Rating</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,ContractID,AttachmentID,ContractorID,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contract</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Attachment</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>Contractor</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	If "unique" is defined for the *Identifier* field, this is sufficient for the identification of a rating. Otherwise, further alternative keys must be specified.
AttachmentID	For resolving the rating, one of the three fields is required. The AttachmentID field links the rating to an attachment object.
ContractID	The ContractID field links the rating to a contract object.
ContractorID	The ContractorID field links the rating to a contractor object.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ContractID*	**Contract**	ContractIdentifier ContractSCObjectName
AttachmentID*	**Attachment**	AttachmentIdentifier AttachmentContractIdentifier AttachmentSCObjectName
ContractorID*	**Contractor**	ContractorIdentifier ContractorName

* You can either resolve the *ContractID*, *AttachmentID* or *ContractorID* field.

Example of a Rating import file:

```
<ImportData>
<Rating>
        <MandatorName>Standard</MandatorName>
        <ContractIdentifier>C001_0001</ContractIdentifier>
        <Identifier>Ratin_001</Identifier>
        <Comment>Testdata</Comment>
</Rating>
</ImportData>
```

5.2 Spider Contract system objects

5.2.1 AttachmentStatus

The object **AttachmentStatus** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AttachmentStatus</Name>
<Public>true</Public>
<TableName>AttachmentStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The AttachmentStatus can be resolved via the *Name* field.

Example of an AttachmentStatus import file:

```
<ImportData>
<AttachmentStatus>
        <Name>AttachementStatus_InProgress</Name>
        <Initial>False</Initial>
        <Active>True</Active>
        <Archive>False</Archive>
        <Description>Testdata</Description>
</AttachmentStatus>
<AttachmentStatus>
        <Name>AttachementStatus_Final</Name>
        <Initial>False</Initial>
        <Active>True</Active>
        <Archive>False</Archive>
        <Description>Testdata</Description>
</AttachmentStatus>
</ImportData>
```

5.2.2 AttachmentStatusSequence

The object **AttachmentStatusSequence** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AttachmentStatusSequence</Name>
<Public>true</Public>
<TableName>AttachmentStatusSequence</TableName>
<PrimaryKeyNames>AttachmentStatusID,NextAttachmentStatusID,SCObjectID</PrimaryKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>AttachmentType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>AttachmentStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>NextAttachmentStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
SCObjectID*	**AttachmentType**	AttachmentType
SCObjectID*	**SCObject**	SCObjectName
AttachmentStatusID	**AttachmentStatus**	AttachmentStatusName
NextAttachmentStatusID	**NextAttachmentStatus**	NextAttachmentStatusName

* The field SCObjectID can either be resolved via the *AttachmentType* or the *SCObjectName*.

Example of a ContractorCategory import file:

```
<ImportData>
<AttachmentStatusSequence>
        <AttachmentStatusName>AttachementStatus_InProgress</AttachmentStatusName>
        <NextAttachmentStatusName>AttachementStatus_Final</NextAttachmentStatusName>
        <SCObjectName>Subject Service</SCObjectName>
        <Rank>0</Rank>
</AttachmentStatusSequence>
</ImportData>
```

5.2.3 ContractorCategory

The object **ContractorCategory** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContractorCategory</Name>
<Public>true</Public>
<TableName>ContractorCategory</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The ContractorCategory can be resolved via the *Name* field.

Example of a ContractorCategory import file:

```
<ImportData>
<ContractorCategory>
        <Name>Distributor</Name>
        <Description>Our main distributor</Description>
</ContractorCategory>
</ImportData>
```

5.2.4 ContractorStatus

The object **ContractorStatus** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContractorStatus</Name>
<Public>true</Public>
<TableName>ContractorStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The ContractorStatus can be resolved via the *Name* field.

Example of a ContractorStatus import file:

```
<ImportData>
<ContractorStatus>
        <Name>planned</Name>
        <Initial>False</Initial>
        <Active>True</Active>
        <Archive>False</Archive>
        <Description>Testdata</Description>
</ContractorStatus>
</ImportData>
```

5.2.5 ContractStatus

The object **ContractStatus** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContractStatus</Name>
<Public>true</Public>
<TableName>ContractStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The ContractStatus can be resolved via the *Name* field.

Example of a ContractStatus import file:

```
<ImportData>
<ContractStatus>
        <Name>planned</Name>
        <Initial>False</Initial>
        <Active>True</Active>
        <Archive>False</Archive>
        <Description>Testdata</Description>
</ContractStatus>
</ImportData>
```

5.2.6 ContractStatusSequence

The object **ContractStatusSequence** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContractStatusSequence</Name>
<Public>true</Public>
<TableName>ContractStatusSequence</TableName>
<PrimaryKeyNames>ContractStatusID,NextContractStatusID,SCObjectID</PrimaryKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>ContractType</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>SCObject</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>ContractStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
<PreceedingImport>
        <DependantObjectName>NextContractStatus</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Field	DependantObjectName	AK / Resolution by
SCObjectID*	**ContractType**	ContractType
SCObjectID*	**SCObject**	SCObjectName
ContractStatusID	**ContractStatus**	ContractStatusName
NextContractStatusID	**NextContractStatus**	NextContractStatusName

* The field SCObjectID can either be resolved via *ContactType* or *SCObjectName*.

Example of a ContractStatusSequence import file:

```
<ImportData>
<ContractStatusSequence>
        <ContractStatusName>Ended</ContractStatusName>
        <NextContractStatusName>Archived</NextContractStatusName>
        <SCObjectName>Maintenance contract</SCObjectName>
        <Rank>0</Rank>
</ContractStatusSequence>
</ImportData>
```

5.2.7 Currency

The object **Currency** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Currency</Name>
<Public>true</Public>
<TableName>Currency</TableName>
<PrimaryKeyNames>Code</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The currency can be resolved via the *Name* field.

Example of a Currency import file:

```
<ImportData>
<Currency>
        <Code>USD</Code>
        <Name>USDollar</Name>
        <ExchangeRate>0.5</ExchangeRate>
</Currency>
</ImportData>
```

5.2.8 ParaValueDef

The object **ParaValueDef** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ParaValueDef</Name>
<Public>true</Public>
<TableName>ParaValueDef</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Key,Value</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Key	Use these two fields to resolve an entry from an individual value list (ParaValueDef).
Value	

Example of a ParaValueDef import file:

```
<ImportData>
<ParaValueDef>
        <Key>Financing type</Key>
        <Value>Leasing</Value>
        <Text>Leasing</Text>
        <Initial>False</Initial>
        <Active>True</Active>
        <Rank>5</Rank>
</ParaValueDef>
</ImportData>
```

5.2.9 PaymentMode

The object **PaymentMode** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>PaymentMode</Name>
<Public>true</Public>
<TableName>PaymentMode</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The PaymentMode can be resolved via the *Name* field.

Example of a PaymentMode import file:

```
<ImportData>
<PaymentMode>
        <Name>custom</Name>
</PaymentMode>
</ImportData>
```

5.2.10 TaskStatus

The object **TaskStatus** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>TaskStatus</Name>
<Public>true</Public>
<TableName>TaskStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The TaskStatus can be resolved via the *Name* field.

Example of a TaskStatus import file:

```
<ImportData>
<TaskStatus>
        <Name>pending</Name>
        <Initial>False</Initial>
        <Active>True</Active>
        <Archive>False</Archive>
        <Description>Testdata</Description>
</TaskStatus>
</ImportData>
```

5.2.11 ThemesGroup

The object **ThemesGroup** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ThemesGroup</Name>
<Public>true</Public>
<TableName>ThemesGroup</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>ResponsibleEmployee</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The ThemesGroup can be resolved via the *Name* field.

Field	DependantObjectName	AK / Resolution by
ResponsibleEmployeeID	**ResponsibleEmployee**	ResponsibleEmployeeDomainLogin ResponsibleEmployeeStaffNo ResponsibleEmployeeFullName

Example of a ThemesGroup import file:

```
<ImportData>
<ThemesGroup>
        <Name>Sales</Name>
        <ResponsibleEmployeeDomainLogin>JohSm</ResponsibleEmployeeDomainLogin>
</ThemesGroup>
</ImportData>
```

5.3 Spider Contract lookup objects

5.3.1 Assignee

The Lookup object **Assignee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Assignee</Name>
<Public>false</Public>
<TableName>Contractor</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,Name,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The assignee can be identified via the *Identifier* field. Usually, the field is defined as unique.
Name	Alternatively, the assignee can be resolved via the *Name* field.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.2 Assignor

The Lookup object **Assignor** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Assignor</Name>
<Public>false</Public>
<TableName>Contractor</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,Name,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	The assignor can be identified via the *Identifier* field. Usually, the field is defined as unique.
Name	Alternatively, the assignor can be resolved via the *Name* field.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.3 AttachmentType

The Lookup object **AttachmentType** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>AttachmentType</Name>
<Public>false</Public>
<TableName>Attachment</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>TypeLookUp</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The type (AttachmentType) can be resolved via the *Name* field.

5.3.4 ContractType

The Lookup object **ContractType** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ContractType</Name>
<Public>false</Public>
<TableName>Contract</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>TypeLookUp</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The ContractType can be resolved via the *Name* field.

5.3.5 CostCentre

The Lookup object **CostCentre** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>CostCentre</Name>
<Public>true</Public>
<TableName>Core_CostCentre</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name,ProductArea,AccountingArea,FullName,MandatorID</AlternativeKeyN
ames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name **ProductArea** **AccountingArea**	In principle, these three fields are required for the identification of a CostCentre object. If the fields *ProductArea* and *AccountingArea* are not used in Spider, it is sufficient to identify the CostCentre object via the *Name* field.
FullName	This field is a combination of the following fields: *<Name> <ProductArea> <AccountingArea>*
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.6 Employee

The Lookup object **Employee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Employee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.7 InitiatorEmployee

The Lookup object **InitiatorEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>InitiatorEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The initiator of a task (InitiatorEmployee) can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The initiator of a task (InitiatorEmployee) can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.8 Folder

Zweck: Zugriff auf Verzeichnisse innerhalb einer Dokumentenakte eines Spider Objektes. Folder werden beim Importobjekt *Document* benötigt.

Das Lookup Objekt **Folder** wird nach folgenden Regeln aufgelöst:

Excerpt from the ImportConfiguration

```
Name>Folder</Name>
<Public>false</Public>
<TableName>Folder</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Path</AlternativeKeyNames>
<ImportType>Folder</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Beschreibung
Path	Ein Import wird über den Pfad identifiziert. Die einzelnen Bestandteile werden dabei mit einem „/" getrennt

5.3.9 GenericReference

Purpose: A Spider object is resolved via **GenericReference**. To do this, **Objectname** and **ObjectIdentifier** have to be specified. The **Objectname** is the name of the object class, e.g.: Employee, Contract, etc. The **ObjectIdentifier** is the identifier for resolving the object instance and depends on the object class.

The Lookup object **GenericReference** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>GenericReference</Name>
<Public>false</Public>
<TableName>sovObject</TableName>
<AlternativeKeyNames>ObjectName,Identifier,MandatorID</AlternativeKeyNames>
<ImportType>GenericReference</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
ObjectName	A general reference can be resolved using the field *Objectname*.
Identifier	A general reference can be resolved using the field *Identifier*.
MandatorID	The field filters the objects based on one mandator.

5.3.10 LegalEntity

The Lookup object **LegalEntity** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>LegalEntity</Name>
<Public>false</Public>
<TableName>Core_LegalEntity_02</TableName>
<PrimaryKeyNames>MandatorID,ID</PrimaryKeyNames>
<AlternativeKeyNames>MandatorID,Path</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
MandatorID	The field filters the objects based on one mandator.
Path	The path can be used only for identification, but not for creating a new legal entity! The path is a combination of superordinated path (ParentID) and the name. Each component is separated by a "/" sign.

5.3.11 Mandator

The Lookup object **Mandator** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Mandator</Name>
<Public>false</Public>
<TableName>Core_Mandator</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The mandator is resolved via the *Name* field.

5.3.12 NextAttachmentStatus

The Lookup object **NextAttachmentStatus** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>NextAttachmentStatus</Name>
<Public>false</Public>
<TableName>AttachmentStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The next AttachmentStatus can be resolved via the *Name* field.

5.3.13 NextContractStatus

The Lookup object **NextContractStatus** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>NextContractStatus</Name>
<Public>false</Public>
<TableName>ContractStatus</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The next ContractStatus can be resolved via the *Name* field.

5.3.14 OwnerEmployee

The Lookup object **OwnerEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>OwnerEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>

<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The owner of a task (OwnerEmployee) can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The owner of a task (OwnerEmployee) can be resolved via the *DomainLogin* if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.15 Parent

The Lookup object **Parent** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Parent</Name>
<Public>false</Public>
<TableName>Contract</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,SCObjectID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	If "unique" is defined for the *Identifier* field, this is sufficient for the identification of a superordinated contract. Otherwise, further alternative keys must be specified.
SCObjectID	The field filters the contracts based on one type.
MandatorID	The field filters the objects based on one mandator.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.16 Reference

The object **Reference** is imported according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>Reference</Name>
<Public>false</Public>
<TableName>Contract</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Identifier,SCObjectID,MandatorID</AlternativeKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
Identifier	If "unique" is defined for the *Identifier* field, this is sufficient for the identification of a superordinated contract. Otherwise, further alternative keys must be specified.
SCObjectID	The field filters the contracts based on one type.
MandatorID	The field filters the objects based on one mandator. When creating the object, the MandatorID has to be specified.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName
ReferenceID	**Reference**	ObjectName Identifier MandatorName
SCObjectID	**SCObjec**	SCObjecName

5.3.17 ReferenceObjectType

The Lookup object **ReferenceObjectType** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ReferenceObjectType</Name>
<Public>false</Public>
<TableName>ReferenceObject</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>TypeLookUp</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The type of a reference object can be resolved via the *Name* field.

5.3.18 PurchaserEmployee

The Lookup object **PurchaserEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>PurchaserEmployee</Name>
<Public>false</Public>
<TableName>core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>

        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The PurchaserEmployee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The PurchaserEmployee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.19 RequesterEmployee

The Lookup object **RequesterEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>RequesterEmployee</Name>
<Public>false</Public>
<TableName>core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>

        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The RequesterEmployee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The RequesterEmployee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.20 ResponsibleEmployee

The Lookup object **ResponsibleEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>ResponsibleEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
<PreceedingImport>
        <DependantObjectName>Mandator</DependantObjectName>
        <LookUp>true</LookUp>
        <Insert>false</Insert>
        <Update>false</Update>
</PreceedingImport>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The ResponsibleEmployee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The ResponsibleEmployee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

Field	DependantObjectName	AK / Resolution by
MandatorID	**Mandator**	MandatorName

5.3.21 SendToEmployee

The Lookup object **SendToEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SendToEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

5.3.22 SendCcEmployee

The Lookup object **SendCcEmployee** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SendCcEmployee</Name>
<Public>false</Public>
<TableName>Core_Employee</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>StaffNo,Firstname,Lastname,FullName,DomainLogin,MandatorID</Alternat
iveKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
StaffNo	The Employee can be resolved via the *StaffNo* field if this is kept up-to-date.
DomainLogin	The Employee can be resolved via the *DomainLogin* field if this is kept up-to-date.
FullName*	This field is a combination of the following fields: *<Last Name>+<First Name>+<DomainLogin>*
MandatorID	The field filters the objects based on one mandator.

* FullName consists of last name, first name and DomainLogin. All data are required. Otherwise, the resolution would have to be carried out by using the primary key.

5.3.23 SCObject

The Lookup object **SCObject** is resolved according to the following rules:

Excerpt from the ImportConfiguration

```
<Name>SCObject</Name>
<Public>false</Public>
<TableName>SCObject</TableName>
<PrimaryKeyNames>ID</PrimaryKeyNames>
<AlternativeKeyNames>Name</AlternativeKeyNames>
<ImportType>Default</ImportType>
```

Application of the alternative keys

AlternativeKeyNames	Description
Name	The corresponding SCObject can be resolved via the *Name* field.

CHAPTER 6

Notes for error evaluation

In this chapter

6.1 Schema file

During the import process, the import service creates a schema file named "SpiderProduct-Name_ImportSchema.xsd". Use this file to test expected and possible field names.

6.2 Incorrect import files

If you try to import a file which is not recognized as an XML file no output will take place. No output file nor schema file will be written. The error message can be found in the *ProductName* exception log.

If there is an import file with incorrect XML structure, or if there are other internal errors, an output file and a schema file will be written.

6.3 Verbose

- If the Verbose mode is turned on at the importer, successful as well as failed imports (data records) will be written into the output file.
- If the Verbose mode is turned off at the Importer, only failed imports (data records) will be logged.

For the Import Service, the Verbose mode can be set also in the **Config table** of the Spider Importer. The configuration is as follows:

Section: **Spider.Importer.Service.Import**

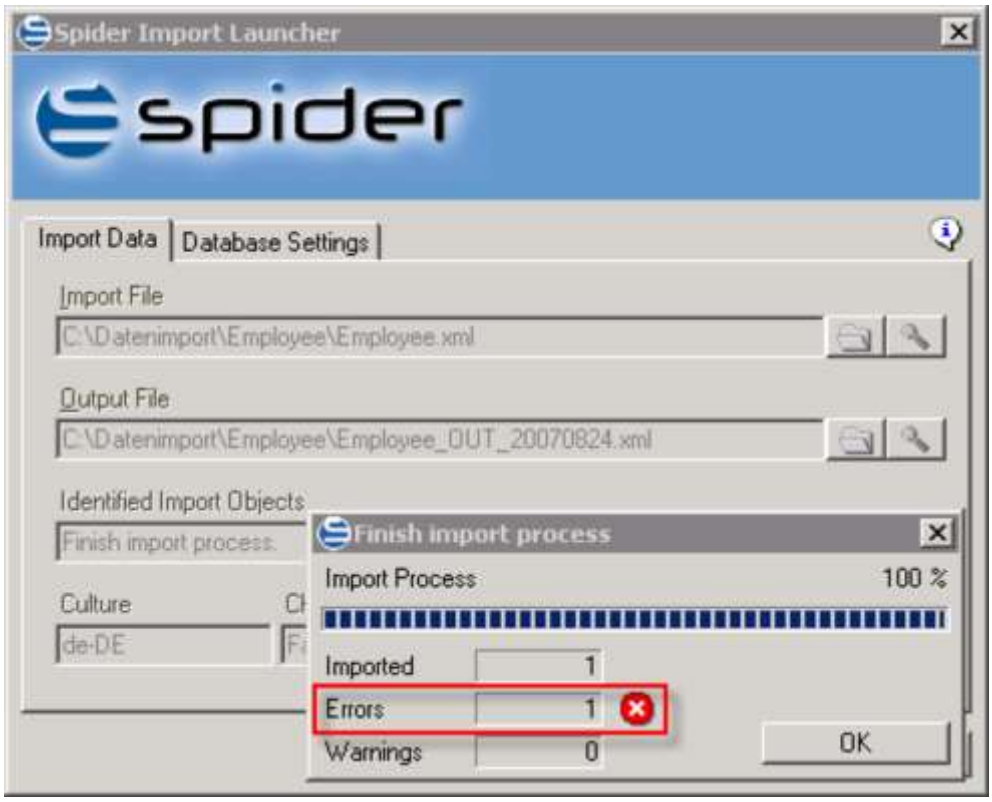
Key: **Verbose**

Value: **TRUE / FALSE**

6.4 Evaluation of feedback files

During the import process, feedback files (output files) are created. They document the import results. If the **Verbose mode** is turned on at the importer, successful as well as failed imports will be written into the output file.

Should the import process fail **(Error)**, it is recommended to inspect the **output file** to find out where the error has happened.



The following is an example for an output file in two blocks:

Block 1: The file contains the following message in the import status: **"ERR" (Error)**. Furthermore, the statement **ImportMsg (ImportMessage)** with the following error message is shown: **"Employee - The value in DomainLogin has to be unique".** This means that the import process for this object (EmployeeID=100) has failed. The error has to be analyzed. It should be checked whether e.g. the *ID* with the value of *100*, which is contained in the import file, is actually available in the database.

Block 2: The file contains the following message in the import status : **"OK"**. This means that the import process for this object (EmployeeID=2) has been successful.

Block 1: EmployeeID=100	<Employee> **<ImportStatus>ERR</ImportStatus>** **<ImportMsg>Employee - Der Wert in DomainLogin muss eindeutig sein.</ImportMsg>** <ID>100</ID> <MandatorID>0</MandatorID> etc.
Block 2: EmployeeID=2	<Employee> **<ImportStatus>OK</ImportStatus>** <ID>2</ID> <MandatorID>0</MandatorID> etc.