

# ELO XML Importer

**[Date: 2017-11-27| Program version: IM.17.110.000 with Java 1.6 or later]**

The ELO XML Importer program is a servlet for mass data import into an ELO repository. For this, the documents are stored in a communications directory together with an XML control file. The servlet checks this directory at specific intervals and processes the new control files. Documents and control files that have been processed are deleted. If a malfunction occurs, the servlet reattempts to accomplish its task for an unlimited period of time. A restart is not necessary.



**Information:** Following program version 10.00.010, the version number pattern changed: The version number of the ELO XML Importer no longer contains the main release. Instead, the main release number is added according to the license number in the status display and the log file.

## Contents

1	General.....	2
1.1	Program task.....	2
1.2	Program installation.....	2
2	Structure of config.xml .....	3
3	XML document import .....	6
3.1	Structure of the XML import control file .....	6
3.2	Explanations for individual tags .....	7
3.3	Starting the import process.....	13
3.4	Column index .....	14
3.5	Creating a structure .....	14
3.6	Creating receipt files .....	16
4	Automatic keywording update .....	17
4.1	Differences between the individual tags .....	18

## 1 General

### 1.1 Program task

The ELO XML Importer program is a servlet for mass data import into an ELO repository. For this, the documents are stored in a communications directory together with an XML control file. The servlet checks this directory at specific intervals and processes the new control files. Documents and control files that have been processed are deleted. If a malfunction occurs, the servlet reattempts to process the task for an unlimited period of time. A restart is not necessary.

### 1.2 Program installation

This program is contained in the ELO installation program. An Indexserver must already be installed for the desired repository, but not necessarily on the same computer/server. Only a rudimentarily configured XML importer is installed, the configuration of which is located in the *config.xml* file (in the installation directory, e.g. ELOenterprise\config\im-Repositoryname) and not in the database. If multiple import directories or multiple XML importers are required at the same time, the *config.xml* file must be edited later. If not, you can skip the following chapter.

## 2 Structure of config.xml

Excerpt from the config.xml file (example for an XML importer):

```
<properties>
<comment>parameters for this web application</comment>
<entry key="import1_encryption_FIBUKey">55-219-106-48-149-511-142-114</entry>
<entry key="import1_encryption_GFKey">89-106-23-789-111-654</entry>
<entry key="import1_ixurl">http://localhost:8080/ix-elo/ix</entry>
<entry key="import1_fileextension">ESW</entry>
<entry key="import1_directory1">C:\ELOenterprise\data\im-elo\import1</entry>
<entry key="import1_directory2">C:\ELOenterprise\data\im-elo\import2</entry>
<entry key="import1_movedirectory">C:\ELOenterprise\data\im-
elo\processed</entry>
<entry key="import1_username">Administrator</entry>
<entry key="import1_passwd">234-167-21-87-88-80-78-122</entry>
<entry key="import1_sigfile">false</entry>
<entry key="import1_ConfirmDir">C:\import_confirm_files</entry>
  <entry key="import1_ConfirmExt">.QIT</entry>
  <entry key="import1_ConfirmOk">Document guid:%GUID% imported</entry>
<entry key="import1_ConfirmErr">Document Import Error</entry>
<entry key="import1_ConfirmStructure">true</entry>
<entry key="import1_ConfirmEmpty">false</entry>
<entry key="import1_sleepTimeUser">120000</entry>
<entry key="import1_keywording_form_content">true</entry>
</properties>
```

The parameters in the configuration file consist of a combined parameter name (key) and a parameter value. The parameter name consists of the importer name and the corresponding parameter. Both are separated by the first underscore (\_). Combined parameters are also separated by underscores, e.g.

```
"import1_encryption_FIBUKey":
```

Here, "*import1*" is the name of the importer and "*encryption\_FIBUKey*" is the corresponding parameter. If several XML importers need to be used independently of each other, you can choose another name like import2, import3, and so on.

Entry to config.xml	Explanation
<code>encryption_name</code>	Encryption key with the password. In the control file, only the encryption key ( <b>name</b> ) is entered; see below.
<code>ixurl</code>	URL for Indexserver access. If the server is called xmlserver and the repository is named elo: http://xmlserver:8080/ix-elo/ix.
<code>directory1</code>	Exchange directory for document and control files.
<code>directory2</code>	Second exchange directory for document and control files (optional).
<code>directoryX</code>	Any number of additional exchange directories.
<code>fileextension</code>	ESW or XML (without a setting, XML import).
<code>movedirectory</code>	Directory to which the import files are moved after import. If the files should be deleted after the import, nothing is entered to the <entry> tag: <entry key="import1_movedirectory"></entry>.
<code>username</code>	ELO user name (e.g. Administrator) for authentication. Important: this ELO user must have the rights <i>Edit documents</i> and <i>Edit repository structure</i> .
<code>passwd</code>	Encrypted (recommended) or unencrypted password for the ELO user name. You can use the ELOenterprise Password Utility to generate a new encrypted password.
<code>sigfile</code>	<b>true</b> , if a signal file (.sig) is required to start the importer. The name of the signal file must be the same as the name of the import file.
<code>sleepTimeUser</code>	Idle pause in ms between the processing of the exchange directories (60000 corresponds to one minute). Values of 100, 3000, 10000, 60000, and 12000 must not be chosen.
<code>keywording_form_content</code>	<b>true</b> , if workflow, encryption key, marker (color), and filing definition should be applied from the keywording form. This only works if no entries have been made in the XML control file.

Confirm...	For a description, see chapter <a href="#">3.6 Quittungsdateien erzeugen</a> .
------------	--

## 3 XML document import

### 3.1 Structure of the XML import control file

When importing new documents, multiple documents can be entered to a single XML import control file. Provided there are no reasons to the contrary, only one document should be provided per file. This method makes it easier to recover data following an error.

The root of the XML tree consists of the *eloobjlist*. It can contain any number of *<obj>* objects. Each object generates an ELO document.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<eloobjlist ver="1.0">
<obj mode="..." dub="...">
<desc value="..."/>
<idate value="..."/>
<xdate value="..."/>
<deldate value="..."/>
<type value="..."/>
<marker value="..."/>
<sreg value="..."/>
<path value="..."/>
<dtype value="..."/>
<owner value="..."/>
<filedby value="..."/>
<fulltext value="..."/>
<arcmode value="..."/>
<encryption>...</encryption>
<acl><access type="..." name="..." subgroup="..."></acl>
<memo>...</memo>
<indexlist><index name="..." value="..."></indexlist>
<destlist>
<destination type="..." value="[...]+L1(1,4)+LA + LD + LK"/>
(for LA, LD, LK and L1(1,4), see comment under the following table
with explanations on the individual tags
</destlist>
<repl>...</repl>
<map name="userdefined"><data key="k1" value="v1"/><data...></map>
<versioncomment>...</versioncomment>
<workflowlist><workflow template="..." name="..."></workflowlist>
<structurelist><structure type="..." value="..."></structurelist>
<docfile name="...">
<sigfile name="...">
<attfile name="...">
  </obj>
</eloobjlist>
```

## 3.2 Explanations for individual tags

Tag	ELO field	Explanation (see chapter 3.1 for example)
<i>obj</i>		<p>The <i>&lt;obj&gt;</i> tag frames a document or folder. Multiple documents within a file are represented by a string of <i>&lt;obj&gt;</i> groups.</p> <p><i>mode</i> attribute:</p> <p>If folders are only to be created if they do not yet exist, this tag contains an additional <i>mode</i> attribute with a value of <i>check</i> or <i>checkMD5</i>. With <i>check</i>, the target from the first destination entry is checked first. If the object already exists at this location, the editing of this entry is canceled — it is not entered a second time. <i>checkMD5</i> checks whether the document file (if one has been configured) already exists in ELO by checking the MD5 hash value. In this case, it cancels processing.</p> <p><i>guid</i> attribute:</p> <p><i>&lt;obj guid="(B66B68360A-87F4-4AC6-874A-9FE92DF9E6)"&gt;</i></p> <p>The object is saved with the GUID "<i>(B66B68360A-87F4-4AC6-874A-9FE92DF9E6)</i>" entered to the <i>guid</i> attribute.</p> <p><b>license attribute:</b> (only in connection with GUID)  <i>&lt;obj guid="(B66B68B60A-87F4-1AC3-874A-9FE92DF9E6)" license="B620BBB2F2B39BCAB6A599D218D5D8"&gt;</i></p> <p>Normally, the ELO XML importer needs to be unlocked with a serial number to run. There is, however, an exception: it can be run without a special license if the source only provides specially designed XML files. In this case, a <i>guid</i> attribute and a <i>license</i> attribute are entered to the <i>&lt;obj&gt;</i> tag.</p> <p><i>dup</i> attribute:</p>


		<p><code>&lt;obj dup="From"&gt;</code> in connection with an index field</p> <p><code>&lt;index name="From" value="XXXX"&gt;</code>. First, a search is performed for a document in the repository with the keywording named above. If the document exists, it is downloaded and imported from the XML file, instead of the document.</p>
<i>desc</i>	Short name	The <i>value</i> attribute cannot be empty.
<i>idate</i> <i>xdate</i> <i>deldate</i>	<p>Filing date (internal date)</p> <p>Document date (external date)</p> <p>Expiration date</p>	<p>If the <i>value</i> attribute is empty, the current date at the time of importing is entered. The document date, on the other hand, remains empty.</p> <p><i>Xdate</i> with attribute <code>&lt;mode="current"&gt;</code>: The current date is entered to the document date.</p> <p>The date fields are filled with an ISO date (YYYYMMDD). The year has to be entered as a four-digit number.</p> <p>The <i>idate</i> field may contain a time in addition to the date (YYYYMMDDHHMMSS). Please note that the seconds must be entered, even though they will not be saved (you can simply enter a fixed number of 00).</p>
<i>type</i>	Keywording form	The <i>value</i> attribute cannot be left empty, as it contains either the number of the keywording form or its name. Default values of the keywording form are applied.
<i>dtype</i>	Document type	<p>You can define an object as various document types by using a value from 254 to 310.</p> <p>You can also create additional structure elements here. These can be assigned values from 1 to 253 and must not have a document entry (from ELO 10, this check is omitted).</p>



<i>access</i>	Keys	The <i>name</i> attribute contains the name. If a name cannot be assigned upon import, it is ignored. The <i>type</i> attribute can contain an <i>r</i> (read access), <i>w</i> (write access), <i>d</i> (delete), <i>e</i> (edit document), <i>l</i> (edit lists), or a combination of these five characters. <b>AND group:</b> AND groups can be defined with the <i>subgroup</i> attribute. Separate multiple groups with a semicolon – ";".
<i>acl</i>		All keys (< <i>access</i> >) are located in the < <i>acl</i> > tag. The < <i>acl</i> > tag can additionally contain the attribute <i>mode="new"</i> . This attribute indicates that all existing document permissions (keys) will be deleted from the keywording form and only the new ones from the XML control file will be entered. Assigned permissions apply only to documents, not to folders.
<i>memo</i>	Extra text	If the extra text contains explicit line breaks, these must be specified with \n or \r\n. Line breaks from the XML source are not applied to the ELO entry.
<i>marker</i>	Color marking	This specifies the color marking for the logical document entry. Only the marker number, not its name, can be entered here.
<i>sreg</i>	Version number	Defines which current version number will be displayed in the keywording dialog box. Maximum 8 characters in length.
<i>map</i>	<i>Additional information</i> in the keywording form dialog box (with predefined map domain)	In the < <i>map</i> > tag, all name-value pairs are contained within the < <i>data</i> > tag, which in turn contains the attributes <i>key</i> and <i>value</i> . The < <i>map</i> > tag can be given the <i>name</i> attribute, which creates a user-defined map domain. See chapter 16: "Maps (8.0)" of the Indexserver Programming Guide by Wolfgang Imig for more details.  Note: User-defined maps that are generated using the <i>name</i> attribute cannot be viewed or edited in the client.

<i>owner</i>	Owner	The <i>value</i> attribute contains the name of the owner, who can only be set by the administrator.
<i>Filedby</i>	Person who filed the document/folder	Can only be set with administrator rights. The user name must already exist in the repository.
<i>Fulltext</i>	Full text database	Possible values are <i>ON</i> (document is added to the full text database) or <i>OFF</i> (useful if the full text flag is set in the keywording form, but the document should not be added to the full text database).
<i>arcmode</i>	Document status (filing mode)	Possible values are: <i>FREELYEDITABLE</i> (version control disabled), <i>VERSIONCONTROLLED</i> (version control enabled), or <i>REVISION-CONTROLLED</i> (non-modifiable).
<i>destination</i>	Filing location	<p>The <i>value</i> attribute contains a filing target in the ELO "lookup index" format. The first filing location automatically becomes the main entry in ELO; all additional locations are references (see additional remarks under the table). If a filing location is defined using the format ¶folder1¶folder2¶etc., the importer creates folders as needed.</p> <p>Optionally, the filing path can also be entered in the index notation of the ELO client in order to automatically generate it from the keywording or via the document ID. In this case, the filing location must already exist.</p> <p>With the <i>type</i> attribute, you can define the keywording form for newly created folders. If this parameter is missing, the <i>Folder</i> keywording form is used.</p> <pre>&lt;destination type="Invoice" value="[...] + LA"/&gt;</pre> <p>– see below for a description of the placeholders (LA, LK, etc.).</p>

<i>index</i>	Index field	The <i>name</i> attribute contains either the number of the index field (1-51), or the group name of the index field. Since the latter is not necessarily unique (since a group entry can occur multiple times), the numerical form must be chosen in these cases. If the group name is <i>ELO_FNAME</i> , the value is entered to the 51st index field (row for the file name).
<i>structure</i>	Additional folders	<p>The <i>value</i> attribute contains a filing target that is created empty if it does not yet exist. In contrast to the destination tag, no reference is created to the document here; the target remains empty. With this <i>structure</i> tag, you can ensure that when creating a folder, a complete folder structure is created (even those that will not be filled through this import — see additional remarks below). You should use this function sparingly, since every structure path must be checked to determine it exists every time a filing process is initiated. Depending on the folder structure, this can bring with it noticeable decreases in performance.</p> <p>With the <i>type</i> attribute, you can define the keywording form for newly created folders. If this parameter is missing, the <i>Folder</i> keywording form is used.</p>
<i>versions-comment</i>	Comment	Version comment
<i>docfile</i>	Document file	The <i>name</i> attribute contains the file name of the object. It can remain empty, which creates a document without a document file. If a name is entered without a path, this will refer to the directory that contains the XML file. Every document must contain its own document file. If multiple documents exist compressed within a file, these must be first unpacked in an extra pass before the XML file is transferred to ELO. The file name will be saved in the version comment of the DocHistory table.

<i>sigfile</i>	Signature file	With this optional entry, the newly created document is given a signature immediately. The signature file must be created using the ELO signature components. A signature file can only be specified if a document file exists.
<i>attfile</i>	Attachment	Using this optional entry, the new document is created with an attachment. An attachment can only be specified when a document file exists.
<i>path</i>	Filing path	<p>The number of the filing path when filing a new document. If this field is missing, the preset from the keywording form definition will be used. If no value is entered there, the current default filing path is selected.</p> <div>  <b>Please note:</b> If an invalid value is preset, the document cannot be filed.         </div>
<i>repl</i>	Replication sets	Normally, an XML entry inherits the replication set list from the target folder. However, if you want to file the document to the chaos folder, or set up custom replication sets for the documents, you can use this tag to specify replication sets. The list contains comma-separated 32-bit integer values that result in a bit vector for the desired sets.
<i>encryption</i>	Encryption key	Name of the encryption key.
<i>workflow</i>		<p>Select an existing workflow with the <i>template</i> attribute.</p> <p>The <i>name</i> attribute sets the name of the workflow that is started.</p>

### 3.2.1.1 Comment

For the filing paths (<*destination*> and <*structure*> tags), the following placeholders can be used:

- *LA*: The filing date is applied to the filing path ("idate"). For version IM.16.070.000 and higher, if no filing date has been set in the XML file, the current date is applied.
- *LD*: The document date is applied to the filing path
- *LK*: The short name is applied to the filing path

- *L2(1, 15)*: The contents of the second index field are applied to the filing path starting with the first character and with a length of 15 characters

If the XML control file is encoded in UTF-8, enter `encoding="UTF-8"` to the first line of the XML file. Otherwise, specify `encoding="ISO-8859"` as described above.

The following characters must be entered as stated below:

```
<    &lt; ;
>    &gt; ;
&    &amp; ; Example: <destination value="[¶Monterrey&amp;Cohen]" />
"    &quot; ;
`    &apos; ;
```

### 3.3 Starting the import process

The ELO XML Importer monitors one or more directories according to its configuration. As soon as a data stream is detected, the XML file is read, the required structural elements are created in the repository, and the documents are filed. The server decides whether a new stream is available based on the exclusive availability of the XML file. A data stream can be created as follows:

The source creates an XML file.

For each document, the document file is copied to the target directory and a corresponding `<obj>` entry is added to the XML file.

The XML file is closed.

After the source has closed the XML file, ELO can start its processing.

Alternatively, the source can construct the XML file using a different extension than `.xml` (such as `.$$.$`). The source application renames the file to `.xml` when the process is complete and the file has been closed. This method is normally the most reliable, but it requires the source to be able to perform renaming.

If the source does not keep the XML file open permanently and renaming is not possible, you can alternatively set ELO to not recognize the completion of the action according to its unrestricted availability, but rather according to an additional control file.

1. The source creates the XML file `xyz.xml`.
2. The documents are written.
3. The XML file is closed.
4. The signal file `xyz.sig` is created (empty).

ELO starts processing as soon as the signal file exists.

The HTML status page in the browser presents another option for starting the import process. Click the link *Import now* to start processing manually. This will also refresh the cache of the corresponding server, which is useful if, for example, the configuration has been changed via the ELO client in the meantime (new keywording form, user, etc.).

### 3.4 Column index

In ELO, it is possible to store multiple values to an index field, which is called a *column index*. This format is then useful when a range of equivalent data is available for an entry (such as a list of order numbers for a document named *Invoice*). It is very easy to use this column index entry in the XML control file. If multiple `<index>` entries with the same name attribute exist within the `<indexlist>`, they are automatically combined into a column index.

Example: Index field 1 contains the order numbers for an invoice; the numbers 123, 140, and 210 have been entered. The section of the XML file will then look like this:

```
<indexlist>
<index name="1" value="123"/>
<index name="1" value="140"/>
<index name="1" value="210"/>
</indexlist>
```

The index entries here can be placed in any order you wish; they do not need to be sorted. Please note that an entry with the format:

```
<index name="1" value="123,140,219"/>
```

yields a completely different result. This creates a single large entry instead of a column index. With an entry in this format, it is not possible to efficiently search for order number 140 over the SQL server.

### 3.5 Creating a structure

The required filing structure is automatically created according to the `<destination>` and `<structure>` tags as required. Here you will only have limited influence on the keywording and the keywording form type of the elements created. However, there is a possibility to explicitly create the structure using custom `<obj>` entries. You will then have complete influence on the object parameters and the keywording assignment. To prevent the folder from being created again each time, and in the process incorrectly creating multiple instances of it, the `<obj>` entry must be marked as *checked*, which is done via the mode attribute pass (`<obj mode="check">`).

An example of creating a structure:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<eloobjlist ver="1.0">
  <obj mode="check">
    <desc value="Folder99"/>
    <xdate value="20041011"/>
```

```
<type value ="Development project"/>
<dtype value="2"/>
<indexlist>
  <index name="DEV_TYPE" value="Test project 2"/>
</indexlist>
<destlist>
  <destination type="Entwicklung" value="[¶XML Docs]"/>
</destlist>
</obj>
</eloobjlist >
```

In this example, it will first be checked whether a folder with the name *Folder99* already exists in the *XML Docs* folder. If so, the *<obj>* entry will be ignored. Otherwise, this folder will now be created with the information from the XML control file, where you will have great influence over the basic parameters (keywording form, date entries, etc.) and the keywording.

If the *XML Docs* folder in this example does not yet exist, it will also be created automatically.

### 3.6 Creating receipt files

Entry to config.xml	Explanation
<i>ConfirmDir</i>	Here you will enter the directory to which receipt files will be written. This entry must not have a \ at the end (as is the case for all directory entries in the XML profiles), such as E:\ELOenterprise\confirm.
<i>ConfirmExt</i>	File extension for the receipt files (.Q/T)
<i>ConfirmOk</i>	Text in the receipt file for "documents correctly filed"  You can use the following placeholders here:  %ID%: outputs the object ID of the new entry.  %GUID%: outputs the GUID of the new entry.  %DESC%: outputs the short name of the new entry.  %IX:nn%: outputs the index field nn (starting with 0).
<i>ConfirmErr</i>	Text in the receipt file for "errors while filing".
<i>ConfirmStructure</i>	If <i>true</i> , receipt file for folders.
<i>ConfirmEmpty</i>	If <i>true</i> , empty receipt files will also be written.

By using the entries *ConfirmDir*, *ConfirmExt*, *ConfirmOk*, *ConfirmErr*, *ConfirmStructure*, and *ConfirmEmpty*, receipt files can be created for the imported data sets. The receipt file will be assigned the same name as the XML control file with the file extension defined under *ConfirmExt*. A line is written to the receipt file for each document entry in the XML control file, containing the text set in either *ConfirmOk* or *ConfirmErr*.

EXCEPTION: If the option is selected to check whether the documents exist and a new document or folder is not created because of it, nothing will be written to the receipt file either. Generating an empty receipt file can be set as an option in the config.xml file. If the XML structure of the input file is incomplete or damaged, processing may halt before the end of the document. If this occurs, fewer lines will be written to the receipt file than are blocks in the XML control file.



## 4 Automatic keywording update

The bulk import service can also be implemented for automatic keywording comparisons. Here the documents will only be filed in ELO with a minimal keywording (such as a barcode). From an application, an XML file will then be created with the complete keywording information. The service automatically adds to the keywording. The format of the XML file here will essentially be identical to that of the import file.

```
<?xml version="1.0" encoding="utf-8" ?>
<eloupdatelist ver="1.0">
  <obj>
    <source value="..." />
    <desc value="..." />
    <idate value="..." />
    <xdate value="..." />
    <acl>
      <access type="..." name="..." />
      ...
    </acl>
    <memo>...</memo>
    <indexlist>
      <index name="..." value="..." />
      ...
    </indexlist>
    <destlist>
      <destination value="..." />
      ...
    </destlist>
    <workflowlist>
      <workflow template="..." name="..." />
    </workflowlist>
    <docfile name="..." />
  </obj>
  ...
</eloupdatelist>
```

## 4.1 Differences between the individual tags

Tag	Name in import data set	Explanation
<i>eloupdatelist</i>	<i>eloobjlist</i>	This difference enables the service to differentiate between both types of lists.
<i>obj</i>	<i>obj</i>	<i>mode</i> attribute:  The value " <i>versioning</i> " can be entered here. The document will then be created if it does not yet exist.
<i>source</i>	./. nicht vorhanden	With this field, the existing, minimally keyworded ELO entry will be addressed. This can be the internal ELO object ID ( <i>#1234</i> -> object ID 1234), or the repository path in the form [ <i>¶sord1¶sord2...</i> ], or one or more index fields: " <i>BARCODE=1234567;REGNR=C2Z68</i> " if you wish to select an entry with the code 1234567 in the <i>BARCODE</i> index field and an entry with the code <i>C2Z68</i> in the <i>REGNR</i> field (the group name should be specified, not the field name).
<i>desc</i>	<i>desc</i>	If the <i>value</i> attribute of this tag remains empty, the short name will be retained from the existing data set.
<i>idate, xdate</i>	<i>idate, xdate</i>	If the <i>value</i> attribute of this tag remains empty, the date information will be retained automatically from the existing record.
<i>index</i>	<i>index</i>	The index tag can also contain a <i>mode</i> attribute. If this is set to the value " <i>new</i> ", this entry will overwrite all existing entries in this index field. If this is not set, new index values will be added to the existing ones in column index format.

<i>destlist</i>	destlist	The <i>&lt;destlist&gt;</i> can remain empty, which will keep the entry in its original location in the repository. If the <i>&lt;destlist&gt;</i> is not empty, the document is moved from its original filing location to the first entry in the list. References will be created for all additional list entries. The first entry can also simply be filled with a *. In this case, the document will remain at its original location and the additional references will be created.
<i>docfile</i>	docfile	If a <i>docfile</i> is specified in this update data set, it will be attached as a new version to the logical document.
<i>attfile</i>	attfile	If <i>attfile</i> is specified in this update data set, it will be attached as a new version of a document's attachment.
<i>acl</i>		All keys ( <i>&lt;access&gt;</i> ) are located in the <i>&lt;acl&gt;</i> . The <i>&lt;acl&gt;</i> tag can also contain the attribute <i>mode="new"</i> . This attribute indicates that all existing keys have been deleted and only the new ones will be saved.
<i>dtype</i>	Dokumententyp	<p>Here you can change the document type (values <i>254 to 310</i>) and the type of folder (values <i>1 to 253</i>) (from ELO 10, the check to verify whether the values lie in the range 1 to 253 is omitted).</p> <p>WARNING: No special check will take place here to ensure you are changing a document or a folder. For this reason, please take special care to set correct values in the control file.</p>
<i>arcmode</i>	Dokumentenstatus (Archivierungsmodus)	If the document has been filed with the status <i>Non-modifiable</i> (previously <i>no changes possible, read-only</i> ), the document status cannot be changed again.

<i>map</i>		<i>&lt;map&gt;</i> can contain an attribute <i>mode="new"</i> . This will delete the old map and create a new one.
------------	--	--

All other tags will be used as in the import data set.