

1 Migrating EndNote Data into Fedora

1.1 About this document

Author

Corey Wallis, RUBRIC Technical Officer

Purpose

This technical report outlines how to use the EndNote data migration scripts to ingest EndNote records into a Fedora based repository.

Audience

RUBRIC Project Partners

Other technical staff using EndNote and Fedora

Requirements

A running instance of VITAL, or another Fedora based repository

An XML export of an EndNote Library

Python 2.4 installed on the system to run the migration

The libxml2 library, including the Python bindings, installed on the system to run the migration

The libxslt library, including the Python bindings, installed on the system to run the migration

References

RUBRIC Technical Report: Mapping EndNote data to MARCXML

RUBRIC Technical Report: Mapping MARCXML data to Dublin Core

Official EndNote website

<http://www.endnote.com/>

Official VITAL website at VTLS

<http://www.vtls.com/Products/vital.shtml>

Official ARROW project website:

<http://www.arrow.edu.au/>

Official Python website:

<http://www.python.org/>

Official libxml2 website:

<http://xmlsoft.org/python.html>

Official libxslt website:

<http://xmlsoft.org/XSLT/python.html>

Official py.test tool and library website:

<http://codespeak.net/py/current/doc/test.html>

Official utf-x website:

<http://utf-x.sourceforge.net/>

Official Subversion website:

<http://subversion.tigris.org/>

Official Fedora website:

<http://www.fedora.info/>

Documentation on the FOXML (Fedora Object XML) specification:

<http://www.fedora.info/download/2.1.1/userdocs/digitalobjects/introFOXML.html>

Official Library of Congress website on the MARCXML standard:

<http://www.loc.gov/standards/marcxml/>

Notes

The EndNote to Fedora migration script has been developed on a Linux based system. Python is a cross platform programming language and therefore the scripts should also run under Microsoft Windows, and the OSX operating systems. This has not been tested.

The installation of the Python programming language, the libxml2 and libxslt libraries, including Python bindings, is outside the scope of this technical report. Many Linux distributions, such as Ubuntu, will have these already installed.

The objects created by the EndNote to Fedora migration script have been tested using a VITAL repository. As the objects are valid FOXML objects that should be able to be ingested into other Fedora based repositories. At the time of writing this had not been tested.

The EndNote to Fedora transformation was developed using sample data from the University of Newcastle sourced from their Callista database. Therefore small adjustments may need to be made to the XSL transformations to take into account other institutions requirements.

Due to the modular design of the EndNote to Fedora data migration, these adjustments can be made without the need to modify the Python script, or the other supporting files. If modifications are to be made it is strongly suggested that a copy of the development files be checked out via subversion. This will provide you with the utf-x tests used to develop the original files and provide a basis for customisation using the same utf-x framework.

1.2 Background Information

A component of the work undertaken at RUBRIC-Central is the development of various data migration strategies. These strategies are designed to assist RUBRIC Project Partners to migrate data into, and out of, various systems. The data migrations specifically target the three institutional repository solutions under consideration as part of the project.

Interest was expressed in being able to migrate metadata stored in an EndNote library into a VITAL based repository. This technical report, and the associated Python scripts, comprise the strategy to achieve this goal.

The Python scripts create a series of FOXML objects ready for ingest into a VITAL repository using the standard Fedora based ingest tools. Therefore it is possible that the objects may be able to be ingested into other Fedora based repositories. At the time of writing this had not been tested.

The Python scripts have been developed using a unit testing approach using the testing framework provided by the `py.test` tool and library. More information about the library is available at the website listed in the references section of this technical report. These scripts have been developed using Python version 2.4, and may work with earlier versions. However this has not been tested.

The Python scripts are modular in nature and use functionality provided by modules that have been used in other migration strategies. It is anticipated that this type of architecture will allow modification and customisation as required.

A VITAL repository is the primary target for the data migration, therefore the FOXML objects contain datastreams consistent with a VITAL repository. The datastreams are as follows:

A MARCXML datastream;

A OAI Dublin Core datastream; and

A datastream containing the original EndNote XML.

The MARCXML datastream adheres as closely as possible to the guidelines provided by the ARROW project. The OAI Dublin Core datastream is also based on the guidelines provided by the ARROW project with some minor differences. The two main differences are that there is a `dc.subject` element for each subject, as opposed to one element containing all of the subjects, and the order of some information in some elements. For example the publishers name coming before the place of publication in the `dc.publisher` element.

The conversion of the EndNote XML data into MARCXML, and then into OAI Dublin Core, is achieved using XSL transformations. The stylesheets have been developed using a unit testing approach using the framework provided by the `utf-x` suite. More information about the `utf-x` suite is available at the website listed in the references section of this technical report.

Unfortunately not all of the data stored in the EndNote library for an item mapped directly to the MARCXML template provided by the ARROW project. To ensure that no data is lost the EndNote data for the item is included as part of the object.

1.3 The Python Scripts

The data migration work is carried out by a script written in the Python programming language, for more information about Python see the official Python website listed in the references section of this technical report. The following files make up the scripts used in the data migration.

configuration.xml

The configuration file defines options used by the `endnote_to_fedora.py` script.

endnote_to_fedora.py

The Python script that does all of the work

./modules/fedora_object.py

A Python module that provides a utility class for the creation of FOXML objects.

./modules/ice_utils/xml_normalize.py

A Python module that provides a method of normalising XML documents. It is used by the `fedora_object` utility class. The file is a slightly modified version of the code available as part of the ICE open source project, <http://ice.usq.edu.au>.

./xsl/endnote_to_marc.xsl

The XSL transformation used to convert the EndNote XML data into MARCXML.

./xsl/marc_to_oai_dc.xsl

The XSL transformation used to convert the MARCXML data into OAI compliant Dublin Core.

./xsl/static-information.xml

An XML file that defines three pieces of static information that is used during the XSL transformations.

1.4 Downloading the Python Script

All of the data migration scripts, and associated code libraries, modules and files, are made available via a publicly accessible website at the following URL.

<https://rubric-central.usq.edu.au/svn/Public/code/>

It is possible to download files from this website in two different ways. The first is via a standard Internet browser. The second is via a subversion client.

1.4.1 Download the Python Script via Subversion

If you have the subversion client installed you can download the Python scripts, test files, and other files used during development. The URL that you will need to check out is as follows:

https://rubric-central.usq.edu.au/svn/Public/code/legacy_code/endnote-to-fedora/development/

1.5 How to Migrate the EndNote Data

The following sections of this technical report outline the procedure for using the Python script to migrate the EndNote data into FOXML object and ingest these objects into VITAL.

It is assumed that you have installed Python, the libxml2 and libxslt libraries, including Python bindings. Specific installation instructions for these components is outside the scope of this technical report.

1.5.1 Installing the Python Script

Check out the Python scripts, test files and other files used via svn as follows:

https://rubric-central.usq.edu.au/svn/Public/code/legacy_code/endnote-to-fedora/development/

The Python script, and supporting files, can be found in the following directory

```
~/python
```

1.5.2 Configuring the Python Scripts

The configuration options for the Python script is stored in the **configuration.xml** file. Each option is outlined below, adjust the values to suit your local configuration.

input-file

This configuration option defines the location of the XML file exported from the EndNote library

output-directory

This configuration option defines where the output of the script should be stored. The directory must exist, and be empty, before running the script.

endnote-marc-xsl

This configuration option defines where the XSL file for transforming the EndNote data into MARCXML is stored. The default value should correct.

marc-oaidc-xsl

This configuration option defines where the XSL file for transforming the MARCXML data into OAI compliant Dublin Core is stored. The default value should be correct.

pid-text

This configuration item specifies the textual component of the PID for each FOXML object. To have the repository use the PID specified in the FOXML object set this option to either **test**, **demo**, or the namespace of your repository. If it is set to anything else Fedora will generate a new PID during ingest.

pid-numeric

This configuration item specifies the numeric component of the PID for each FOXML object. If the PID specified by the FOXML object is to be retained, this

value must be at least one higher than the highest PID in the repository.

label-prefix

To assist in locating imported objects using the VITAL Manager it is possible to apply a custom prefix to all object labels. The value of this configuration option is the prefix to use. If the **use-prefix** attribute is set to **Yes**, the label prefix will be applied, if it is set to **No** the label prefix will be ignored.

active-marcxml

To have the MARCXML datastream available via the VITAL access portal set this option to **Yes**. The default option, **No**, ensures that the datastream will only be available via the VITAL manager.

active-endnote

To have the EndNote datastream available via the VITAL access portal set this option to **Yes**. The default option, **No**, ensures the datastream will only be available via the VITAL manager.

active-objects

To have all ingested objects immediately available via the VITAL access portal set this option to **Yes**. If this option is set to **No**, the objects will only be available via the VITAL manager until such time as the status of the object is set to Active.

Once all of the configuration changes have been made, save the file and exit out of your editor.

1.5.3 Configuring the XSL Transformations

There are four pieces of information that are required to meet the needs of the ARROW MARCXML templates that are not available in the EndNote data. To ensure that this information is included in the MARCXML, and by extension the Dublin Core, they are stored in the **static-information.xml** file located in the **xsl** directory. The information is then called upon by the EndNote to MARCXML XSL transformation as required.

Each option is outlined below, adjust the values to suit your local configuration.

institution

This option specifies the name of the University, or institution, that the EndNote data applies to. For example, University of Southern Queensland

copyright-statement

This option specifies a short copyright statement that is applied to all of the EndNote objects.

repository-title

This option specifies the name of the repository.

dest-collection-year

This option specifies the DEST collection year that the EndNote data applies to. For example 2005.

Once all of the configuration changes have been made, save the file and exit out of your editor.

1.5.4 Preparing the EndNote Library for Migration

Note, this data migration was prepared using EndNote version 9. Other EndNote versions may not have the required capability, or may have the commands located under different names or locations.

Open the library file containing the data we need to migrate in EndNote

Click the **File** menu item

Click the **Export** option

Select the **XML** option in the **Save as type** drop down box in the **Save As** window

Choose a suitable place to save the file

Click on the **Save** button

Upload the XML file to the server that is going to be used for the data migration

Ensure the XML file is in the location specified by the **input-file** configuration option

1.5.5 Creating the FOXML Objects for Ingest

To create the FOXML objects for ingest based on the EndNote data exported as an XML file in section 5.4 complete the following procedure:

Ensure all of the configuration options are correct

Invoke the script using the following command

```
./endnote_to_fedora.py
```

Upon program completion the FOXML objects will be in the directory specified in the **output-directory** configuration option

If the program is successful the output of the program will be similar to the following:

```
Loading the EndNote XML Data...
Loading the XSL stylesheets
Converting the EndNote XML data to MARCXML...
Converting the MARCXML data to OAI DC...
Breaking the XML into fragments...
Creating the Fedora Objects...
Warning: Label for object with PID "rubric:476" has been
truncated. Max length is: 255
```

```
Warning: Label for object with PID "rubric:520" has been
truncated. Max length is: 255

Creation of Fedora Objects for import is complete

Number of objects created: 668
```

In this particular instance the label for objects with PIDs “rubric:476” and “rubric:520” exceeded the 255 character limit imposed by Fedora. These labels were therefore automatically truncated.

If an error does occur during the processing of the EndNote data an error message will be printed to the screen. This information can then be used to resolve the error condition before attempting to process the data again.

1.5.6 Ingesting the Items into VITAL

To ingest the items into VITAL complete the following procedure:

If necessary copy the FOXML object files onto the server that is running VITAL

Ensure that the FOXML object files are accessible via the **dbadmin** user

Change to the **dbadmin** user

Navigate to the following directory on the server

```
/usr/vtls/vital/fedora/client/bin
```

Ensure the FEDORA_HOME and JAVA_HOME shell variables exist. If they do not exist, sample commands are outlined below

```
export FEDORA_HOME=/usr/vtls/vital/fedora
export JAVA_HOME=/usr/vtls/vital/java
```

Invoke the following command to start the fedora-ingest utility, where **[files]** is the location of the FOXML files and **[password]** is the fedoraAdmin password

Note: This may take some time to complete

```
./fedora-ingest d [files] foxml1.0 O localhost:8080
fedoraAdmin [password]
```

Further information on the Fedora ingest utilities is available at the following URL:

<http://www.fedora.info/download/2.1.1/userdocs/client/cmd-line/index.html#ingest>

Once the ingest is complete, check the XML log file, as specified by the output of the program, for any errors

If the new objects are to be made available via the VITAL portal, ensure sufficient time has elapsed to allow the VITAL indexer to become aware of the additional objects