

Harvesting OAI-Digital Commons data to DSpace

1 About this document

Author

Bron Dye, RUBRIC Technical Officer

Tim McCallum, RUBRIC Technical Officer

Purpose

This technical report outlines how to use OAI harvest scripts to harvest items from an OAI Digital Commons repository for import to a DSpace repository.

Audience

RUBRIC Project Partners and other users of OAI-Digital Commons repositories.

Requirements

Access to an OAI-Digital Commons repository website

Python 2.4 installed on the system to run the harvest

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

An instance of DSpace for ingest

References

OAI Registered Data Providers

<http://www.openarchives.org/Register/BrowseSites>

Official Python website

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

Official libxml2 website

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

Official py.test tool and library website

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

Official Subversion website

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

Official DSpace website

<http://dspace.org/>

Notes

The OAI harvesting scripts have 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 and the libxml2 library, including Python bindings is outside the scope of this technical report. Many Linux distributions, such as Ubuntu, will have these already installed.

Known Issues

1. Over time, if your agreement/arrangement with Digital Commons expires, there could be a problem with persistent URLs. There is a URL that is imported with the data into DSpace which can link, or refer, back to the source at digital commons. The code we have completed means that the URL will still be displayed in DSpace, however modification of the DSpace display may be possible to hide this link, or it could be deactivated, but still show.
2. One other problem we came up against with the harvest from Digital Commons was the display of some character sets.
 - a. With the New Zealand Maori text, some a s had a macrons over them. The harvest step changed the Unicode representation of characters from `&#` to `&#` and this meant that some characters weren t displayed correctly for us. We simply ran the **replace_macron.py** script through the data to remove the amp; and these were corrected.
 - b. Some Hungarian text was unable to be changed because it was already poorly formatted in the Digital Commons page and the harvest actually made this formatting worse. The project team in New Zealand have located about 4 items with the problem and are manually fixing them.

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.

Interest was expressed in being able to migrate items from an Open Archives Initiative (OAI) repository into other repositories, such as DSpace or VITAL. This technical report, and the associated Python scripts, comprise the strategy to achieve this goal.

The Python scripts create an archive or directory, similar in structure to a DSpace simple archive. Within this directory are created item directories each with a **dublin_core.xml** file storing dublin core metadata, all files relevant to the OAI-Digital Commons repository item and a file listing all relevant files attached to the item. This structure forms the basis of further migration and can then be used for ingest by other repositories.

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.

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.

harvest_to_repos.py

The python script used to access the OAI-Digital Commons URL owned by the project partner

split_xml_into_archive.py

The python script used to split the output file from harvest2repos into raw xml files for individual items of the repository.

harvest_digital_commons.py

This python script harvests any related files, eg: pdf or zip

xsl_transform.py

A Python module is a common python script (located in the migration_toolkit) that converts raw metadata to DSpace dublin core metadata.

dspace_archive.py

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

replace_macron.py

A Python script that removes extra **amp;** added by Digital Commons that interferes with Maori macrons.

4 Downloading the Python Scripts

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>

5 How to Harvest the OAI Data

The following sections of this technical report outline the procedure for using the Python scripts to harvest items from the OAI repository.

It is assumed that you have Python and the Libxml2 library, including the Python bindings, already installed.

5.1 Harvesting the Items in the OAI Repository

The **harvest_to_repos.py** script is the Python script that will harvest all of the items in an OAI repository. The script uses parameters to determine where to extract the data and what to call the output file. To invoke the script enter the following command in a terminal:

```
python harvest_to_repos.py [OAI URL] [dataFileName]
eg:python harvest_to_repos.py
http://digitalcommons.massey.ac.nz/cgi/oai2.cgi outputFile.xml
```

This script creates an xml file of the entire OAI repository for the relevant institution.

5.2 Cater for Unicode errors

Following the creation of output file, the **replace_macron.py** script was run over the xml to alter any “&#” to “&#”. This was required in the case of poorly formatted characters extracted from Digital Commons. The script used as follows:

```
python replace_macron.py [dataFileName]
python replace_macron.py outputFile.xml
```

5.3 Create DSpace simple archive

To split the large xml file into individual items, the Python script **split_xml_into_archive.py** is used. This script creates a temporary DSpace archive, in the current directory. Replace the parameters as follows:

- [dataFileName] the filename of the input xml file to be converted, this is was created by the previous step.
- [archiveName] the name of the dSpaceArchive
- [xpath] match the node for each item in the larger xml file
- [tempOutputFile] the filename of the output xml file
- [wrapperFileName] (optional) if the split has to be wrapped in any tags, if not use False

```
python split_xml_into_archive.py [dataFileName] [archiveName]
```

```
[xpath] [tempOutputFile][wrapperFileName]

eg:python split_xml_into_archive.py outfile.xml
dspaceArchive //record dc_temp.xml False
```

This script will create the following:

```
dspaceArchive/

  00/

    contents

    dc_temp.xml

  01/

    contents

    dc_temp.xml
```

Each item is represented by one item directory within **dspaceArchive**. These files are numbered consecutively; first directory is called **00**, **01** and so on.

5.4 Harvest pdf or related files

The **harvest_digital_commons.py** script is the Python script that harvests any related files.

Replace the following parameters:

- [ArchiveName] the name of the archive to be accessed
- [tempOutputFile] the name of the basic metadata file

```
python harvest_digital_commons.py [ArchiveName][tempOutputFile]

eg: python harvest_digital_commons.py dspaceArchive
dc_temp.xml
```

This script will add the pdf and related files and update the contents file as follows:

```
dspaceArchive/

  00/

    contents

    01Front.pdf

    02Whole.pdf

    dc_temp.xml
```

01/

contents

dc_temp.xml

5.5 Convert basic metadata to DSpace Dublin Core

The `xsl_transform.py` script is a common script that converts the harvested metadata. This script converts the `dc_temp.xml` files in the individual Item directories of the `dspaceArchive` directory to DSpace compliant dublin core, `dublin_core.xml`.

Replace the following parameters:

- [InputFile] the filename of the input xml file to be converted, this is found within the item directory of the archive.
- [XslFilePath] the file path to the stylesheet to be used for the conversion
- [OutputFile] the filename to be used following the conversion, this will be found in the item directory within the archive
- [ArchiveName] the name of the archive to be accessed
- [RemoveInputFile] Remove or retain the input file? - True or False

```
python xsl_transform.py [InputFile][XslFilePath][OutputFile]
[ArchiveName][RemoveInputFile]
eg:python xsl_transform.py
dc_temp.xml ../xsl/recursive_digital_commons_to_dc.xsl
dublin_core.xml dspaceArchive True
```

5.6 Ingesting the Items into DSpace

Determine handle to be used

Before ingesting the items it is necessary to identify the handle of the collection that these items are to be associated with. To achieve this, complete the following procedure:

1. Visit the DSpace homepage in your favourite Internet browser
2. Click on the **Communities & Collections** link in the browse menu
3. Hover the mouse over the link for the collection you wish to import into
4. Determine the handle for the collection by examining the URL displayed in the status bar of your Internet browser. The numbers after the word **handle** are the handle for this collection.

Test the ingest into DSpace to identify errors:

1. If necessary copy the DSpace Simple Archive to the server running DSpace
2. Ensure the DSpace Simple Archive is accessible to the **dspace** user
3. Change to the **dspace** user
4. Navigate to the **bin** directory of the DSpace installation. For example on RUBRIC systems this directory is at the following location

```
/usr/local/dspace/bin
```

5. Invoke the following command replacing:
 - [eperson] with the email address associated with a DSpace eperson with sufficient privileges to add items to the repository
 - [collection] with the collection handle identified in section 5.5
 - [source] with the location of the DSpace simple archive
 - [map] with a suitable location for the map file

```
./dsrun org.dspace.app.itemimport.ItemImport --add  
--eperson=[eperson] --collection=[collection]  
--source=[source] --mapfile=[map] --test
```

6. If the ingest is successful the DSpace utility will display an appropriate success message.
7. If the ingest fails for any reason, examine the error message and take the appropriate action to resolve the error condition.

Live ingest into DSpace

1. Ensure that the DSpace Simple Archive is accessible to the **dspace** user
2. Change to the **dspace** user
3. Navigate to the bin directory of DSpace installation; RUBRIC systems use:

```
/usr/local/dspace/bin
```

4. Invoke the ingest command; replace the following:
 - [eperson] with email address associated with the DSpace eperson with privileges to add items to the repository
 - [collection] with the collection handle(eg: 123456789/23)
 - [source] with the location of the DSpace Simple Archive directory
 - [map] with a suitable location for the unique map file

```
./dsrun org.dspace.app.itemimport.ItemImport --add  
--eperson=[eperson] -collection=[collection]  
--source=[source] --mapfile=[map]
```



Regional Universities Building Research Infrastructure Collaboratively

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

5. Any ingest errors will be displayed, simply correct these and re-run the above ingest command. Make sure the map file is still unique.