RUBRIC Regional Universities Building Research Infrastructure Collaboratively

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

Harvesting ADT 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 the ADT Harvest scripts to harvest items from an ADT repository for import to various repositories.

Audience

RUBRIC Project Partners and other users of ADT

Requirements

Access to an ADT 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

Official ADT website
http://adt.caul.edu.au/

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/

Dspace website

http://dspace.org/

Notes

The ADT 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.

RUBRIC is supported by the Systemic Infrastructure Initiative as part of the Commonwealth Government's Backing Australia's Ability - An Innovative Action Plan for the Future (http://backingaus.innovation.gov.au)

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.

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 items from an Australian Digital Thesis (ADT) repository into other repositories, such as VITAL or DSpace. 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 Archive. Within this directory are created ADT item directories each with a temporary xml file storing dublin core metadata, all files relevant to the ADT item and a file listing all relevant files attached to the ADT item. This structure forms the basis of further migration and can then be used for various 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.

get_adt_html.py

The python script used to harvest items out of an ADT repository.

xsl_transform.py

A Python module that converts ADT metadata to dspace metadata and creates a dublin core .

dspace_archive.py

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

4 Download the Python Scripts 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/migration_toolkit

RUBRIC is supported by the Systemic Infrastructure Initiative as part of the Commonwealth Government's Backing Australia's Ability - An Innovative Action Plan for the Future (http://backingaus.innovation.gov.au)

5 Preparing the ADT Data

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

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

5.1 Harvesting the Items from the ADT Repository

The **get_adt_html.py** script is the Python script that will harvest all of the items in an ADT repository. It does this by capturing the header section of the individual ADT item's html page and converting the text to lowercase.

Script structure:

```
python get_adt_html.py [ADT URL]
```

Argument Definitions:

- [ADT URL] – URL of the ADT repository

Example:

```
python get_adt_html.py http://adt.usq.edu.au/adt-
QUSQ/public/index.html
```

This script creates a temporary DSpace archive, **dspaceArchive**, in the current directory. Any files the script is unable to locate, will not be created. The user is notified if the files are missing.

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

The datastreams for each object are stored in the corresponding numbered directory. For example the datastreams in the **00** directory contain three datastreams for this object. They are the two PDF files, **01front.pdf** and **02whole.pdf**, and a text file containing all of the text extracted from the two PDF files named **contents** and a temporary xml file, **dc_temp.xm**l, containing basic dublin core metadata.

5.1.1 DSpace simple archive example

```
dspaceArchive/

      00/

            01front.pdf

            02whole.pdf
            contents

            dc_temp.xml

      01/

            01front.pdf

            02whole.pdf

            03appendix.pdf

            contents

            dc_temp.xml
```

5.2 Convert basic metadata to DSpace Dublin Core

The **xsl_transform.py** script is the Python script that will 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**.

Script structure:

python xsl_transform.py [InputFile][XslFilePath][OutputFile][ArchiveName] [RemoveInputFile]

Argument Definitions:

- [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

Example:

```
python  xsl_transform.py dc_temp.xml ../xsl/adtHtml_to_Dc.xsl

 dublin_core.xml dspaceArchive True
```

6 Ingesting the Items into DSpace

6.1 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.

6.2 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.

RUBRIC is supported by the Systemic Infrastructure Initiative as part of the Commonwealth Government's Backing Australia's Ability - An Innovative Action Plan for the Future (http://backingaus.innovation.gov.au)

6.3 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.dpsce.app.itemimport.ItemImport --add

   --eperson=[eperson] –collection=[collection]

   --source=[source] --mapfile=[map]
   ```

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.