The simple persistent unique digital store (SPUDS)

University of Stellenbosch

Library and Information Services

Hilton Gibson (hgibsonATsun.ac.za)

1. Introduction

This paper introduces the concept of the simple persistent unique digital store which is an attempt to provide a means of digital object storage that will survive one hundred years into the future using open standards , technologies and formats. In addition it will attempt to make the digital object store independent of the software used to ingest, store, manage and disseminate the objects. Doing this greatly eliminates the risk of the present interdependence of the digital object store with the institutional repository management software and adds greater utility to future digital store management software systems.

"Preservation can be thought of as communication with the future. Information that is understood today is transmitted to an unknown system in the future where it will be interpreted and displayed. The future system may have not only different hardware and software, but also different standards for encoding information. Effectively, we need to send into the future not only the information (records), but also a description of the environment that is being used to manage and read those records."

Taken from: The International Journal of Digital Curation, Issue 1, Volume 3, 2008, Page 8.

2. Terminology

2.1 Persistent

On the internet the URL or URI of a digital object is crucial. This must remain persistent so that if the digital object is cited or referenced as a hyperlink in another paper it will be found and not suffer from linkrot.

We can create persistence by using a date and time based file system.

2.2 Unique

In the digital world it is very easy to copy digital objects, so how do we ensure that the digital object stored in an institutional repository is the original ?

We can create uniqueness by using UUID's for each container object and individual digital object.

3. What do we use now ?

At the moment, December 2009, the following two are the most widely used software packages for building and maintaining institutional repositories according the opendoar website.

- http://www.dspace.org with 502 installations.
- http://www.eprints.org with 261 installations.

The digital objects and store are located as follows for the above:

- `DSpace => $DSPACE_HOME/assetstore`
- `EPrints => $EPRINTS_HOME/disk0`

None of the above use a time/date based file system for storing digital objects. None of them use UUID's to create unique digital objects and stores.

In one hundred years time how can any of the above satisfy a future researcher that the digital object is unique and has remained persistently so during the years to 2109.

4. How do we do it ?

4.1 IR digital store home location standardisation

First, I propose a standard for defining the location, on the servers file system, of the digital object store. This can be defined for any institutional repository software. For example on a UNIX filesystem:

```
$IRSTORE="/home/irstore"
```

Or if there are several stores, then as follows:

```
$IRSTORE1="/home/irstore1"
$IRSTORE2="/home/irstore2"
```

4.2 Unique top level date based digital object container structure

4.2.1 Top level date and time based container

Just as there are top level domain names on the internet, so we can create persistent top level digital object store containers. For example, assume several digital objects were created on the 25th of December 2009 between 16h00 and 17h00, then we could create a top level container for them as follows:

```
$IRSTORE/2009/12/25/16/
```

This gives us year, month, day and hour granularity for creating many digital containers. This container granularity will remain persistent until the year 9999, significantly more than 100 years of persistence.

4.2.2 UUID identifier per hour per deposit

However in that hour we may want to create another container. So we append with a UUID reference and this completes the top level container name. See below for the completed container name.

```
$IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/
```

On an Ubuntu server you can easily create a UUID using the "uuidgen" command in a bash shell or xterm. UUID's can also easily be created using JAVA and Perl.

See: http://en.wikipedia.org/wiki/Uuid#Implementations for possible implementations.

4.3 Deposit of digital objects and conversion to UUID identified digital objects

The software used to ingest and submit digital objects must be able to symlink the objects using UUID's.

For example submitting three digital objects using the top level container above:

- thesis.pdf
- image1.png
- image2.png

The first thing to do is symlink the digital objects using UUID's as follows:

- `a87937c4-5e06-4140-993c-501e0fb5a29d.pdf is a symlink to thesis.pdf.`
- `ed120832-6f6b-42b6-9b32-ddbdfc4fc66c.png is a symlink to image1.png.`
- `ca51fcdf-f0d1-481b-942f-5ac07000a80d.png is a symlink to image2.png.`

Then the following are created per digital object:

1. An MD5 or SHA1 hash file.
2. An XML file to record metadata details.

Then a manifest of files contained in the top level container folder is created as an XML file in the top level folder.

This results in the following unique folder structure and content:

1. `$IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/`
2. `$IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af.xml`
3. `$IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/thesis.pdf`
4. `$IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/image1.png`
5. `$IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/image2.png`
6. `$IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/a87937c4-5e06-4140-993c-501e0fb5a29d.pdf`
7. `$IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/a87937c4-5e06-4140-993c-`

501e0fb5a29d.xml

8. $IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/a87937c4-5e06-4140-993c-501e0fb5a29d.hash
9. $IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/ed120832-6f6b-42b6-9b32-ddbdfc4fc66c.png
10. $IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/ed120832-6f6b-42b6-9b32-ddbdfc4fc66c.xml
11. $IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/ed120832-6f6b-42b6-9b32-ddbdfc4fc66c.hash
12. $IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/ca51fcdf-f0d1-481b-942f-5ac07000a80d.png
13. $IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/ca51fcdf-f0d1-481b-942f-5ac07000a80d.xml
14. $IRSTORE/2009/12/25/16/83e94b7c-5d21-444d-8ecb-97203e0898af/ca51fcdf-f0d1-481b-942f-5ac07000a80d.hash

The underlined items are symlinks.

Per item description

Item 1. TLF (Top Level Folder) = This is the top level folder in which the digital objects are stored.

Item 2. TLF.XML = This is the XML file which has a manifest of objects stored inside the container folder

Item 3. The original thesis file.

Item 4. The original image1 file.

Item 5. The original image2 file.

Item 6. The symlink of the thesis file.

Item 7. The XML file for the thesis metadata details.

Item 8. The hash of the thesis file.

Item 9. The symlink of the image1 file.

Item 10. The XML file for the image1 metadata details.

Item 11. The hash of the image1 file.

Item 12. The symlink of the image2 file.

Item 13. The XML file for the image2 metadata details.

Item 14. The hash of the image2 file.

4.4 Metadata

The XML files per digital object record the metadata details using the following XSD schemas.

- Dublin Core 2006: http://dublincore.org/schemas/xmls/qdc/2006/01/06/dc.xsd
- Dublin Core OAI: http://www.openarchives.org/OAI/dc.xsd
- MADS: http://www.loc.gov/standards/mads/mads.xsd
- METS: http://www.loc.gov/standards/mets/mets.xsd
- MODS: http://www.loc.gov/standards/mods/v3/mods-3-2.xsd
- PREMIS: http://www.loc.gov/standards/premis/premis.xsd

Think of the XSD schemas as simply forms to complete per digital object for the metadata with the details stored in the XML files. This can be done during digital object submission and updated later when doing digital store management.

4.5 Conclusion

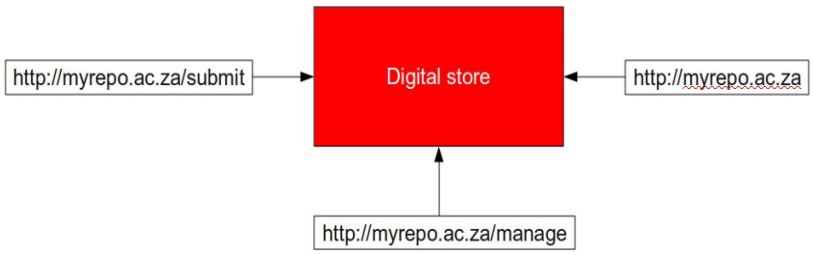
This completes the creation of one deposit into the SPUDS store. This mechanism allows the digital object detail to be stored with the digital objects independent of the institutional management software. This has some very compelling advantages which are discussed in the next section.

5. Advantages of the digital store.

5.1 Create collection metadata easily by "crawling" the digital store archive.

Because the metadata schema of the digital objects and the container is available in the file system, all that is needed to populate a digital collection management system is to "crawl" the POSIX file system, read the content of the XML files and create an index, similar to the way that Google web crawlers crawl the internet to create a searchable index.

5.2 The possibility of one digital store and several store management solutions to manage the digital store.



According to the OAI model, three main activities take place regarding a digital store.

- SIP: Submission Information Processing or Ingest.
- AIP: Archive Information Processing or Store Management.
- DIP: Dissemination Information Processing or Client access.

Having one store allows different software management systems to perform one of the above specialised functions. For example:

- SWORD for SIP
- EPrints for AIP
- DSpace for DIP

Some type of management work flow between the three systems could be developed similar to the inbox/outbox method of existing email systems.

5.3 Backing up and moving digital store archives easily.

Since everything is a file and no metadata detail is stored in a database, the digital store archive can be easily archived and compressed using the UNIX "tar" command. The resultant "tarball" can then be copied to another location as a backup regularly. You can also use the same method to easily move the digital store to another server.

5.4 Making the raw digital store archive available for indexing on the internet.

At the moment the following are used to identify digital objects and provide information about collections:

5.4.1 Registries

- http://www.doi.org
- http://www.handle.net

5.4.2 Collections

- OAI-PMH
- Sitemaps

These are not common standards or protocols used by internet search engines. This greatly impacts on the visibility on the internet of the digital object store.

5.4.3 We can solve this by using the POSIX file system and an Apache web server for simple indexing

Using the Apache web server and standard HTML it is easily possible to make the digital object store searchable and therefore indexed by search engines which greatly increases the visibility of the digital stores contents.

With the Apache web server, if you make the "document_root" equal to the $ITSTORE value mentioned earlier then the digital store is automatically crawled and indexed. For those items you do not want to be crawled then you simply remove the world readable permission on the POSIX filesystem for the item.

6. Creating authentic digital objects

Now that we have uniquely persistent digital objects the next step would have been be to ascertain their authenticity. Using a system of digital signatures and time stamping verification this may have been possible.

However, during the last three years it seems there has been a decline in confidence of digital signatures and time stamping in general. The complexity of the system also seems to have impacted on the decline in usage. The concept of authentic digital objects has been briefly addressed by :

- http://www.ijdc.net/index.php/ijdc/article/view/103/86
- http://www.diglib.org/architectures/dcoverview.htm

However, the following companies provide some sort of authenticity service, providing of course, you trust a third party to authenticate your digital objects.

6.1 Companies

1. http://www.adobe.com/security/partners_cds.html
2. http://www.entrust.net
3. http://www.verisign.com/products-services/index.html
4. http://www.thawte.com/products/index.html

6.2 References

- http://en.wikipedia.org/wiki/Key_server_(cryptographic)
- http://en.wikipedia.org/wiki/Pretty_Good_Privacy
- https://help.ubuntu.com/community/GnuPrivacyGuardHowto
- http://www.rossde.com/PGP/pgp_keyserv.html

7. Example

If resources can be made available, a demonstration system could be built using the following schemas.

7.1 Metadata standards

- Dublin Core 2006: http://dublincore.org/schemas/xmls/qdc/2006/01/06/dc.xsd
- Dublin Core OAI: http://www.openarchives.org/OAI/dc.xsd
- MADS: http://www.loc.gov/standards/mads/mads.xsd
- METS: http://www.loc.gov/standards/mets/mets.xsd
- MODS: http://www.loc.gov/standards/mods/v3/mods-3-2.xsd
- PREMIS: http://www.loc.gov/standards/premis/premis.xsd