The Australian National University: DSpace Testbed Program

Major contributions to the open source development of DSpace

During 2005, the ANU DSpace tested program has committed to provide one equivalent full time (1 EFT) staff member to develop the DSpace code base. As Scott Yeadon is already a Code Committer, this has formed a substantial part of his duties since January 2005, and will continue until the end of the year.

As a DSpace Committer, Mr Yeadon's activities include:

- release co-coordinator for DSpace version 1.3
- evaluate, provide feedback and apply patches
- support DSpace community via maillists
- provide support for evolution of DSpace architecture (including review and feedback, testing of new components and (future) development of new components)
- keep an eye on developments in repository software and repository issues beyond DSpace-specific matters

Specific projects which are significant contributions to the development of DSpace include the **User-Driven Batch Upload** and **Statistical Analysis Tool**.

Peter Raftos
June 2005

DSpace-in-a-Box

Initial releases of DSpace were reasonably difficult to install, requiring some days of time and effort on the part of experienced systems administrators. While the ANU DSpace testbed program had the time, resources and inclination to go through the procedure, there was some question as to whether smaller institutions would be both willing and able to do so. Since national outreach is a clear element of APSR, it was deemed useful to produce a simple, quick "vanilla" installation tool for DSpace, which could be fitted onto a single CD-ROM or downloaded from a web site. This installation needed to be as simple as possible.

The completed tar file (containing installation files and documentation) can be found at:

http://sts.anu.edu.au/drs/downloads/index.php.

Peter Raftos
June 2005

DSpace-Cocoon Trial: People of the Rivermouth

Preserving Compound Digital Objects

A compound digital object in the context of this paper is a digital object comprised of two or more bitstreams (files) which have an association such that all bitstreams are required, but not necessarily at one time, to form a cohesive whole. A web site, CD product, book or journal are examples of compound digital objects. A web site for example can be a collection of individual files (text, images, sound and video) associated through HTML markup. A traditional book is a collection of chapters (which could in turn be divided further) whose order and association is decided by the author/creator and often hardwired through use of a table of contents.

What are the issues in preserving and providing access to these materials?

The issues of preservation and access are not so different from those affecting simple objects such as a collection of images. Migration strategies, preservation versus access formats, audit metadata, data integrity, etc. The additional level of complexity comes in the form of future access of the item as a compound object. Proprietary software allows users to easily create impressive works, seamlessly melding any number of objects of a variety of formats together in a highly integrated (and often non-standard) way. While this is great for the user, what it means is that the associations, rendering and behaviour of the objects (i.e. those aspects which define the compound object) are within the proprietary software application. All this information is required in order to preserve the compound object yet it is stuck in a proprietary form which from experience we know does not lend itself well to preservation.

So how do we get the relationship information out? We may not be able to. But we must start thinking about how to avoid this problem continuing in the longer term.

To make preservation easier, open standard formats must be used. Besides being likely to have greater longevity than proprietary formats or standards, it also allows repository owners greater freedom to plan for migration and greater control over when to migrate. Open standards are free and open to everyone and so anyone has full access to specifications, thus the likelihood of migration software being available (and also free) is greater.

To provide for long-term preservation and facilitate wider access, the source materials need to be stored in open and standard formats within a managed repository framework. In addition, the relationships between the source items that are embedded within the Director publication and crucial to the overall cohesion of the collection also need to be captured.

People of the Rivermouth: A DSpace Case Study

Purpose

This part of the paper provides details of an approach in the preparation and submission of complex objects into a repository for ongoing preservation management. It also covers a possible mechanism for the creation of new complex items, such as learning objects or even deployable web sites, directly from the repository.

The paper also covers some of the issues encountered while dealing with more complex objects and possible solutions to some of these issues. Many of these issues will be familiar to those involved in preservation or digital asset collection and management.

Background

Over the past fifty years, various scholars from a range of disciplines have conducted research in relation to the Anbarra people of northern Arnhem Land - their culture, society, history, land and environment. There was recognition from within the research community that there was something extraordinary with the availability of such a large body of material centred on a small community of Aboriginal Australians. For example, musicologists had published recordings of and written papers on the nature of Anbarra song-cycles; Kim McKenzie's 1978 film *Waiting For Harry* showed Anbarra mortuary rites; a great body of visual materials had been collected (especially by Rhys Jones and Betty Meehan) from the late 50s onwards. This recognition of such a rich and diverse collection of materials created a desire to give an overall shape to past and ongoing research, and to disseminate the information more

widely. These two activities were to be the focus of the *Rivermouth* project.

In the late 1990s the Australian Research Council (ARC) provided majority funding for the *Rivermouth* project. At the core of the project were the Joborr Texts, a collection of twenty Gidjingarli texts created by Frank Gurrmanamana in 1960 to explain aspects of Anbarra culture to leading anthropologist Les Hiatt. Les recorded these texts in Gidjingarli along with his own notes, and the translation of the texts was one of the first exercises to be undertaken by the *Rivermouth* project.

The materials were collated over a four-year period culminating in the release of a combined CD and book. In 2002, *People of the Rivermouth: The Joborr Texts of Frank Gurrmanamana* CD and accompanying book were released under joint publication by the Australian Institute of Aboriginal and Torres Strait Islander Studies (AIATSIS) and the National Museum of Australia.

To understand the significance and importance of this work, it is important to understand the significance of the Joborr Texts. According to Kim McKenzie, Frank had a strong pride in his culture and a desire to develop an understanding of his culture by those outside it. He probably saw a relationship with anthropologists and film-makers as a perfect vehicle for taking his culture to the outside world.

The Joborr Texts are unique in that they are not traditional or prescriptive stories – they are the creative output of a master raconteur describing his culture at a point in time, a series of stories told to Les Hiatt to explain Anbarra culture. What started as an exercise where Les asked Frank to explain the act of bestowal became an intense period of creativity on Frank's part. As Les explains: “In the course of a month [Frank] set out the main institutions of his own culture in the form of a series of discourses – not between known, living or deceased individuals – but among anonymous persons representing the kinships statuses involved in the life-cycle of a male Anbarra” - *Frank Takes Over*, CD.

These texts were recorded and published on the *Rivermouth* CD as a series of plays, each play communicating a different aspect of Anbarra culture through the interactions of its cast.

Betty Ngurrabangurraba, one of Frank's daughters, has taken it upon herself to continue Frank's work in telling the world about Anbarra culture: “My Dad used to tell us not to leave our culture and ceremony, to keep it in our mind. And when we get old, to teach our children to hold that culture, to not lose it.” - *My Dad, Forty Years On*, CD.

Over time, the culture of the Anbarra has inevitably been affected by white occupation. In “Forty Years On”, a reflection by Betty on her culture, Betty observes: “White men are coming and telling us what to do. We keep forgetting our culture and Joborr. ... Old people have already died, you know”. Of the young Anbarra Betty observes “Some, they know. And some, they don't”. “I should be here with my family's land. Keep the culture and Joborr with me. Hold it tight.”

Preserving Rivermouth

The *Rivermouth* CD contains all twenty Joborr Texts combined with still images, video, audio and textual material (referred to hereon in as source materials and digital objects). The CD was created using Macromedia's Director on a Macintosh. While Director enabled the creation of an excellent interactive CD publication and valuable resource, both the Director and CD format are not suitable for long-term preservation. Retaining *People of the Rivermouth* in a Director CD-only format only provides short-term access to this valuable and unique collection of information. Some issues associated with the long-term viability of continued access via the Director CD format and the CD media itself:

- The granularity of the source material is lost since it becomes integrated into a single Director product;
- Director is proprietary software and as such is unlikely to remain static very long. For the CD to remain readable it is likely to require a greater level of management (migration, product tracking) to maintain accessibility than would be necessary if using open standards.
- Operating systems do not remain static. At the time of writing if using the CD with a Mac the CD can only be read if the Mac Classic (Mac OS 9 emulation capability) is installed;
- Director does not create its product using open standards making reverse-engineering or extracting materials from a Director CD in the future difficult if not impossible;
- Director serves a specific purpose and does not provide for granular access, preservation and

management of source materials;

- CDs will not be a dominant media in the long-term. CDs and their reader/writer hardware and software will continue to exist and evolve, but will ultimately be surpassed by newer and better technologies incompatible with current technologies.

To provide for long-term preservation and facilitate wider access, the source materials need to be stored in open and standard formats within a managed repository framework. In addition, the relationships between the source items that are embedded within the Director publication and crucial to the overall cohesion of the collection also need to be captured.

The Repository and Complex Objects

As part of the Australian Partnership for Sustainable Repositories (APSR) project, The Australian National University (ANU) is establishing a testbed repository using the DSpace repository software. While a testbed in the APSR sense, the ANU DSpace instance is intended to mark the start of the implementation of an ANU institutional repository. DSpace was chosen as the repository software primarily because it has an established community around it, is functional "straight out-of-the-box", is a broad spectrum repository and is Open Source Software (OSS).

People of the Rivermouth is an example of what is considered a complex object. We consider a complex object to be any object comprising two or more bitstreams that have an association such that all bitstreams are required, but not necessarily at one time, to form a cohesive whole. The *Rivermouth* CD is itself a complex object made up of many bitstreams and their associations, all integrated within a proprietary rendering engine. The *Rivermouth* CD itself is not ideal for preservation - while the raw binary data can be preserved, access capability certainly cannot be guaranteed. Not only will underlying hardware technology change potentially rendering older software unusable, but as time progresses, software to drive older versions of current or surpassed proprietary products certainly will not be available. The only guarantee that could be made by a managed repository is that the raw binary data can be preserved, but access capability almost certainly not. It is difficult to see any value in preserving an interactive multimedia product that we know will only be viewable in raw binary form in even a few years' time.

Fortunately in this instance the CD's creator, Kim McKenzie, still had the raw source materials (digital objects) used for the CD. Kim had also the foresight to maintain the relationships between digital objects in a modular set of simple structured text files using the Director configuration syntax. These text files are very structured and so easy for a computer program to parse and reformat. Had only the Director CD itself been available the ability to retrieve all its individual digital objects and their relationships is doubtful.

A quick review of the source material showed that while the *Rivermouth* CD is a very complex product, at its core is a collection of individual digital objects and their relationships rendered in a specific manner. Perhaps all or a great many complex objects could be looked at in this way which means the strategy taken for managing the *Rivermouth* material in DSpace is applicable on a much wider scale. For example, books and journals can be reduced to a collection text documents and images; a book, its chapters, sections and pages can be created using relationships between the text and image objects.

Besides making the problem simpler, reducing complex objects to their constituent parts for storage in a repository also means that the constituent parts can not only be discovered individually but can be used in the future in ways not considered or not possible now. It also allows individual objects to be combined in different ways for access through different media types. How to provide a framework for this?

Repository Formats

Ideally the digital objects destined for the repository should be of high quality, in a non-lossy (if compressed) open format. It is also important to consider future use of digital objects. While a small JPEG taken with a digital camera may be perfect for the task at hand, the choice of this format could preclude it not only from long-term preservation but also block alternative avenues of access (e.g. in generation of derivatives of adequate quality for delivery in another medium).

For example, uncompressed TIFF and JPEG2000 are suitable for images; XML is perfect for text and documents. While PDF is a common document format, it's important to remember that Adobe owns the specification and currently provides a free license for generation and reading of PDFs. This is not the same as an open standard and its status as a cost-free or supported format may not last long into the

future. This is a real danger in adopting any proprietary format for long-term preservation.

Repository policies will need to address how to handle digital objects submitted in non-preservable formats (e.g. MS Word). For example, defining the level of preservation support based on format (as per the DSpace Known, Supported, and Unknown format classifications) may be a way of encouraging use of more open and accessible formats, or at least a recognition of a need to convert formats prior to import into a repository. ANU is still developing policy however it is likely to be in-line with the DSpace classifications and will also likely offer automatic conversion for some formats.

The digital objects in *Rivermouth* were provided to the DSpace team in the following formats:

- images – JPEG
- video – in process
- audio – WAV (originally in AIFF but WAV supplied by Kim on request from DSpace team)
- text files (Joborr Texts, glossary, cast information) and relationships – ASCII text files

***Defining* Rivermouth**

The disassembly of a complex object such as *Rivermouth* prior to storage in a repository and the capture of its relationships in a simple and open form provides the possibility of a variety of outputs being generated from the same materials. With *Rivermouth* for example, a CD product, deployable website or book can potentially be created from the same raw materials passed through different paths of a rendering or publishing process.

Extending this idea further, using relationship information we can combine/relate any object within (or outside) the repository. If a vocabulary could be developed for not only defining known, existing complex objects, there is no reason this same vocabulary cannot not be used for creating new objects out of objects already contained within the repository (and in theory outside the repository as well). If this vocabulary could be expressed using XML (e.g. RDF or with RDF-like concepts) then Open Source processing tools (parsers, viewers) and even entire processing frameworks (e.g. Cocoon) immediately become available to work with the material.

What does this involve? Creating a *Rivermouth*-specific process is fairly simple. Creating a more generic framework takes more time and effort, but not overly complex.

XML is the perfect format for the well-structured *Rivermouth* texts and relationships. Converting the texts to a generic XML markup was performed using XSLT stylesheets, however, capturing the important semantic meaning of the content in these texts through elements and attributes for the most part cannot be automated. Nor can capturing the relationships. Both these aspects require preparation work by the repository submitter.

Definition Files

Importing the objects into DSpace was performed using a variant of the existing Batch Importer developed at ANU. The Batch Importer uses the DSpace item importer but performs all the setup and allows metadata mapping to be specified in an XML file.

All the *Rivermouth* objects were loaded into a single collection.

Scott Yeadon
June 2005

ANU Institutional Repository Policy: draft

Demetrius – Institutional Repository

Title	Institutional Repository Policy
Purpose	The Digital Resource Services program within the Division of Information has created the following policies and procedures pertaining

	to various aspects of managing the ANU's Institutional Repository, known as Demetrius.
Relevant to	All contributors to the ANU's Institutional Repository.
Responsible Officer	Director Scholarly Technology Services, Program Leader Digital Resource Services
Introduced	XXXXXX
Review Date	XXXXXX
Related Policies	
Keywords	digital repository, DSpace, institutional repository, digital preservation, Demetrius, digital archiving

Name of Repository
Demetrius, the ANU's Institutional Repository

The repository's URL
http://dspace.anu.edu.au

http://demetrius.anu.edu.au

Service Provider
The Digital Resource Services program, Division of Information.

Description of the Service
The ANU's Institutional Repository is a professionally managed digital repository service that aims to make widely available the output of the University's research and teaching activities while at the same time preserving it for long term access. The repository is based on the open source software platform DSpace and provides a submission mechanism, long-term stable storage, a system of persistent identifiers and a mechanism based on the Open Archives Initiative, OAI, protocol for metadata harvesting that enables federation with other information providers. Access to stored materials is possible, but may also be limited at the request of the relevant Community.

Copyright holders
Creator(s) or their agent(s) who give irrevocable, royalty-free permission to the ANU to distribute for educational and research purposes via Demetrius.

Contact
Email address: dspace@anu.edu.au

Location: Digital Resource Services, Scholarly Technology Services, Division of Information, WK Hancock Building - East Wing [122], The Australian National University

Definitions
Community

The repository is organised into a number of distinct communities. A Community is an organisational unit of the University with a teaching or research focus. e.g. faculties, departments, centres, research units, etc. A Community must have long-term stability, and be able to assume responsibility for setting Community policies. Each Community must be able to assign a coordinator who can work with Division of Information staff.

Sub-community

Exists within a Community or within other Sub-communities or within a Collection.

Collection

Each Community can have as many Collections and Sub-communities as their content defines.

Item

An item is a logical content object. It is composed of:

- One or more Bitstreams (a Bundle)
- Metadata defining those Bitstreams
- A distribution license pertaining to the IR's use of the Item
- A persistent identifier. Currently a handle

Bitstream

Bitstreams are the individual files and it is at the Bitstream level that preservations treatment would be applied if necessary.

Bundle

Logical collection of Bistreams within an Item.

Tombstone

Record that an Item existed and has either been withdrawn or superseded.

Complex Object

Any digital object made up of a number of files, *and the relationships between those files*. A web page or multimedia presentation is both examples of complex objects. Also known as a *Compound Object*.

Institutional Repository

Central facility to store, safeguard and make discoverable and (as appropriate) retrievable, digital scholarly materials created or generated by ANU staff or associates. Consists of staff (Digital Resource Services program personnel), services, software tools, infrastructure and policy framework.

Demetrius

Synonym for *Institutional Repository*.

DSpace

Open source repository platform. The software at the heart of Demetrius.

Metadata Policy

The repository uses a subset of Dublin Core metadata elements, http://dublincore.org/, to create the repository interface features. This set is used across all Communities and Collections. Additional collection-specific metadata can also be specified.

Third parties may harvest metadata from the archive for end-user services by arrangement with Demetrius and Community administrators.

Submission Policy

Communities set appropriate policy for content for inclusion in the repository.

Content Guidelines

All Communities and Collections are visible to the global Internet. Access restrictions based on a user name and password system can be applied to limit access to items or bitstreams.

1. The work must be produced, submitted or sponsored by ANU teaching and research areas.
2. The work must be education- or research-oriented.
3. The work must be in digital form at the time of submission.
4. The work should be complete and ready for distribution.
5. The author/owner should be willing and able to grant ANU the right to preserve and distribute the work.
6. If the work is part of a series, other works in that series should also be contributed so that the repository can offer as full a set as possible.
7. The content in the archive is subject to the general publication policy of ANU.

Format Support

The repository supports a wide range of content types: text, images, audio and video.

- Everything put in DSpace will be retrievable.

- We will recognise as many file formats as possible.
- We will support as many known file formats as possible.
- When a file is uploaded to DSpace, we assign it one of the following categories:
  1. **Supported.** File format is recognised as appropriate for archiving: generally, file is non-proprietary.
  2. **Known.** File format may not be most appropriate for archiving; however, the format is extremely common.
  3. **Unknown**. Not recognised. Storage is at depositor's own risk.

Format support will follow the principles developed by the Australian Partnership for Sustainable Repositories (APSR) in the *APSR Sustainability Issues Discussion Paper* (Bradley, 2005).

- In general, *supported* files are open source and recognised by the international community of repositories and cultural institutions as being stable and reliable for archival purposes. DRS undertakes to convert such files for the depositor should they become obsolescent; however, DRS is not responsible for the downstream effects of changing file formats.
- *Known* files are common and generally have well-understood methods for access or otherwise working with them. They are not considered to be workable long-term archive formats. DRS *may* undertake to convert such formats or provide assistance should it become necessary; this will be on a "best effort" basis.
- Unknown files are everything else. DRS is prepared to assist depositors in converting such files to more appropriate format *before* deposition: the client has the right to waive this assistance and deposit unknown file types, but DRS *will not* undertake any conversion work in future.

Note that support levels for file formats are based on their archival properties, *not* ease of presentation for the user. Presentation of stored materials is discussed below.

Change of format support levels

File formats may be upgraded or downgraded at the discretion of DRS or some future supervisory body. This will be done in consultation with Communities, and the international repository and cultural institution groups. Effects of such a change will be advertised well in advance. It is not expected that such upgrades/downgrades will be done lightly or commonly.

Community & Collection Policies

The ANU's Institutional Repository is a partnership between ANU communities and the ANU Division of Information.

Community responsibilities

A Community agrees to:

- arrange for submission and description of content
- make decisions about Community and Collection definitions
- notify the Division of Information of organisational changes affecting submissions
- reply to annual reconfirmation of Community information
- understand and observe DOI policies relevant to the repository, and educate Community submitters regarding these policies
- ensure appropriate permission for Items submitted when copyright owner is other than author(s) or the ANU
- decide upon a submission workflow for each collection
- Be responsible for ensuring that, on deposition, the material and its metadata is complete and correct

Community rights

A Community retains the right to:

- decide policy regarding content to be submitted (within repository guidelines)
- decide who may submit content within the Community

- limit access to content on a Collection-by-Collection basis. Detailed access rules will not be undertaken by the repository
    - Access to Items may be limited; however, no Community has the right to limit access to its metadata or tombstones
- receive a copy of submitted content upon request
- withdraw items and collections (as outlined in "Withdrawal Policy.)
- approve addition of or elimination of Sub-communities
- Make limited customisations to Community web pages (within the limits available through the DSpace software)

Division of Information responsibilities

DOI (specifically, the DRS) agrees to:

- retain and maintain content submitted to the repository
- distribute content according to community decisions. Unless stipulated otherwise, access will be granted to the general public
- preserve content using accepted preservation techniques
- provide access to repository research
- notify communities of significant changes to content, e.g. format migration
- if DOI ceases to support the repository, return collections to existing Communities and transfer to ANU Archives Collections of Communities that have ceased to exist

Division of Information rights

DOI (specifically, the DRS) retains the right to:

- redistribute, sell or amend metadata for Items in the repository with prior, written approval of the relevant Community
- refuse or de-accession items or Collections under certain circumstances – as outlined in "Withdrawal Policy".
- renegotiate terms of original agreement with Communities
- perform appraisal for long-term archiving when Communities cease to exist or within thirty years of the creation of a Collection
- move Collections to reflect current understanding between ANU and Communities
- migrate items if Supported format is in danger of obsolescence
- set quotas (size of files, number of items) to determine what constitutes free service and after which point to charge a fee.
- Charge fee for activities requiring extensive centralized support from the Division of Information (e.g. large amount of de-accessioning)

ANU responsibilities

ANU will:

- Set policy at the Institute level regarding issues that affect the repository, e.g. copyright rules, thesis requirements, etc.
- Support functions mandated by existing policies.

Withdrawal of items from the Repository

The Division of Information foresees times when it may be necessary to withdraw Items from the repository. It has been decided that under some circumstances items will be removed from view, but to avoid loss of the historical record, all such transactions will be traced in the form of a note in the <Description.provenance> field of the Dublin Core record. The content of the note should be one of

the following:

- "removed from view at request of the author"
- "removed from view at ANU's discretion"
- "removed from view at Division of Information's discretion"
- "removed from view by legal order"

Since any repository item that has existed at some time may have been cited, Demetrius will always supply a tombstone when the Item is requested, which will include the original metadata (for verification) plus one of the above withdrawal statements in place of the link to the object. The Item metadata should be visible, and may be searchable. These Items *may* also be made unavailable for metadata harvesting.

Privacy Policy

The personal information we received through Demetrius is used solely for purposes of the functioning of the system, and for the specific research purposes described below.

This system collects personal information from:

- users involved in the submission of content and metadata to the repository
- users who subscribe to the repository alerting service

Personal information collected by the repository will not be used for any commercial or philanthropic purpose not directly connected with or approved by the ANU. (or: Will be used in accordance with the ANU's privacy policy).

We do not disclose information about individual visits to the IR web site, or personal information, to any outside parties except when we believe, in good faith (i) that the law requires it, or (ii) that disclosure is necessary to protect the rights and property of repository users.

Any Repository records used in a publicly accessible forum, such as demonstrations, presentations, or research papers, will be scrubbed of specific references to real people and personal information.

Copyright and access

The copyright status of each item in the Repository shall be identified. The copyright may be identified at the collection level (all content in the collection having the copyright conditions) or may be identified at the item level. The objective is that the archive content can be freely used for purposes of teaching and research. The Creative Commons statement of copyright is favoured <http://creativecommons.org/>.

Specialised collections might have limited accessibility. For example, ethnographic items might have a particularly cultural sensitivity that would limit their availability.

Cost Model

The Division of Information may from time to time determine a charging model for the repository, which might involve a charge for the storage (GByte) used by a Community. For the moment no such charges are being levied.

Demetrius will not support, for the foreseeable future, pay-per-view, pay-per-download or any other charging mechanism levied on visitors to the DSpace installation web site. Copyright holders may of course choose to limit access from the IR and charge for access via their own web site or other system: this is not part of the IR's responsibility.

Publishing/Presentation

In general, the IR is *not* a publishing facility. The primary purpose of the IR is to provide a safe and reliable facility for storing research and teaching resources for the ANU and the campus community. Archival file formats will always be given preference over presentation formats, where these differ.

Where complex objects are stored in the IR, it will not be the responsibility of Demetrius to reconstruct and present these. However, DRS staff may be assigned (at the discretion of the DRS) to assist with the online or offline publication of complex objects stored in the IR or to undertake such projects in order to develop useful techniques or skills.

Clearly, however, there will be a need to make available derivative versions of bitstreams stored in DSpace. Examples include:

- Thumbnail or slightly larger preview versions of TIFF files (TIFF files are not viewable in web browsers)
- MP3 samples of larger sound files

Provision of these will be at the discretion of DRS, in negotiation with individual Communities. Under no circumstances will these derivative files be considered part of the deposited Item.

Digitisation

The IR is primarily concerned with materials that have been born digital or have already been digitised. DOI does not provide formal resources or facilities for the rest of campus to convert analogue materials to digital form. However, the IR will provide advice on appropriate file formats or other characteristics.

File formats

Summary document to be made available on ANU web site.

Current Demetrius copyright licence

Non-Exclusive Distribution Licence

In order for Australian National University (ANU) to reproduce, translate and distribute your submission worldwide, your agreement to the following terms is necessary. Please take a moment to read the terms of this licence.

By submitting this licence, you (the author(s) or copyright owner) grant to the ANU the non-exclusive right to reproduce, translate (as defined below), and/or distribute your submission (including the abstract) worldwide in print and electronic format and in any medium, including but not limited to audio or video.

You agree that ANU may, without changing the content, translate the submission to any medium or format for the purpose of preservation.

You also agree that ANU may keep more than one copy of this submission for purposes of security, back-up and preservation.

You warrant that the submission is your original work, and that you have the right to grant the rights contained in this licence. You also warrant that your submission does not, to the best of your knowledge, infringe upon anyone's copyright, or contain any libelous or defamatory material.

If the submission contains material for which you do not hold copyright, you warrant that you have obtained the unrestricted permission of the copyright owner to grant ANU the rights required by this licence, and that such third-party owned material is clearly identified and acknowledged within the

text or content of the submission.

IF THE SUBMISSION IS BASED UPON WORK THAT HAS BEEN SPONSORED OR SUPPORTED BY AN

AGENCY OR ORGANISATION OTHER THAN ANU, YOU WARRANT THAT YOU HAVE FULFILLED ANY

RIGHT OF REVIEW OR OTHER OBLIGATIONS REQUIRED BY SUCH CONTRACT OR AGREEMENT.

ANU will clearly identify your name(s) as the author(s) or owner(s) of the

submission, and will not make any alteration, other than as allowed by this

licence, to your submission.

Peter Raftos
June 2005

Project reviews and case studies

User-Driven Batch Upload

Version 0.1

The default metadata capture and upload interface provided in DSpace is too specific and does not lend itself to multiple simultaneous customised interfaced (very few collections have exactly the same requirements). Uploading many files with their metadata can be a laborious procedure. This batch uploader bypasses these problems.

Introduction

To provide a means for users to upload new or legacy collections in an efficient manner, a mechanism for batch uploading to the DSpace repository via a Web interface is required. Currently batch loading is performed by members of DRS, a mechanism which is not scalable in the long term. Current configuration and preparation scripts are written in Perl and are one-off scripts that have been modified for each collection requiring importing. The functionality of the scripts will be re-organised and re-written in Java and requirements for users to prepare their data will be documented. The onus will thus be put back on the users to prepare their data in a way DSpace will accept it.

Implementation Plan

Due to new technologies being experimented with, the batch update process will be delivered in an incremental manner. Increments will be defined in a manner that provides a useful and logical breakdown of development tasks and each increment will generate a deliverable. Not all increments will provide a deliverable that is useful to end-users.

The following outlines the proposed development approach, broken down into increments.

Increment 1 – Documentation, Metadata and Content Preparation

Generation of planning documentation and basic functionality for user-driven configuration of user metadata to Dublin Core. Development of interfaces to support all preparation and testing functions

Increment 2 – Upload to DSpace

Integration of existing DSpace batch import and production deployment.

Increment 3 - Finalisation

Lower priority/supporting functions, not core to allowing users to load material into DSpace.

Increment 4 – n – Additional Functions

Additional functionality required arising out of production use or advanced features for use with DSpace 1.2

Technical requirements

Cocoon will be used to manage web page flow and session information where appropriate.

Cocoon Forms will be used for defining data entry screens.

XHTML will be used for non-form pages.

Java will be the programming language used. Technologies used by Cocoon (XSLT, formflow language, etc) will be used where appropriate. Design will ensure as far as possible that reusable functionality is not tightly embedded in Cocoon, so as to maximise re-use potential. Java Servlets are likely to be the technology used when implementing re-usable functionality.

Work required on DSpace

Work breakdown is as follows:

Increment 1

Document and review use cases

Document a batch upload component design based on the use cases

Design and develop user interface based on use cases (including technology proof-of-concept using Cocoon forms)

Generator/Transformer/Servlet development

Internal Acceptance Testing (DRS testing only)

Increment 2

Integrate existing DSpace batch import functionality

User Acceptance Testing

Additional development based on user feedback

Production Deployment

Increment 3 (underway as of June, 2005)

Will be assessed on completion of increment 2 but may include:

User Interface/Functionality changes based on user feedback

Automatic derived image generation (default image profile)

Customised derived image generation (format, dimensions, footer text)

Provide METS/MARC import option

Provide integration of batch upload with workflow system

Statistical Analysis Tool

DSpace logs access to Items in held Collections on several dimensions. However, there is no tool to filter, analyse and represent the data. As access rates are of interest to anyone with a Collection, it was decided to develop a simple and robust statistical filtering and analysis tool on behalf of the DSpace community.

Introduction

Requirements and specifications:

The purpose of the exercise was to produce a DSpace extension which allows the on-demand generation of meaningful statistical information for DSpace administrators and Collection owners.

Users should be able to add new or modify existing (predefined) queries.

Significant dates

- First contact with client: Dec. 2004
- Tested on **dev** server: May 2005
- Live on **DSpace**: May 2005

Technical requirements

The application must be fully compliant to the current DSpace architecture.

Should be loosely coupled.

The technology used must be open source.

Display/rendering requirements

Output specification:

the output specifications dependent on specific DSpace instances, so the system should produce generic raw XML data and should provide a mechanism where customisation layer can be specified.

Work required on DSpace

The following components were added or modified:

Postgres Database:

three new Tables: item_view_log

bitstream_view_log

temp_log

stored procedure (trigger function) : log_processor, trigger will filter out events of interest and put into specific event tables (default implementation being item_view and bitstream_view events)

Log4j were used to record logs into the Postgres database.

An **XML** document "Querylist.xml" to specify predefined queries.

XSLT stylesheets to transform the raw data into required output.

The predefined ones used to generate raw XML, HTML table or dataSet

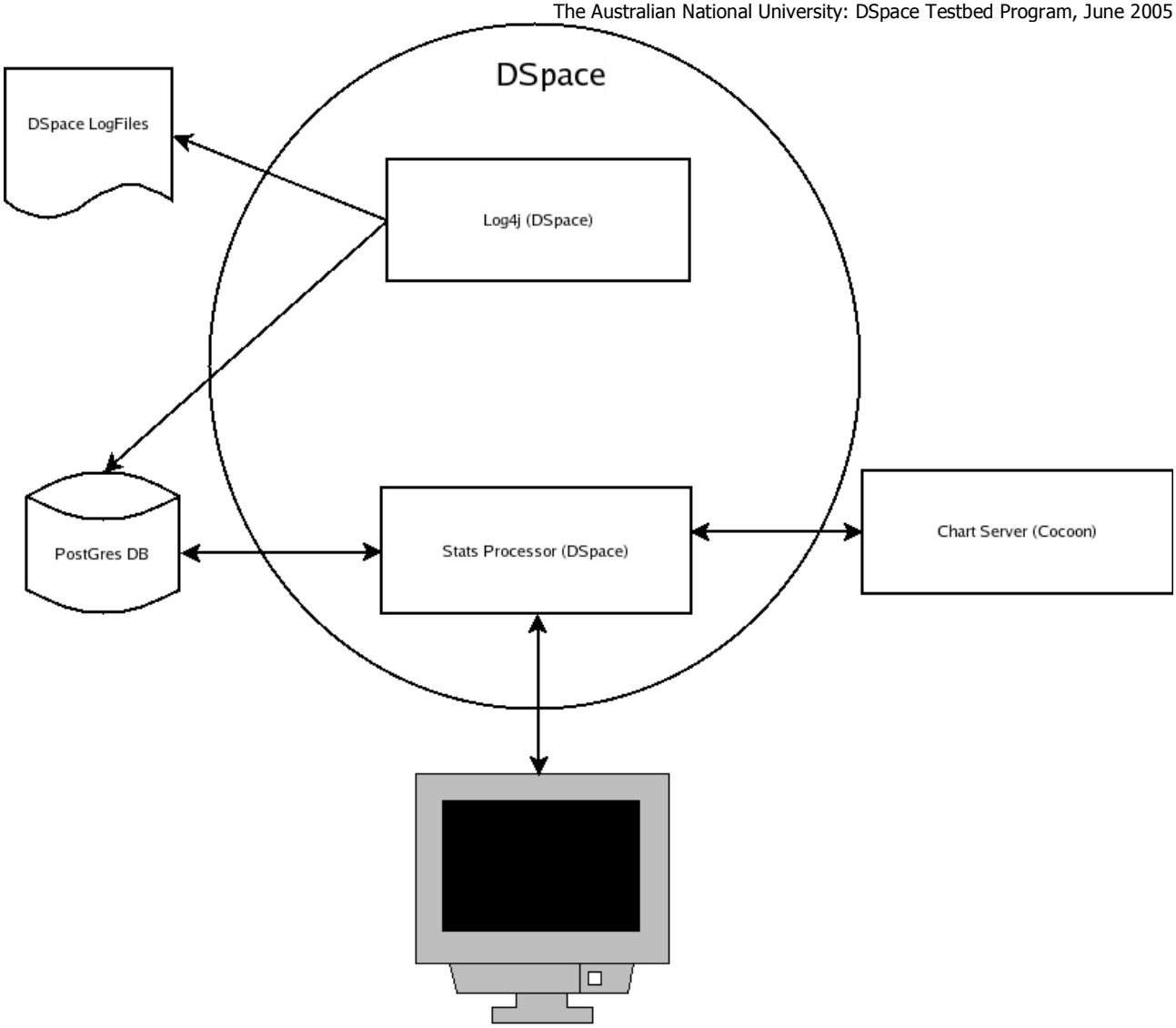
Servlets/JSP : used

to generate forms based on the queries in Querylist.xml,

to execute the query and optional transformation of the resultset

to display result of the query.

Cocoon (optional component): chart generator for Graphical presentation of the resultset .



transaction and data flow by example:

User accesses a bitstream, this event is logged by DSpace

Log4j appends the event message to the database

produces the following entry in the temp_log table:

Date = 2005-06-11:23:54,461

logger = org.dspace.app.webui.servlet.RetrieveServlet

priority = INFO

message = anonymous:session_id=600F15A4E8056EBD23C575F056A0474C:view_bitstream:bitstream_id=6110:ip_addr=150.203.59.132

The trigger function analyses the data and records the following data in the

View_bitstream_log table:

bitstream_id = 6110

item_id = 6109

session_id = 600F15A4E8056EBD23C575F056A0474C

user_id = anonymous

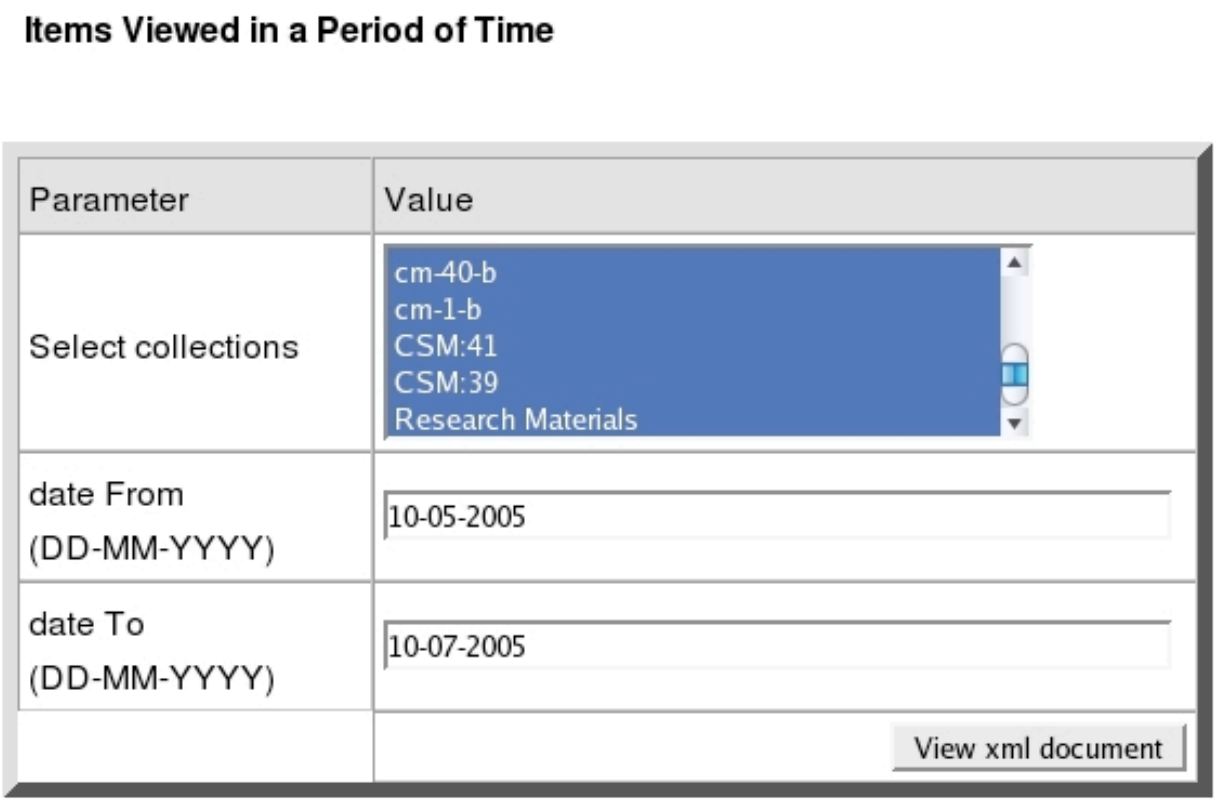
date = 2005-06-22

time = 11:23:54.4610

remote_ip = 150.203.59.132

This data can be retrieved by queries specified in the querylist.xml file:

For example, using the form generator servlet (in the following example we use the 'items-viewed-in-collection-inTime'), the data can be retrieved in XML, HTML or graph (Cocoon only, JPEG or SVG)



The Query used to generate stats:

<query name="items-viewed-in-collection-inTime" title="Items Viewed in a Period of Time">

Optional transformation specification.

Each should define type and stylesheet.

The result processor will transform the result set using the nominated by the users.

```
<option name="use-xsl-transform" type="xml" stylesheet="identity.xsl" render-to="xml document"/>
<option name="use-xsl-transform" type="html" stylesheet="resultset2table.xsl" render-to="table"/>
<option name="use-xsl-transform" type="graph" stylesheet="resultset2dataSet.xsl" render-to="linechart.jpeg"/>
```

Required Parameters

the from generator will produce input fields for each <param> elements to collect parameter values

```
<param src="collectionlist4Eperson" name="collection IDs" id="p3"/>
<param name="date From (DD-MM-YYYY)" id="p1"/>
<param name="date To (DD-MM-YYYY)" id="p2"/>
```

the SQL query definition element

the report generator will replace the parameters with the input values and execute the query.

```
<sql>SELECT collection_id, date, name, sum(volume) FROM (
      SELECT c2i.collection_id, vil.date, cl.name, count(vil.item_id) AS volume
        FROM view_item_log vil, collection2item c2i, collection cl
        WHERE vil.item_id = c2i.item_id
        AND c2i.collection_id IN (p3)
           AND cl.collection_id = c2i.collection_id
           AND date &lt; to_date('p2', 'DD-MM-YYYY')
        AND date &gt; to_date('p1', 'DD-MM-YYYY')
           AND vil.remote_ip not IN (select remote_ip FROM ip_filter)
        GROUP BY 2,3,1
    UNION ALL
        SELECT c2i.collection_id, vbl.date, cl.name,count(vbl.item_id)
        FROM view_bitstream_log vbl, collection2item c2i, collection cl
        WHERE vbl.item_id = c2i.item_id
        AND c2i.collection_id IN (p3)
           AND cl.collection_id = c2i.collection_id
           AND date &lt; to_date('p2', 'DD-MM-YYYY')
        AND date &gt; to_date('p1', 'DD-MM-YYYY')
           AND vbl.remote_ip not IN (select remote_ip FROM ip_filter)
        GROUP BY 2,3,1
     ) AS Foo
   GROUP BY 2,3,1
   ORDER BY 2</sql>
</query>
```

The output generated by the transformers for the appropriate collection and date range will produce the following resultset:

XML:

<resultset EPersonID="1" EpersonName="Leo Monus" QueryTitle="Items Viewed in a Period of Time" date="Wed Jun 22 16:11:54 EST 2005" p1="10-05-2005" p2="10-07-2005" p3="4,5,9,10,11,12,13,14,16" rName="items-viewed-in-collection-inTime" render-to="xml document" scaleBy="day" xsltype="xml"><result><collection_id>10</collection_id><date day="131" month="05" week="20" year="2005">2005-05-11</date><name>CM:1</name><sum>1</sum></result><result><collection_id>9</collection_id><date day="131" month="05" week="20" year="2005">2005-05-11</date><name>CM:40</name><sum>14</sum></result><result><collection_id>11</collection_id><date day="131" month="05" week="20" year="2005">2005-05-11</date><name>cm-40-b</name><sum>21</sum></result><result><collection_id>5</collection_id><date day="131" month="05" week="20" year="2005">2005-05-11</date><name>test..1..2..3</name><sum>7</sum></result><result><collection_id>12</collection_id><date day="132" month="05" week="20" year="2005">2005-05-12</date><name>cm-1-b</name><sum>13</sum></result><result><collection_id>11</collection_id><date day="132" month="05" week="20" year="2005">2005-05-12</date><name>cm-40-b</name><sum>18</sum></result><result><collection_id>4</collection_id><date day="132" month="05" week="20" year="2005">2005-05-12</date><name>potr loadtest</name><sum>28</sum></result><result><collection_id>5</collection_id><date day="132" month="05" week="20" year="2005">2005-05-12</date><name>test..1

HTML:

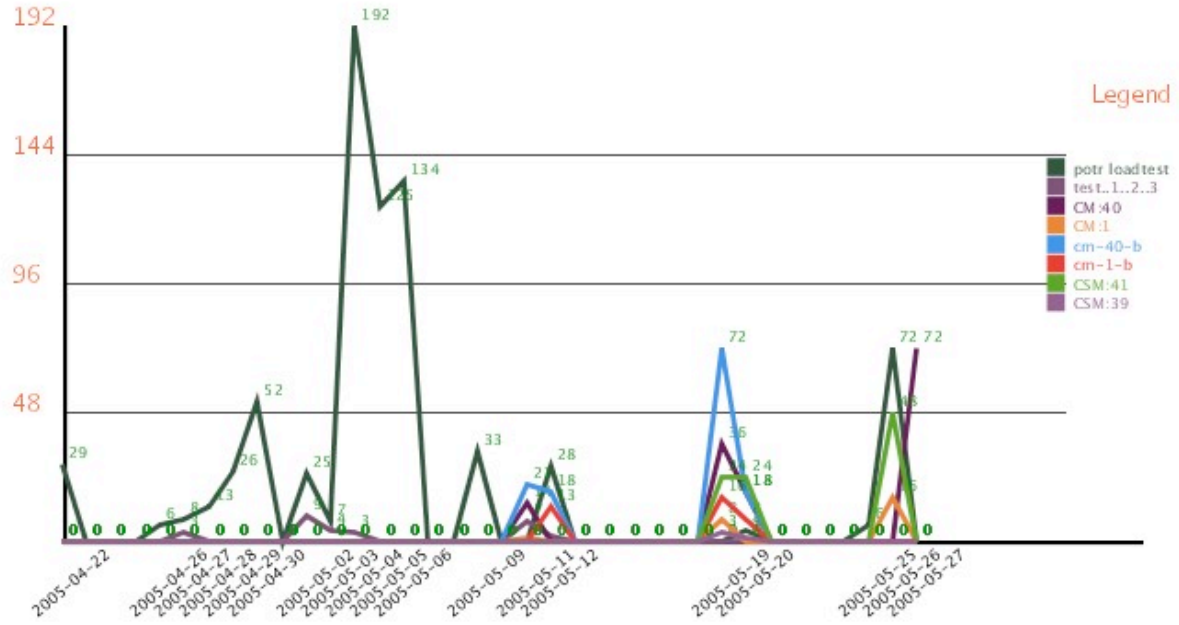
Items Viewed in a Period of Time

EPerson : Leo Monus

Date : Wed Jun 22 16:19:36 EST 2005

collection_id	date	name	sum
10	2005-05-11	CM:1	1
9	2005-05-11	CM:40	14
11	2005-05-11	cm-40-b	21
5	2005-05-11	test..1..2..3	7
12	2005-05-12	cm-1-b	13
11	2005-05-12	cm-40-b	18
4	2005-05-12	potr loadtest	28
5	2005-05-12	test..1..2..3	2
10	2005-05-19	CM:1	8
12	2005-05-19	cm-1-b	16
9	2005-05-19	CM:40	36
11	2005-05-19	cm-40-b	72
14	2005-05-19	CSM:39	3
13	2005-05-19	CSM:41	24

GRAPH:



Leo Monus
June 2005

Image thumbnail (derivative) generator

This work was undertaken as part of the Noel Butlin Archive early adopter deposition project. The thumbnail (derivative) generator was required initially to produce thumbnails as required by PictureAustralia, but was expanded to generate other image sizes, branded with ANU copyright information and the Item handle. This tool was released to the DSpace community following repeated requests.

Introduction

In late 2003 Peter Raftos approached the Noel Butlin Archive as part of a general effort to try and informally gauge interest and seek early adopters to contribute materials to an Institutional Repository. The materials needed to be of a simple form and ideally have associated metadata.

The Noel Butlin Archive had digitised assets in the form of TIFF images stored on a set of 23 CDs. In addition they had been maintaining metadata for these images in an Access database. These images and metadata seemed ideal for importing into DSpace as:

- the metadata was both substantial and structured;
- the assets were already in digital form;
- the assets were of a simple form and good quality, i.e. high-res TIFF images;
- the collection size was small, but large enough to provide for a useful exercise in determining how best to deal with images in DSpace;
- combining the metadata and image files and organising the images within the DSpace community and collection structures would provide for a good demonstration how improved management of the materials could be attained through loading them to a single repository.

Technical requirements

The images themselves were not complex objects. All were of TIFF format of varying file sizes ranging from as little as 4MB up to 100MB. A few images' profile information was examined to determine their resolution which turned out to be 600dpi.

The metadata for the images was maintained in an Access database. An exercise of mapping the Archive' metadata to the DSpace Dublin Core metadata was then undertaken. While it is possible to create new metadata items in DSpace, we were able to effectively map all metadata to core Qualified Dublin Core elements.

Over 12Gb of disk was needed to hold all the images.

To extract the metadata from Access we could have used Java with JDBC/ODBC or some other programming language using ODBC. However this approach is less than optimal for two main reasons:

- it reduces reusability of the code as it ties the code to database only imports and may require queries specific to the client database to be written as part of the metadata extraction process;
- it increases the complexity of what is basically once-off code for legacy collection imports.

A copy of the Access database was provided by the Archive and this was exported to a tab-delimited text file. A Perl script was written to extract and create XML metadata files for each image using a simple mapping table mapping from Access fields to Dublin Core. This process should be repeatable for other similar collections. These XML files are required by the DSpace item importer. A snippet of one such file is shown below.

```
<dublin_core>
...
...
<dcvalue element="title" qualifier="none">Saloon bar in the Bondi Junction Hotel</dcvalue>
<dcvalue element="relation" qualifier="ispartofseries">Photograph album comprising 81 folios of interior photographs of city and country hotels in New South Wales, taken during the 1930s and 1940s.</dcvalue>
...
...
</dublin_core>
```

There are only two elements used - the root **dublin_core** element and a **dcvalue** element for each item of metadata. The **dcvalue** element has two attributes: element whose value is the unqualified Dublin Core element name and **qualifier** whose value is a Dublin Core qualifier for that element, or "none" if using unqualified Dublin Core. Each image has one of these files generated prior to import to DSpace.

The images were divided up into collection directories manually (according to the collection they belonged to) and their associated XML files placed there with them. A Java program was then run to prepare the directories and files for import into DSpace. The DSpace ItemImporter (a Java program included with the DSpace distribution) was then used to import the image files and metadata into DSpace. The DSpace ItemImporter is run once for each collection.

This process resulted in all items being loaded to DSpace. Some modifications were made to enhance DSpace treatment of loaded images, but this is discussed in the next section.

The main issues encountered with this particular image collection were:

- metadata identifiers and images not matching;
- identifiers in the database without associated images;
- images without metadata;

- multiple items of metadata of different types in a single database field;
- inconsistencies in metadata.

The above issues did not cause significant problems for import to DSpace however may be areas to be explored for recommendations to areas compiling metadata for their collections.

Work required on DSpace

The following modifications were made while establishing the Noel Butlin collections:

- Local copies of JSPs were created to reflect ANU DSpace.
- An ANU JSP Tag Library was created in order to show derived image thumbnail on the item list and larger derived image on the detailed item view JSPs. The JSPs using these new tags were the local copies, not core DSpace JSPs, although the tag library borrows heavily from the DSpace tag library. The DSpace **web.xml** configuration file needed to be updated to include the ANU DSpace tag library and tag classes.
- A series of image classes were written to profile master images and create their derived images. For each image a viewing-sized JPEG and thumbnail are generated using the commercial product Image Alchemy (v1.13). Some utility classes were also written to support the primary image processing classes. All classes were bundled into a jar file **anu-dspace.jar**. The DSpace configuration file **dspace.cfg** needed to be updated with various new configurable items to support the image generation.
- The free ImageMagick (v5.5.7) image processing software was used to append the identifier and ANU branding to the large item view JPEG. ImageMagick was regarded as being too slow for generating derivatives, especially if on-the-fly generation is required in the future.

All the above work only required changes to DSpace configuration files to implement and no core DSpace code needed to be modified.

The sample collections can be found on the development DSpace at http://dspace-dev.anu.edu.au:8080/

The production DSpace can be found at http://dspace.anu.edu.au:8080/

Appended documents

- Javadocs for the ANU DSpace java classes can be found at http://dspace-dev.anu.edu.au:8080/anu-javadoc

Conclusion

- Need to determine a set of recommendations which we can pass to clients thinking of creating digitised images (e.g. Image specifications, opportunities for outsourcing, etc)
- Metadata quality and consistency is likely to be an issue in most collections, so will need to work closely with groups creating or maintaining metadata. Mapping to DSpace/DC is not an issue, but consistency and granularity are important for automation and searching/categorising respectively
- The image processing at the moment is largely limited to TIFF masters (which may not be an issue) and also to single image items. There may be a need to specify via metadata which images in a single item should be used for creating the derived image from. For example, in the Intercolonial Mutual collection there existed for around 20 images the master TIFF and a modified TIFF. Both images require to be archived but the derived image had to be created from the modified version. By default, both TIFFs had derived images created and then had to be manually reorganised such that the derived (or "access") image was the only one ever shown.

PictureAustralia collaboration

To ascertain that Collections could be harvested by PictureAustralia, where this was deemed appropriate by the Collection owners.

Version: 0.2
Last Updated: 24 August 2004

Introduction

The Noel Butlin Archive digitised images were well suited for exploring DSpace's capability to expose its metadata to a harvester. The images were ready for loading to DSpace and each image had associated metadata.

The exercise was to establish contact with the National Library of Australia (NLA) to determine the work involved in becoming part of their PictureAustralia service, and have a small set of test images harvested.

Technical requirements

Tony Boston from NLA was provided with output generated by the metadata provider bundled with DSpace. The feedback from this was used to build a metadata provider whose output conformed to the PictureAustralia schema.

Details of the PictureAustralia schema and about joining the PA service is available at http://www.pictureaustralia.org/join.html

Work required on DSpace

DSpace implements OAI-PMH by utilising and extending OCLC java classes so implementing a data provider will generally require only the DSpace-to-target-format java class(es) to be written.

The following modifications were made to create a PictureAustralia metadata provider:

- a PictureAustralia Crosswalk was written. This is a single java class which maps the DSpace Dublin Core metadata fields to the PictureAustralia schema (which is essentially Dublin Core plus additional elements);
- the PictureAustralia Crosswalk was registered in the DSpace **oaicat.properties** file;
- an XML configuration file is required to be set up for every collection wishing to expose metadata to PictureAustralia. This allows metadata mapping from DSpace to PictureAustralia to be configurable at a collection level. The root directory for these configuration files is specified in the **dspace.cfg** file. This model is also able to be extended for use by additional DSpace metadata provider implementations.

All the above work only required changes to DSpace configuration files to implement and no core DSpace code needed to be modified. Further metadata providers can be implemented the same way.

The staging PictureAustralia data provider can be found at http://dspace-dev.anu.edu.au:8080/dspace-oai and associated queries can be run by specifying prefix=pa (see http://www.openarchives.org/OAI/openarchivesprotocol.html for list of queries supported by OAI-PMH)

The production PictureAustralia data provider can be found at http://dspace.anu.edu.au:8080/dspace-oai

Appended documents

Extract from a PA configuration file for collection with database ID of 2.

```
<?xml version="1.0" ?>
<md.export-config service="pa" collection="2">
    <export-map>
      ...
      ...
        <mapping>
```

```
            <target-element>dc:title</target-element>
            <source-element>title</source-element>
        </mapping>
        ...
        ...
    </export-map>
    <export-map.extensions>
        <mapping>
            <target-element>dc:rights</target-element>
            <content>Please contact the Noel Butlin Archive for copyright information on this
material</content>
        </mapping>
    </export-map.extensions>
</md.export-config>
```

This XML file specifies the mapping from DSpace metadata fields to PA metadata fields.

The root element is **md.export-config** which contains a single mandatory **export-map** element and a single optional **export-map.extensions** element.

The **service** and **collection** attributes are not currently used but are likely to be required (or may be changed) in the future.

The **export-config** element contains many **mapping** elements, each containing a single **target-element** and one or more **source-element** elements. For each **mapping** element, the data provider simply maps the DSpace metadata field specified in each **source-element** to the element specified in the **target-element**.

The **export-map.extensions** element is unlikely to be used but if required is used to map a fixed constant string into a metadata field for exposure. In the example above a fixed digital rights statement is mapped to the PictureAustralia **dc:rights** element. Only one **content** element is permitted per **mapping** element.

Conclusion

- The most time-consuming part of creating data providers is ensuring the metadata mapping is correct and sensible. It is probably sensible to show the minimum useful set of metadata on PictureAustralia. More detailed metadata can be obtained by the PA user by navigating to the DSpace site via the PictureAustralia images.
- The PictureAustralia harvesting model is very simple and was very easy to get going using DSpace. DSpace has provided an OAI-PMH compliant data provider implementation which reduced the work required.

MODIFICATIONS FOR DSPACE 1.2

Overview

Under DSpace 1.2 the identification of sets changed from using the community:collection hierarchy to a modified collection handle identifier. For example, previously the Noel Butlin Archive Collection could be harvested using the setSpec 1 (community #1) which would harvest the entire set of collections within the specified community.

The 1.2 mechanism, implemented presumably partly because of the complexities caused through sub-communities, meant that this one setSpec then became around 9 separate provider URLs (one for each collection). Rather than create a maintenance issue for PictureAustralia, e.g. if internal set organisation

changed again, or more collections are exposed, an alternative approach was adopted.

Note that this maintenance issue already existed before DSpace 1.2, but at the community level. The new set implementation highlighted the issue because the setSpec was brought down to collection level.

Modifications - Description

The approach taken was to create a mechanism by which exceptions to the DSpace OAI implementation could be picked up, altered for compatibility with the DSpace implementation and then fed back into the DSpace OAI processing.

The initial extension was to allow specification of setSpecs identified by an arbitrary string rather than a collection handle. DSpace handles any setSpec beginning "hdl_" as a collection. The extension allows a set to be given any name (other then starting with "hdl_"!) and then arbitrary collections and communities can be assigned to the set. These are then turned into a list of collections which are provided for subsequent (existing) processing by the data provider.

Modifications - Specific

A new XML configuration file was added to allow the specification of high-level sets. An example file (containing the PictureAustralia configuration) is as follows:

```
<sets>
        <set name="pa">
                <items source="community" id="1885/1"/>
                <items source="community" id="1885/14"/>
        </set>
</sets>
```

The **name** attribute is what needs to be used in the setSpec (the ListSets will include all set names so automatic harvesting of all sets is not affected).

The **items** element is used to specify what communities and/or collections comprise the set. Its **source** attribute indicates "collection" or "community" and the **id** attribute specifies the handle. Note that the **source** attribute is not used - the HandleManager's object resolver method is used to determine a community or collection – it acts as a visual aid only.

Item level support is not included as this would require more significant changes to the core DSpace code.

Three DSpace core java files needed to be slightly modified:

- DSpaceOAICatalog.java
- DSpaceRecordFactory.java

Harvest.java

These classes needed to be modified to accept/return an array of Collections rather than a single collection (slight modification)

A new class OAIUtil was also created which:

- takes a non-handle set name and resolves it to a list of collections

returns all set names which have been configured in the XML file

An additional config item "export.oai.extensionfile" was added to dspace.cfg which points to the XML config file discussed previously.

The oaicat.properties (and corresponding template file) was modified to point the ANU versions of the above catalog and recordFactory classes.

Collaboration with MusicAustralia

It was anticipated that, following the ease of interoperation with PictureAustralia, getting the same results with MusicAustralia would not be difficult. The selected Community for trialing was the Australian Anthology of Music (AAM), administered in part by the School of Music, Faculty of Arts.

The high-quality WAV files were extracted, along with technical metadata, from the AAM's 42 music CDs earlier this year. We are grateful to the National Library of Australia for its assistance in providing technical expertise and the use of its Quadriga system. These files have bee uploaded to the test bed's DSpace development server, and a trial web presence (via the Cocoon XML publishing framework) has been developed.

We are now awaiting copyright clearance to make the compositions available on the web (so that they can be accessed through MusicAustralia). This work is being done by the School of Music and has proven more involved than first thought. This is the only delay in going live and has proven a valuable learning experience.

Peter Raftos
June 2005