ECHO DEPository Technical Architecture Phase 1 Final Report

Report of project activities from Fall 2004 through 2007

University of Illinois at Urbana Champaign in partnership with OCLC

Contributors: Matt Cordial, David Dubin, Janet Eke, Joseph Futrelle, Thomas Habing, Leah Houser, Patricia Hswe, William Ingram, Joanne Kaczmarek, Robert Manaster, Joel Plutchak, Beth Sandore, John Unsworth

December 2008

Table of Contents

1. Preface ... 4
1.1. About the ECHO DEPository Project (Phase 1) ... 4
1.2. About This Document ... 4
1.3. Review of Project Objectives and Deliverables ... 4
1.3.1. Models and Tools to Support Web Archiving ... 4
1.3.2. Repository Evaluation and Interoperability ... 5
1.3.3. Long-term Semantic Preservation Research ... 6

2. Archiving the Web : the Web Archives Workbench ... 8
2.1. Overview ... 8
2.2. The Web Archiving Problem ... 8
2.2.1. The Ubiquitous Web ... 8
2.2.2. Volume and Selection of Web Content ... 9
2.2.3. The Importance of Context ... 9
2.3. The Arizona Model: An Archival Approach to Web Archiving ... 10
2.3.1. Background ... 10
2.3.2. An Archival Approach ... 11
2.3.3. Arizona Model Summary ... 11
2.4. The Web Archives Workbench: Implementing the Arizona Model ... 12
2.4.1. Development Considerations ... 12
2.4.2. Overview of the Web Archives Workbench ... 14
2.4.3. A Tour of the Web Archives Workbench ... 15
2.4.4. Web Archives Workbench Tools Summary ... 20
2.4.5. Behind the Scenes: OCLC's Technical Implementation of the Web Archives Workbench ... 21
2.5. Findings - User Feedback ... 24
2.5.1. Limited Resources and Limited Time ... 25
2.5.2. Complexity of the Tools ... 25
2.5.3. Web Content Delivery ... 25
2.6. Conclusions and Next Steps ... 26

3. Repository Evaluation and Interoperability ... 27
3.1. Repository Evaluation ... 27
3.1.1. Building an Evaluation Framework: Applying the Trusted Digital Repository Checklist to Repository Evaluation ... 27
3.1.2. Repository Testing: Ingest and Export Tests On Four Key Open-source Repositories ... 28
3.1.3. Testing Approach and Methodology ... 32
3.1.4. Repository Testing Findings: Narrative Reports, and Annotated Audit Checklist Commentary ... 34
3.1.5. Conclusion and Next Steps ... 35
3.2. Hub and Spoke Architecture (HandS): Supporting Repository Interoperability and Emerging Preservation Standards ... 36
3.2.1. HandS Overview ... 36
3.2.2. The Need for Interoperability and Preservation Support ... 36
3.2.3. Hub and Spoke Key Principles ... 37

3.2.4. METS Profile ... 39
3.2.5. HandS Workflow Cycle ... 44
3.2.6. HandS Technical Implementation ... 50
3.2.7. Lessons Learned ... 55
3.2.8. Next Steps: the Hub and Spoke ... 56
3.2.9. Conclusion ... 57

4. Preserving Meaning, Not Just Objects: Semantics and Digital Preservation 58
4.1. Introduction: The Need for a Semantics of Preservation Approach ... 58
4.1.1. The Preservation Semantics Problem ... 58
4.1.2. Our Goal ... 59
4.2. The Problems: Understanding Semantic Preservation ... 59
4.2.1. Problems Posed by Descriptive Practice and Structures ... 59
4.2.2. Understanding the Semantic Preservation Problem: Summary ... 64
4.3. Toward More Capable Archives and Repositories ... 64
4.3.1. Recap: The need for automated inference capability ... 64
4.3.2. BECHAMEL and Building a Metadata Ontology ... 65
4.3.3. Overcoming Semantic Problems in Metadata Encoding: A Resource and Description Vocabulary ... 65
4.3.4. Resolving Semantic Ambiguity: an Inference Example ... 66
4.3.5. Automated Inference as a Preservation Service ... 68
4.4. System Architecture ... 69
4.4.1. Architecture: Overview ... 69
4.5. Lessons Learned and Next Steps ... 71
4.6. Conclusion ... 71

5. A Note from the PIs ... 73

6. References ... 74
6.1. Archiving the Web: the Web Archives Workbench ... 74
6.1.1. Resources ... 74
6.2. Repository Evaluation and Interoperability ... 75
6.2.1. Repository Evaluation ... 75
6.2.2. HandS Tools Suite ... 75
6.3. Preserving Meaning, Not Just Objects: Semantics and Digital Preservation .. 77

7. Appendices ... 80
7.1. Web Archives User Guide ... 80
7.2. Web Archives Workbench Implementation Guide ... 80
7.3. Annotated Trusted Digital Repository Checklist ... 80
7.4. Using the Audit Checklist for the Certification of a Trusted Digital Repository as a Framework for Evaluating Repository Software Applications (DLib article) ... 80
7.5. Repository Testing Findings: Narrative ... 80
7.6. Repository Findings Commentary Using the Annotated Trusted Digital Repository Checklist ... 80
7.7. Resource Description Vocabulary: An Ontology of Metadata Descriptions ... 80
7.8. Sustained Access to Ejournals: Context Value, and Future Prospectus ... 80

1. Preface

1.1. About the ECHO DEPository Project (Phase 1)

The ECHO DEPository (Phase 1) is an NDIIPP-partner research and development project at the University of Illinois at Urbana-Champaign (UIUC) in partnership with OCLC, the National Center for Supercomputing Applications); the Michigan State University Library; and an alliance of state libraries from Arizona, Connecticut, Illinois, North Carolina and Wisconsin. Our aim is to support the digital preservation efforts of the Library of Congress by addressing issues of how we collect, manage, preserve, and make useful the enormous amount of digital information our culture is now producing.

Phase 1 project activities (Fall 2004 through 2007) included developing web archiving tools, evaluating existing repository software, developing an architecture to enhance existing repositories' interoperability and preservation features, and modeling next-generation repositories for supporting long-term preservation.

1.2. About This Document

This narrative report provides a detailed overview of each of the areas of work described below. The attached appendices provide specific additional project deliverables. A collected archive of all project deliverables, including posters, presentations and publications, is forthcoming. Several sections include material contributed by the same authors who wrote articles on ECHO DEP projects for an issue of *Library Trends,* guest-edited by Patricia Cruse and Beth Sandore, to appear in the Winter 2009 issue (specifically, *Library Trends,* Volume 57, Number 3).

1.3. Review of Project Objectives and Deliverables

1.3.1. Models and Tools to Support Web Archiving

Goals:

- Articulate a methodology for selecting digital materials at the aggregate level based on archival principles, and use provenance, functional analysis, and context analysis to facilitate meta-tagging for retrieval.
- Build a suite of open source software tools that support identification, capture, and description of web sites.

Deliverables:

- The Arizona Model, an archival approach to web archiving (developed by the Arizona State Library and Archives)
- The Web Archives Workbench suite of tools (developed by OCLC)

Overview

Traditionally, web archiving methods have focused on either manual or automated capture approaches, both problematic. Manual item-level selection fails due to the enormous number of resources on the web, while fully automated web-capture approaches risk burying substantive materials under a mountain of irrelevant information. To address this fundamental problem, OCLC built a suite of open-source web archiving tools that bridge the gap between manual selection and automated capture. Based on the Arizona Model, which provides for integration of both human and machine processes, the Web Archives Workbench (WAW) comprises four tools to help archivists identify, describe, select and harvest web-based content for storage in any repository.

Details

See Section 2 of this report, and Appendix items 6.1 and 6.2.

1.3.2. Repository Evaluation and Interoperability

Goals:

- Install, configure, test, and evaluate existing open-source digital repository systems, with particular regard to support for interoperability and emerging preservation standards.

Deliverables:

- A repository evaluation framework based on the 2005 RLG/NARA *Audit Checklist for the Certification of a Trusted Digital Repository, Draft for Public Comment*, with mapping to current version
- Repository testing findings
- The Hub and Spoke (HandS) tools suite supporting repository interoperability and preservation metadata
- PREMIS-based METS profiles

Overview

To help understand how well existing repository systems support today's digital preservation efforts, we evaluated four existing open-source repository systems (DSpace, ePrints, Fedora and Greenstone). Evaluation activities included the ingestion and manipulation of half a terabyte of

heterogeneous content in each repository system, and the development of a preservation-focused repository evaluation framework based on emerging standards for trusted digital repositories.

Early evaluation findings led to the development of the Hub and Spoke (HandS) Architecture, a proof-of-concept suite of tools to enhance the interoperability and preservation features of systems tested. The HandS suite supports today's libraries' efforts to manage content in multiple repository systems and to preserve valuable preservation data. It includes the development of a common standards-based method (a PREMIS-based METS profile) for packaging content that allows digital objects to be moved in and out of more repositories more easily while supporting the collecting of technical and provenance information crucial for long-term preservation. This model has potential wide applicability, and is already in use in several real-world archiving projects.

Details

See Section 3 of this report, and Appendix items 6.3, 6.4 6.5 and 6.6.

1.3.3. Long-term Semantic Preservation Research

Goals:

- Research techniques to migrate the semantic content of documents and document structures across generations of encoding schemes.

Deliverables:

- Articulation of semantic preservation problems posed by current metadata practice
- Development of a formal metadata description ontology
- Demonstration of automated inference using reasoning software

Overview

Current first-generation repository systems preserve the structure of information, not its meaning or semantics. When we move content from one system to another, this structure may be subtly or unsubtly transformed. To meaningfully preserve our digital content over time, we therefore have to infer meaning or semantics from structures that change over time. Because of the volume of information to be preserved, we need to be able to do this with automated tools.

The UIUC Graduate School of Library and Information Science (GSLIS) collaborated with the National Center for Supercomputing Applications

(NCSA) to contribute to the development of next-generation archives with semantic analysis capabilities to reduce long-term preservation risks. Using repository technology developed at NSCA and automated reasoning tools developed at GSLIS, we model how semantic inference capability may help next-generation archives head off long-term preservation risks. This work includes articulating semantic preservation problems posed by current practice, and analyzing real-world data migration examples to develop a formal understanding of how descriptive information about archived digital resources is structured. This understanding is presented in a formal metadata description ontology.

Details

See Section 4 of this report, and Appendix item 6.7.

2. Archiving the Web : the Web Archives Workbench

2.1. Overview

A core deliverable of the ECHO DEPository Project's first phase was OCLC's development of the Web Archives Workbench (WAW), an open-source suite of Web archiving tools for identifying, describing, and harvesting Web-based content for ingest into an external digital repository. Released in October 2007, the suite is designed to bridge the gap between manual selection and automated capture based on the "Arizona Model," which applies a traditional aggregate-based archival approach to Web archiving. (By "aggregate-based archiving," we mean archiving items by group or in series, rather than individually.) Core functionality of the suite includes the ability to identify Web content of potential interest through crawls of "seed" URLs and the domains they link to; tools for creating and managing metadata for association with harvested objects; website structural analysis and visualization to aid human content selection decisions; and packaging using a PREMIS-based METS profile developed by the ECHO DEPository to support easier ingestion into multiple repositories.

The next sections provide an overview of the Web archiving problem; background on the Arizona Model; an overview of how the tools work and their technical implementation; and a brief summary of user feedback from testing and implementing the tools. Appendix items include the Web Archives Workbench User Guide (6.1), which provides detailed screen-by-screen documentation of the tool suite's functionality. The WAW Implementation Guide is provided in Appendix 6.2.

2.2. The Web Archiving Problem

2.2.1. The Ubiquitous Web

For a broad range of organizations, Websites are now the delivery mechanism of choice for nearly any type of information content. Much of this content is created and disseminated in electronic formats only, with printed (hard) copies considered just a courtesy or convenience. The electronic format environment, while expedient for current access purposes, presents challenges for anyone charged with preserving the information over time. These challenges include the sheer volume of Web-published information, traditional issues of selection and description, as well the technical challenges associated with long-term preservation of digital objects.

2.2.2. Volume and Selection of Web Content

An immediate challenge of Web archiving is assuring that all content of long-term relevance delivered through the Web is identified and collected (i.e., harvested). Difficulties arise first from the task of selecting pertinent content for preservation from the enormous volume of information streaming from Web servers at any given point in time. Selection decisions will be influenced by the charge of the individual responsible for capturing specific content types (such as a librarian or archivist) based on appraisal, or collection development; on policies created in concert with the mission of the institution or organization; and on the audience, or user community, being served. The sheer volume of content published on the Web makes a fully manual perusal of online resources infeasible. Volume is still a factor even when Web crawlers—as explained below—are engaged.

The dynamic nature of the Web also creates problems for selection and harvesting of content. URLs can change overnight; resources can be taken off-line with little or no notice; and new, related content can be added in new or different directories than those visited previously by a Web crawler harvesting an organization's website. Although Web crawling automates archiving of a website, it is quite possible for Web crawlers simply to miss content because of a "robots exclusion protocol" (activated by the site creator to make parts of a site "uncrawlable") or because of the impenetrable character of the Deep Web (where content, such as a results page to a Web form, is inaccessible to a Web crawler or Web spider).[1] In addition, the vast measure of the Web renders scalable Web crawling an almost intractable technical challenge. Knowing where to find all content eligible for harvesting according to collection development and appraisal policies becomes nearly impossible without intentional coordination or without Web crawling tools and resources that are designed for, and take account of, the fluid nature of website content and the massive scale of the Web.

2.2.3. The Importance of Context

Context is about understanding relationships between different and discrete pieces of information. It is about understanding why the information was created, by which individual or organization, and at what point in time. Contextual information can help define the boundaries and the scope of harvested content.

As with analog objects, much of the usefulness of digital objects which make up our cultural record depends on our having descriptive and contextual information about them. Once content is identified and harvested, it is necessary to provide access to the digital object. Such content access means that attention should be paid to capturing accurate metadata along with the content itself. This contextual metadata

[1] Web archiving. (2009, March 2). In Wikipedia, The Free Encyclopedia. Retrieved 21:56, March 2, 2009, from http://en.wikipedia.org/w/index.php?title=Web_archiving&oldid=274363238

will help describe the origin (or "provenance") of the resource, as well as why and when it was created. (For example, is the discovered resource one in a series of annual reports from a particular state agency? Is it a single publication summarizing research findings? Or does it encompass results from a specific survey taken as part of a larger effort to revamp community services?) In the case of a digital object, metadata not only supports human interpretation of content, it is needed to provide crucial technical information for maintaining long-term viability of the object itself.

2.3. The Arizona Model: An Archival Approach to Web Archiving

The Web Archives Workbench tool suite is premised on the principles of the "Arizona Model," an aggregate-based approach to Web archiving designed to bridge the gap between human selection and automated capture. "Aggregate-based" means that rather than archive items singly, or individually, they are organized (grouped) in series, or in aggregates. The Arizona Model was developed in 2003 by Richard Pearce-Moses of the Arizona State Library and Archives.

2.3.1. Background

Most state libraries and archives have mandates to collect state agency publications and make them available to the public. To this end, there are well-established depository systems that have worked with paper publications for many years. In a Web environment the nuances of determining what a publication is, or who is responsible for selection and collection of particular information resources, becomes less clear. Nonetheless, to meet these mandates librarians and archivists must still identify, select, acquire, describe, and provide access to state agency information "published" on websites.

In early attempts to develop a collection of state agency electronic publications, two approaches came about. According to Pearce-Moses, Cobb, and Surface (2005), the first approach has its premise in "traditional library processes of selecting documents one by one, identifying appropriate documents for acquisition; electronically downloading the document to a server or printing it to paper; then cataloging, processing, and distributing it like any other paper publication." (175) While this approach ensures that valuable documents will be gathered, its dependence on manual selection limits archiving to only a very few items. Scaling this process in accordance with the vastness of Web-based documents would necessitate an expansion in personnel that few state libraries have the funding to address. (Pearce-Moses, Cobb & Surface, 2005) Alternatively, in the other approach, software tools that automate regularly occurring Web crawls are engaged. As Pearce-Moses, Cobb, and Surface assert, this model "trades human selection of significant documents for the hope that full-text indexing and search engines will be able to find documents of lasting value among the clutter of other, ephemeral Web content captured in the process." (176) Yet, while this model relieves librarians and

archivists of the upfront onus of selection and organization, at the same time it may unduly burden future searchers, if full-text indexing and search capabilities do not evolve as anticipated.

The Arizona Model, explained in detail below, constitutes a third approach to Web archiving, incorporating both human assessment and automated tools.

2.3.2. An Archival Approach

The Arizona Model applies an archival perspective to curating collections of Web publications. It exploits certain telling parallels between websites and archives: namely, the concept of provenance (i.e., documents classed together stem from the same source) and the organizational structure inherent in both these kinds of collections—directories and subdirectories for websites, and series and subseries for archives. (Pearce-Moses, Cobb & Surface, 2005) In theory, if websites organize Web publications using common file directory structures, information about individual documents within sub-directories could be inherited from parent directories.

In the Arizona Model, which draws on basic archival practice, websites are handled as hierarchical aggregates rather than as individual items, and the original order of the documents (the order in which the creating agency oversaw them) is maintained. Provenance and original order are considered important contextual pieces of information. Retaining documents in the order in which they were originally managed and keeping them clustered together based on the originating agency enhance one's knowledge of the creation and original use of the documents. Provenance and original order also allow for "inheritance" of higher-level metadata meant to describe the home agency from which the documents came and the way the documents were originally arranged.

Finally, an archival approach to curating a collection of Web documents — focusing first on aggregates (collections and series), rather than on individual documents — trims the number of items that need to be appraised by a human down to a more manageable number.

2.3.3. Arizona Model Summary

The Arizona methodology is based on an archival approach to the Web that incorporates both human selection and automated capture. In this approach, Web materials are managed in a way similar to the organization of materials in paper-based archives: as a hierarchy of aggregates rather than as individual items. This approach reduces to a more practical size the sheer volume problem of preserving Web materials, while maintaining a scalable degree of human involvement. It is the guiding model for OCLC's Web Archives Workbench.

2.4. The Web Archives Workbench: Implementing the Arizona Model

The Arizona Model is particularly instructive in its evocation of where, in the practice of archival management, automation can be considered most useful. That is, while technology may be applied for information processing activities such as data searching and tracking, and list construction and classification, tasks for distinguishing whether content is in-scope or is valuable are best reserved for humans. The overall goal behind developing a suite of tools based on the Arizona Model is to achieve a productive complement between automated processing and human decision-making, all the while adhering to established archival principles.

The software that OCLC created, the Web Archives Workbench, comprises five tools to identify, select, describe, and harvest Web-based materials, as well as to keep track of, or log, these activities and to generate reports about them. In doing so, they serve as a conduit between human involvement (via manual selection) and computerized capture of Web content: they convert the archivist's policies for collecting content created on the Web to software-centered rules and configurations. They also assist information professionals by providing the means to add metadata to harvested objects as aggregates. In addition, the tools implement the PREMIS-based METS profiles developed by ECHO DEP (at the University of Illinois) for packaging content; by design these profiles facilitate ingestion into multiple external repositories and support long-term preservation.[2] Packaging is the last step in the WAW workflow, after which the objects are ready for ingest into an external digital repository.

2.4.1. Development Considerations

OCLC led the development of the tool suite. Prior to tool design and development, OCLC carefully considered the user community, which it identified as a blend of librarians and archivists. Significant to its consideration was the issue of terminology: how should tools and features in the Web Archives Workbench be named, or called, if a mixed community of librarians and archivists was to serve as its user base? The word "series," for example, might invoke semantics and usage for an archivist that is different, even unfamiliar, for a librarian. Thus, in exploring the user community, OCLC had archivists look at new types of metadata and asked librarians to think about principles of archiving, such as archival series and the curation of

[2] Two METS profiles developed by ECHO DEP are at work here: the ECHO Dep Generic METS Profile for Preservation and Digital Repository Interoperability (accessible at http://www.loc.gov/standards/mets/profiles/00000015.html) and the ECHO Dep METS Profile for Web Site Captures (accessible at http://www.loc.gov/standards/mets/profiles/00000016.html). The former is the 'top level' format-generic profile, which focuses on implementing PREMIS. The latter, a web capture profile, is an example of a 'sub-profile,' which is used with the first one to provide a structure for more format-specific information.

documents in aggregate rather than as individual items. Eventually, OCLC elected not to devise new terminology for the concepts at issue; not only did the team conclude that terminology was, in essence, a training matter, it also saw that the work of librarians and archivists often overlap—i.e., each is frequently engaged in the milieu of the other.

In doing high-level analysis for the user interface, OCLC arrived at several working assumptions that had some bearing on the design of the tool suite. One assumption was that because the tools in the Web Archives Workbench might change over time, they needed to be "aware" of each other and enable the sharing of data, but—as important—the user should have the ability to opt not to use a tool in the Workbench. Through interviews with librarians and archivists, OCLC also learned that harvesting responsibilities often were shared among individuals; as a consequence, data generated by a tool had to be rendered shareable by multiple users—and simultaneously so. This feature would allow a user to view the work of another. In addition, rather than trying to integrate the Workbench into an institution's many authentication schemes, OCLC incorporated a simple scheme, allowing the Workbench to run with just basic administration. In terms of harvesting, OCLC designed more than one harvesting workflow, so that a user could select the appropriate level of analysis and sophistication for a task. For instance, the Quick Harvest feature is a single-screen launch point that runs a harvest immediately. The Analysis tool, which is part of an extended harvesting workflow, requires more set-up, but it results in a bigger "pay-off" in terms of the website change observations it handles automatically for the user (this is explained below more formally in the description of the Analysis tool).

Finally, where the deposit of harvested information is concerned, OCLC knew that ingest to a variety of repositories, including its own Digital Archive as well as DSpace repositories, would need to be accommodated. A clean, simple interface was created between the point where the Workbench ends and a repository software application would begin; that is, the Workbench generates harvested packages of content in a file system that the repository then picks up and processes. (This is the point in the workflow at which the above-mentioned PREMIS-based METS profiles developed by ECHO DEP is implemented.)

2.4.2. Overview of the Web Archives Workbench

The Web Archives Workbench is a suite of web archiving tools for identifying, selecting, describing and harvesting web-based content based on library and archival practice. It bridges the gap between manual selection and automated capture by transforming collection policies into software-based rules and configurations. It accommodates a variety of web harvesting approaches, including mass harvesting, selective harvesting, and individual document harvesting. Content is packaged using the ECHO DEP METS profile, which is designed to support the collection of PREMIS preservation metadata, and to facilitate ingestion into a variety of external repositories.

The five tools in the Workbench are the Discovery, Properties, Analysis, Harvest, and System tools. Below (Fig. 1) is an overview of the Workbench Workflow, followed by a more detailed tour of the functionality of each tool.

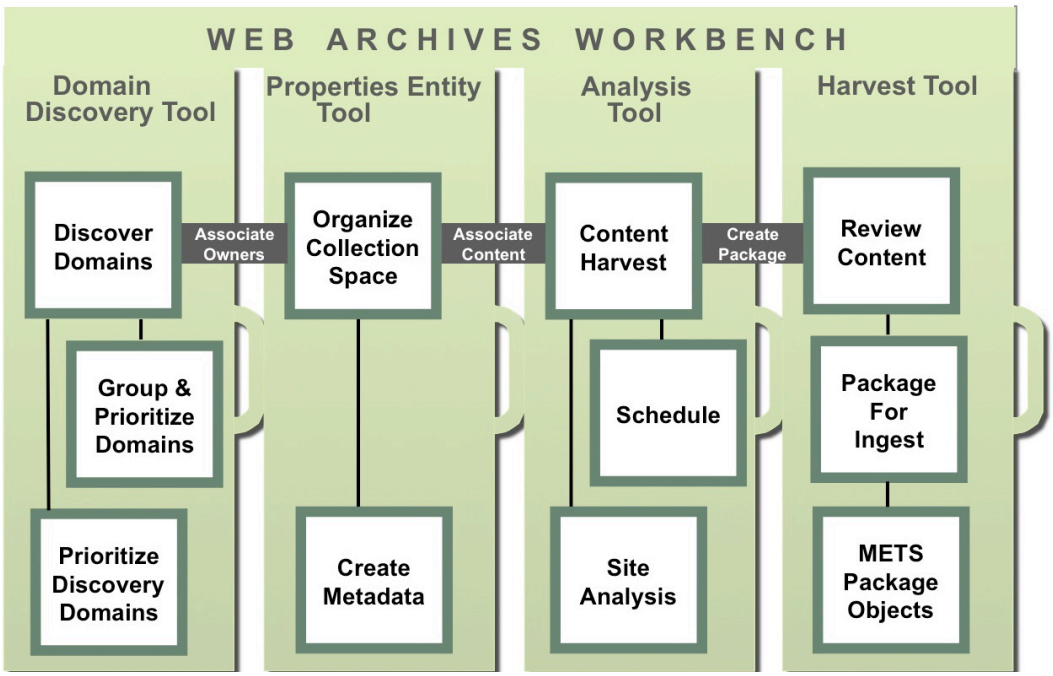


Figure 1: Overview of WAW Workflow

2.4.3. A Tour of the Web Archives Workbench

The screenshot in below displays the main WAW tools screen after the user has logged on. The five tools are exemplified by the topmost row of tabs. (Though the Alerts tab sits in this row, it is less a tool than a feature of the Workbench. It enables users to access a collection of reports and alerts for the Discovery, Properties, Analysis, and Harvest Tools.) In the interface for the WAW tools, a tab is colored in to signify which tool is open, or active, at that particular moment. In Figure 2, for example, the Discovery tab is shaded, because the Discovery tool is currently active. Similarly, the Entry Points tab is shaded, because it is active as a component of the Discovery tool.

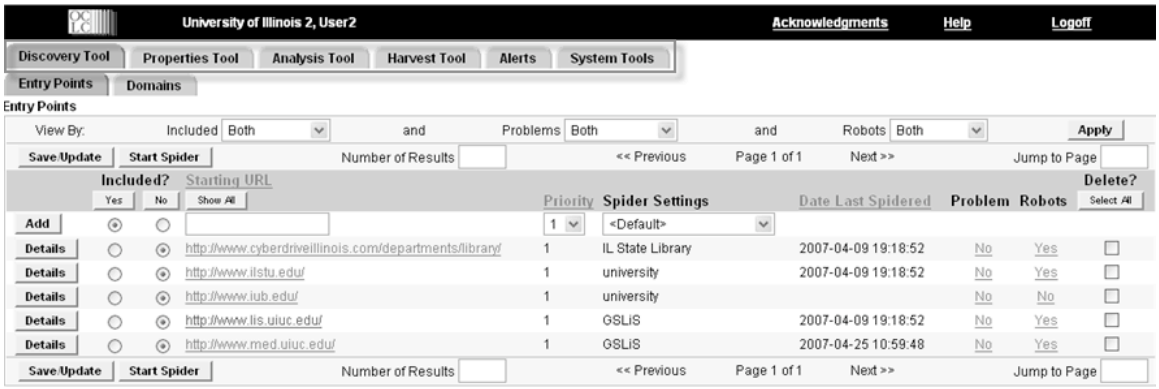


Figure 2: Screenshot of WAW interface home screen

A key advantage to the Workbench tools is that harvesting of Web content may be scheduled so that it occurs on a regular basis. However, the Workbench tools also offer users the alternative of running a one-time harvest. This is known as the Quick Harvest, accessible via the Harvest tab. Quick Harvest is addressed briefly in the discussion below of the Harvest tool.

2.4.3.1. The Discovery Tool: Finding Web Content of Interest

The first step in constructing an archive of Web-based resources is to determine which parts of the Web hold desirable, and thus collection-worthy, content. This step lies at the crux of the Discovery Tool. The Discovery Tool aids in identifying potentially relevant websites by crawling relevant "seed" entry points to generate a list of domains to which the "seed" sites link. (Note: An entry point is a specific website URL where the Discovery Tool will begin to search for domains or collect Web content. A domain is a server on the Internet that may contain Web content and is identified by a high-level address. For example,

http://www.illinois.gov/news/ is a website, and its domain is "Illinois.gov". Domains do NOT include “http://”.)[3]

In an approach that effectively borrows from citation analysis, the Discovery Tool is designed on the idea that on-topic sites likely point to other sites addressing a similar topic. The domains in the generated list are then manually evaluated as in-scope or out-of-scope, based on subject interest and collecting policies. (See Figure 3, which shows a list of domains returned after entry points have been crawled, as well as radio buttons that note the scope for each domain.) This process results in a list of domains defining a sub-set of the Web that is relevant for the user's archiving purposes. Domains marked as in-scope can be associated with an Entity (i.e., creator, or agency, or organization responsible for the Web content). Later, in the Properties and Analysis Tools, metadata associated with entities (creators such as agencies or organizations) can be inherited by content harvested from a particular website.

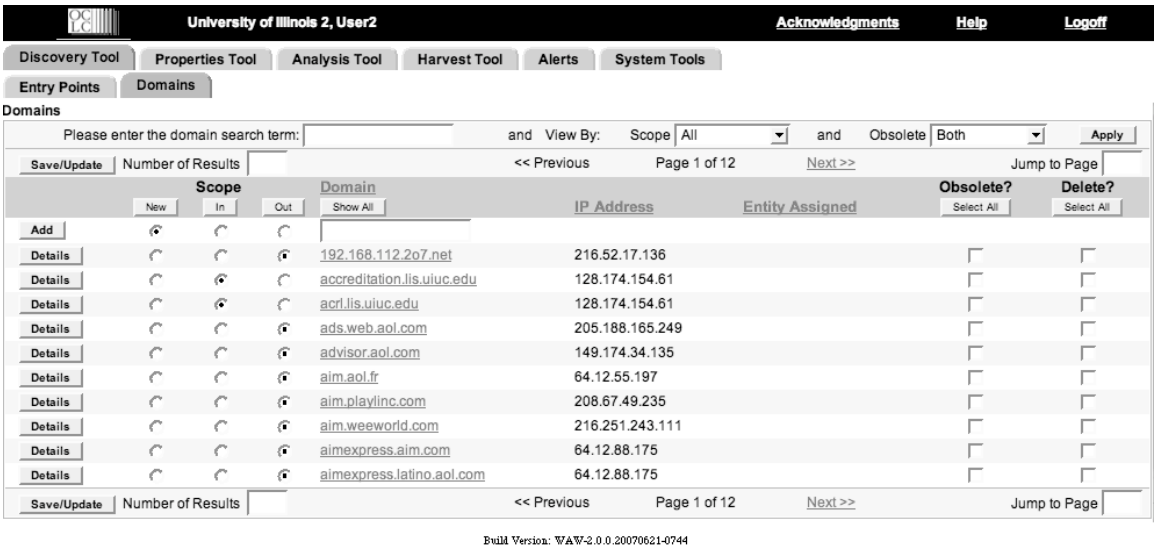


Figure 3: Screenshot of the interface for the Domains feature of the Discovery Tool

In sum, the Discovery Tool is used to:

- Generate a list of potentially relevant domains by crawling seed sites.
- Assign domains as in-scope or out-of-scope.
- Add domains manually to the Domains list.
- Associate domains with entities (creating agencies or organizations).

[3] A note about capitalization in this section that provides a tour of the software: here, entry points and domains are not capitalized, because we are speaking of them in the general use of the Discovery Tool. However, they are also features of the Discovery Tool. When we discuss them as such, they will be capitalized.

2.4.3.2. The Properties Tool: Entering Metadata to Describe Content Creators (Entities)

Another premise of the Arizona Model is that, as much as possible, metadata should be entered only once and be inherited by associated harvested objects. After the Entry Points and Domain features of the Discovery Tool are run, and entities (i.e., content creators) have been associated with domains, metadata about the resulting entities may be entered via the Properties Tool. Besides enabling the management of information about entities, the Properties Tool also allows the user to describe the relationships (e.g., parent/child) of entities with one another, as well as enter other information such as contact information.

Importantly, in addition, the Properties Tool also can be easily engaged to create analyses and series from entities' websites. The purpose of enabling analysis of a website is to examine its structure—i.e., the directories that make up the website. (For more on the Analysis Tool, see below.)

In sum, the Properties Tool is used to:

- Create and manage a list of content creators (entities).
- Assign metadata and other properties to entities.
- Specify websites that entities are responsible for, and create analyses and series (explained below) based on those websites.

2.4.3.3. The Analysis Tool: Visualizing the Structure of a Website

Through the Analysis Tool it is possible to discern whether there is valuable content in the directories that comprise a website and, if so, to identify those chunks of content. "Series" refers to flexible aggregates of content that are analogous to archival series — which may be a whole website or a portion of it (e.g., only PDFs of annual reports), or even one individual page or document from websites. Loosely defined, a series is any collection of Web material that a user chooses to collect in one "bucket." In addition, series are used in order to drive the Workbench harvest operations. While series may be established within the Properties Tool, they can also be established and managed using the Analysis Tool, then harvested and packaged in the Harvest Tool.

The Analysis Tool has two functional areas:

- the Analysis screen, which provides visualization tools to aid in content selection decision-making and in series structure decisions. Here, too, a baseline analysis can be created against which to measure future website analyses;
- the Series screen, where series are created, edited, and managed; Series objects are kept; and Series harvests are regulated.

The Analysis Tool is used to:

- Analyze the structure of a website.
- Enter associated entities.
- Set a baseline analysis for comparison with future analyses.
- Adjust settings, such as spider settings and change notification threshold settings.
- Define a "series" for harvesting (e.g., harvest as an individual object), with option to associate it with an entity.
- Hold series objects prior to harvest.
- Schedule harvests of series.

In addition, operations for holding series objects and harvesting them may be accessed via the Properties Tool.

2.4.3.4. The Harvest Tool: Reviewing, Packaging, and Ingesting Harvested Content

All the harvests in the Workbench, including series harvests (via the Analysis Tool) and quick harvests, are listed in the Harvest Tool. The Harvest Tool is used to monitor the status of harvests and to provide an opportunity to review and modify the harvest before packaging it up and ingesting it into a repository. There may be single-object harvests or multiple-object harvests, depending on whether the option to harvest content as individual objects was selected in the Series details screen of an Analysis-based Series (i.e., in the Analysis Tool). The Quick Harvest feature schedules one-time harvests of content based on a URL inputted directly into the Harvest Tool.

After harvests are complete they may be reviewed, at which time additional metadata may be assigned. The user can render, or display, the harvested content within the WAW tool, off the Harvest Results page. The user can actually "step into" the harvested content at both the harvest starting point and at any other point in the website (via the website file structure display), and the software will render the website appropriately. The purpose of the display feature in the Web Archives Workbench is to allow the user to verify the correctness of what was harvested—“correctness” meaning that all the information expected to have been collected is collected. Once the harvested content is confirmed as correct, it then can be ingested into the user's local repository.

In sum, the Harvest Tool is used to:

- Monitor the status of harvests scheduled in the Analysis Tool.
- Delete completed harvests.
- Review completed harvest content, whether single-object or multi-object, prior to ingest.
- Review completed harvests; if desired, edit metadata and/or include/exclude content.

- Ingest harvested content into a repository.
- Launch a one-time quick harvest using the Quick Harvest Tool.

2.4.3.5. The Alerts Tab: Workbench Notifications

As mentioned above, the "Alerts" tab is not a tool but, rather, a feature for notifying the user of a variety of systems information. This information includes notification about errors, incomplete processes, completed processes, and new information (such as the discovery of a new domain, or a new folder encountered during analysis). In short, the Alerts Tab is used to review reports and alerts about Workbench functions.

2.4.3.6. The System Tools: Monitoring and Managing Workbench Activities

The System Tools tab contains a number of behind-the-scenes functions that affect and report on activities of the five main tools of the Workbench.

The System Tools are divided into four functional areas:

- the Audit Log page, which displays recent Workbench activities and events;
- the Spider Settings page, where the user can configure default Domain, Analysis, and Harvest spider settings, as well as create additional Domain, Analysis, and Harvest spiders with custom settings. Specifically, types of spider settings include — but are not limited to — depth (how deeply a website should be crawled, or spidered) and parameters of time (when, how frequently, and for how long);
- the Import/Export page, through which the user can import or export a variety of metadata commonly used in the Workbench. These include entities, domains, and subject headings.
- the Reports page, which generates printable reports on activities of the main five Workbench tools. It offers a view of in-development entity and series reports.

2.4.4. Web Archives Workbench Tools Summary

The Web Archives Workbench implements an archival approach to the selection and preservation of digital (Web-based) content. The Workbench automates much of the methodology embraced by the Arizona Model, particularly beyond the initial selection decisions made by the archivist (e.g., deciding at the start of the archiving process which website, or which part of a website, to capture and preserve). After selection parameters are set, the Workbench facilitates the capture and management of the digital materials in hierarchical aggregates--not unlike the archiving of print-based materials.

OVERVIEW OF WEB ARCHIVES WORKBENCH TOOLS		
Discovery Tool	• discover domains • group and prioritize domains	Comprising the Entry Points and Domains tabs, the Discovery Tool helps to identify potentially relevant web sites by crawling relevant "seed" Entry Points to generate a list of domains that they link to. At the end of this process the users have a list of domains that defines the sub-set of the web relevant for their archiving purposes. From here, the Properties and Analysis Tools are used to manage creator information about domains, and associate this information with harvests of content.
Properties Tool	• organize collection space • create metadata	Comprising the Entities tab, the Properties Tool is used to maintain information about content creators or 'Entities' (e.g., government agencies), and associate them with the domains and web sites they are responsible for. The Properties Tool also allows users to describe the relationships (e.g., parent/child) of Entities with one another, as well enter high-level metadata about them that may be inherited by content harvested from their web sites. Importantly, the Properties tool can also be used to create and associate Series with Entities' web sites. Series and harvests are then further managed using the Analysis and Harvest/Package Tool.
Analysis Tool	• visualize site structure • associate metadata • schedule harvests	Comprising the Analysis tab and the Series tab, the Analysis Tool provides website structure visualization tools to aid content selection decisions, and allows users to define archival Series, associate metadata with these series, and schedule recurring harvests of Web content. Harvesting activities are then monitored and managed in the Harvest Tool..
Harvest Tool	• review content • package for ingest in external repository	Comprising the Harvester and Quick Harvest tabs, the Harvest Tool lists all harvests within the Workbench, including Series harvests scheduled using the Analysis Tool as well as Quick Harvests. It is used to monitor their status, initiate the final harvesting and ingest steps for the completed harvests tracked in the Harvest Tool, including reviewing harvest contents and metadata before ingest. This is the final step in the Web Archives Workbench workflow. It also offers a separate Quick Harvest feature.
Systems Tools	• reports and settings	The System Tools manage and monitor Workbench activities, reporting on operations undertaken in the four other tools. It has four functional sections: an Audit Log page (shows recent Workbench activities); a Spider Settings page (parameters for spidering may be set here); an Import/Export page (for moving metadata); and a Reports page (for producing printable reports about activities performed by the other tools).

2.4.5. Behind the Scenes: OCLC's Technical Implementation of the Web Archives Workbench

An ISO 9001 company, OCLC has an externally audited Quality System based on the requirements of ISO 9001 as an aid for ensuring that products meet user expectations and specified requirements. OCLC's project development lifecycle is a process that specifies how OCLC services are marketed and developed. This process includes lifecycle documents such as project plans, requirements, design, test plans, operations support plans and post-project reviews. The Web Archives Workbench program followed this lifecycle.

The WAW program was divided into three main projects and many smaller releases in order to reduce risk and to create a feedback loop allowing refinement of the requirements based on previous releases. There were three major software releases, plus approximately 20 additional releases over the course of the three-year program. The three main development projects were based on the main areas of functionality of the tool suite: (1) Domain and Entity, (2) Analysis and Packager, and (3) Site Analysis and Change Management. Though the Domain and Entity features in WAW were somewhat functionally simple, the Domain and Entity project carried a significant amount of risk because it built the technical foundation on which the rest of the project would rest. The Site Analysis and Change Management tools were risky due to the usability issues involved in clearly representing to the user the process of harvesting and evaluating changes to websites. Throughout the project one of our main concerns was how to represent the Arizona Model in a clear and usable way in software. (This concern is addressed in the section "The Web Archives Workbench Workflow.")

Based on early discussions, the system began to be seen as a "workbench," into which components and systems would be incorporated and dropped over time—perhaps because users would prefer to apply some of their local tools or perhaps because they would have multiple tools for a given task. Additionally, each component would grow its data quality over time, therefore forcing the rest of the system to adapt easily to evolving specifications and data versions. Therefore, the architecture is designed for location, interface, and data-exchange transparencies, which means that changes in those three main areas are expected to drive all other system characteristics.

The high level technical architecture of the system was specified using the Reference Model - Open Distributed Processing (RM-ODP).[4] This framework uses various

[4] For the specification, see "The ISO Reference Model for Open Distributed Processing – An Introduction," at http://www.enterprise-architecture.info/Images/Documents/RM-ODP2.pdf.

views of a system, including a domain model view, an information view, an application view, and a technology and deployment view.[5] Using this framework, OCLC created the following early domain model of the system. (See Fig. 4 on next page.) Some of the boxes in this domain model were later removed from the requirements, as our understanding of the system to be built changed over time.

The architecture consisted of several layers: client, integration, service, and persistence. The client layer consisted of a user interface implemented using the Struts framework as a model-view-controller to structure to the code. The second layer is a Web services layer that provides the hooks for a client to talk to the application (although the code was not used in this way). This layer also provides integration between tools and translation between the internal and external representations of the data. Each developing WAW tool (Entity, Analysis, Domain, etc.) implemented a consistent Helper API to allow the user interface layer to Add/Update/Delete/Search single or multiple objects. The Oracle database provided a persistence layer. Once the high level design was produced, a detailed design was produced for each tool. OCLC created use cases for all main activities in each of the tools.

[5] In RM-ODP the architecture of a system is described by 5 views (essentially 5 different points of view) reflecting the separation of responsibilities between business sponsors, developers, and support staff. Those views are:

1. Enterprise - community, enterprise objects (domain model), objectives (requirements / use cases), roles
2. Information - schemas, object attributes, data boundaries, constraints, semantics
3. Computational - components, interfaces, interactions, contracts
4. Engineering - transparencies (location, access, failure, persistence), nodes, channels
5. Technology - technologies & products (the only dependence on specific products and implementation packages)

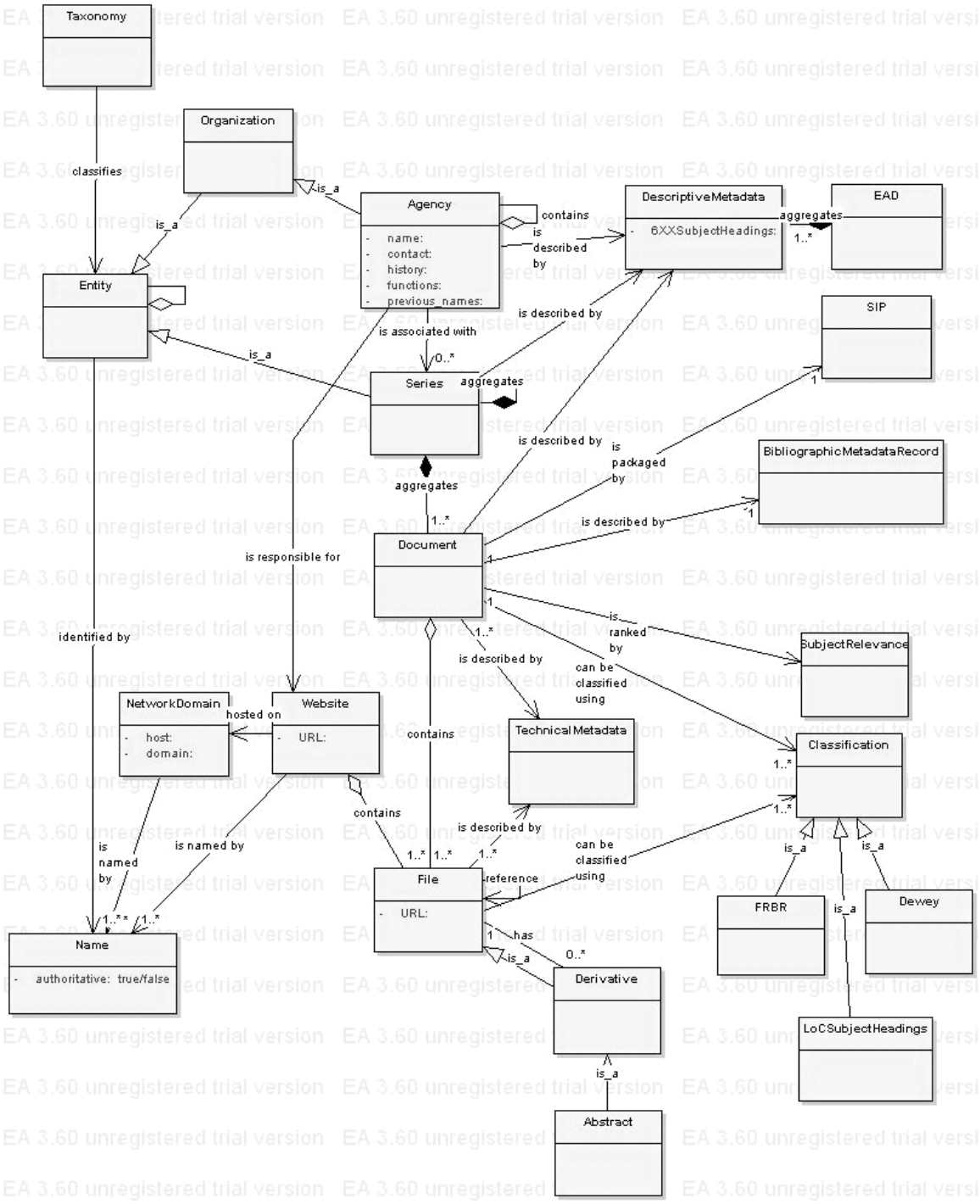


Figure 4: Diagram showing OCLC's early domain model of the system that eventually developed into the WAW suite of tools.

Each developer worked in his own "sandbox," where a WAW interface was set up for his exclusive use. The work of multiple developers was integrated into a

development test environment called "Baseline." This way, product managers and test analysts could review work in progress in Baseline. When Baseline was ready it was migrated into a Quality Assurance environment, where formalized testing was done against a test plan. For major installs, Baseline was also installed at UIUC for additional testing. The final step of the development process was to deploy the software into a production environment.

The Web Archives Workbench was released as an open-source package on SourceForge in October 2007. Release documentation includes detailed installation instructions and a detailed User Guide for understanding and using the tools.

- WAW Release home page: https://sourceforge.net/projects/webarchivwkbnch/
- Administration Guide: https://sourceforge.net/project/showfiles.php?group_id=205495 (also provided as an Appendix item in this report)
- User Guide: https://sourceforge.net/project/showfiles.php?group_id (also provided as an Appendix item in this report)
- WAW software package: http://webarchivwkbnch.cvs.sourceforge.net/webarchivwkbnch/webarchivwkbnch/

The Administration Guide has runtime environment requirements for WAW. It also has a list of all 3rd-party software used by WAW in the Incorporated Code section of the document. The third-party software is included in the WAW distribution (refer to link for WAW software package). An OCLC subscription is not required to use WAW or to use this third-party software. Please see the HOWTO-build-install-locally.txt file in the WAW deployment for additional information.
The WAW tools, as developed by this project, will continue to be made publicly available indefinitely through SourceForge. In addition, in 2008 OCLC released a new array of services incorporating components of the WAW tools into a workflow with CONTENTdm, WorldCAT, and the OCLC Digital Archive.

2.5. Findings - User Feedback

Testing of the WAW tools was undertaken in varying degrees by the original project content partners, as well as by several volunteer organizations. Feedback about their experiences working with the tools was gathered during large-group project meetings at OCLC, as well as through phone conversations and e-mail exchanges.

The overall response indicates that the Web archiving approach of the WAW tools was "elegant" and worth consideration, but in practice content partners generally did not implement the full functionality of the tools. Thus, the potential benefits of

applying an archival approach to the Web were not realized completely. Reasons for this partial implementation have to do with inadequate resources and time toward training for the use of the tools, which also points up their complexity (explained in further detail below). The Web Archiving Workbench is powerful and extensive in terms of web harvesting and content, or series, analysis, but—according to the feedback from our content partners—at a cost of heuristics and usability. Not surprising, the Quick Harvest functionality (which, because series analysis is not an option in it, involved fewer steps and thus less management than the regular Harvest tool) was engaged most often; for some, the Quick Harvest feature became a much-valued component of their daily workflows. Changes in content delivery approaches—such as from static Web pages to database-driven pages—constituted another reason for not applying the full functionality of the tools.

2.5.1. Limited Resources and Limited Time

During their participation in the ECHO DEPository project, state library and archives partners remained under continual operational pressures to respond to the need for capturing content from agency websites. Some partners tested the WAW tools while continuing to use other Web content capture approaches in order to meet their immediate obligations, leaving fewer resources to focus on the WAW tools. Because the tools were still under development, testing of the various phased releases may also have been difficult to incorporate into daily workflows. Support from the project (in the form of interns) had been planned but was geared to the early releases of the Workbench, before the full functionality of the tools was implemented. In hindsight, putting project resources toward direct work with content partners, as originally intended, might have resulted in more use of the full functionality of the tools, especially if timed more specifically to coincide with later, more fully functional, software releases.

2.5.2. Complexity of the Tools

According to user feedback, the Quick Harvest and Discovery tools were easiest to use, because they could be set up quickly and incorporated into existing workflows without increasing the need for new resources. The full functionality of the tools involves understanding a process with a greater level of complexity than that presented by the Quick Harvest option; partners reported that it was easier to use the Quick Harvest and Discovery tools, rather than expend time and resources for learning, or testing, the tools suite as a whole. Further, some content partners report that the complicated interface of the tools was a barrier to using them to their fullest potential.

2.5.3. Web Content Delivery

The assumption proposed by the archival model — that a website and its directories are similar to an archival record collection and set of record series — does not apply

today as easily as it did when the model was first proposed in 2003. An increasing amount of content is now delivered through database-driven websites rather than through static Web pages. The relationships between content items that may have been obvious when stored in a file directory are not always apparent when stored in a database. Therefore, crawling domains to find potential content to harvest and applying inherited metadata according to a directory structure are now less useful approaches than they were just a few years ago. Nonetheless, despite this shift in how information is delivered via websites, the concept of content inheriting metadata from previously harvested content, and then associating that content with an existing aggregate collection, continues to be useful for making automated harvest processes more effective.

2.6. Conclusions and Next Steps

State librarians and archivists continue to search for the best methods for capturing Web content based on their specific mandates and the resources they have available to them. Recent developments in Web archiving services and tools provide new opportunities for partnering with others and for exploring new workflows. The Web Archives Workbench tools are one option among many. They automate the methodology prescribed by the Arizona Model, which is premised on key archival practices, such as observation of provenance and adherence to original order. The four main tools (Discovery, Properties, Analysis, and Harvest) enable the identification, selection, description, and packaging of digital content. In addition, the WAW suite includes functionalities for error notification, as well as System tools for overseeing and reviewing Workbench activities (in the form of audit logs, spider settings, metadata import/export options, and reports on the activities launched by other tools in the suite). The lessons learned from developing the Workbench, and the underlying archival model used to direct its development, underscore the merging roles and responsibilities of archivists and librarians in the digital environment and the need to re-evaluate and re-envision workflows.

Moreover, the continuing mission and significance of this work have been affirmed in the second phase of NDIIPP. For example, the University of Illinois, OCLC, and the University of Maryland have partnered to develop a stand-alone, open-source metadata extraction tool intended to provide access to archived content – a kind of next step for the Web Archives Workbench. In addition, in the State Initiatives component of NDIIPP, a selection of state libraries across the nation are collaborating to develop tools and service models for the management and preservation of state government digital materials. These projects address digital preservation in a variety of contexts, including disaster readiness and the recovery of data. Through the State Initiatives work, NDIIPP is addressing the fundamental issue of keeping at-risk state government resources viable as part of our national heritage and record.

3. Repository Evaluation and Interoperability

Another component of the ECHO DEPository project is the evaluation of various open source repository software applications, with a focus on how these applications support activities of an institution or organization interested in providing services associated with a trustworthy digital repository. This section describes the development of an evaluation framework based on the first draft of the 2005 RLG/NARA *Audit Checklist for the Certification of a Trusted Digital Repository, Draft for Public Comment* (Audit Checklist), our repository testing and findings, and how these activities led to the development of a tool suite (the Hub and Spoke) for supporting repository interoperability and the collection of metadata important for preservation.

3.1. Repository Evaluation

3.1.1. Building an Evaluation Framework: Applying the Trusted Digital Repository Checklist to Repository Evaluation

Our goal is to provide an evaluation framework that reflects current thinking on digital preservation standards to help curators of digital collections librarians and archivists assess digital repository systems, with a focus on their ability to support long-term preservation.

The 2005 RLG/NARA *Audit Checklist for the Certification of a Trusted Digital Repository, Draft for Public Comment* provided a vital starting point. The Audit Checklist was developed by a joint task force from RLG and the National Archives and Records Administration (NARA). It provides a means by which an institution can perform a self-evaluation to determine how well it is positioned at an organizational level to provide an expected level of trustworthiness as a digital repository. We considered it to be a state-of-the-art articulation of what it means, at an organizational level, to be a successful curator of digital resources.
We therefore decided to use this document as a starting point, and provide support for using those portions that are relevant to software as a 'lens' for considering repository software systems.

Our project team reviewed each Audit Checklist item with the question in mind, "How might a repository software system support an organization in meeting this criterion?" Some Checklist items are applicable to software applications; others are not. We isolated the relevant items, and, through much discussion, testing and review, developed a system of annotations to describe how each particular relevant Checklist item might be applied to assessment of repository software systems. This adapted Checklist is our Annotated Audit Checklist.

Our Annotated Audit Checklist evaluation framework is provided in this report in **Appendix 6.3**. As the version we used, the 2005 RLG/NARA *Audit Checklist for the Certification of a Trusted Digital Repository, Draft for Public Comment* has now been succeeded by the Trustworthy Repositories Audit and Certification: Criteria and Checklist (TRAC), we have included mapping to equivalent sections of the new version.

Additional details and examples of annotations are provided in **Appendix 6.4**. The following overview is extracted from this source.

3.1.1.1. Findings: the Annotated Audit Checklist As a Repository Evaluation Framework (from Kaczmarek et al, 2006)

We found the process itself of adapting the Audit Checklist as a framework for our repository software application evaluation to be a useful learning experience. Situating our evaluation within the original Audit Checklist provided a framework to discuss repository software applications without losing sight of the larger organizational context. As we used it to document our repository installation and experimentation experiences, we found the annotated Audit Checklist provided a good framework for looking at repository software applications within the context of digital preservation. However, information about other aspects of software not directly related to preservation (e.g., ease of installation, ease of maintenance, programming language used) do not fit well into this framework. Importantly, we have also found that the process itself of annotating the original Audit Checklist provided a forum for the project team members to begin discussions that have opened up opportunities to explore our individual assumptions about various checklist items and our interpretations of terminology. Through these discussions we established directions to take our evaluation activities further.

3.1.2. Repository Testing: Ingest and Export Tests On Four Key Open-source Repositories

The four open-source repository software applications that were tested were DSpace, Eprints, Fedora, and Greenstone. The collection items used as test data are described in detail below. Out testing approach and methodology are explained in Section 3.1.3, also below.

3.1.2.1. Test Data: a Heterogeneous Canonical Set

In order to test the repositories, a number of heterogeneous collections of digital items were identified. Each of these collections had varying structures, formats, and existing metadata. An overview of each collection follows below.

3.1.2.1.1. Aerial Photos

This is a relatively small collection of scanned aerial photographs from a couple of Illinois counties. It consists of 1,021 distinct scanned photographs and their accompanying metadata, which accounts for 2,042 distinct files for a total of 235 megabytes. The metadata are based on Federal Geographic Data Committee (FGDC) Geospatial Metadata Standard. The scanned images were JPEGs of screen resolution quality; the archival quality images were not available for this project.

3.1.2.1.2. DLI (Digital Library Initiative) Journal Articles

The DLI (Digital Library Initiative) collection consists of 85,650 distinct journal articles on the subjects of science, technology, and engineering from five different publishers. This collection was created as part of the Grainger Libraries earlier NSF and CNRI funded Digital Library test bed. This was by far the largest collection with 2,247,455 files for a total of 76,148 megabytes. Each journal article typically consisted of several instantiations, typically including XML and SGML conforming to one of several different DTDs plus a PDF version, but in some cases also Postscript or TeX versions, plus all of the associated files such as images and metadata, which also occurred in several different formats.

3.1.2.1.3. WILL Public Radio Broadcasts

WILL is the local Public broadcasting station, and this collection consists of the audio recordings for a selection of its Focus 580, daily talk radio programs. A total of 310 programs are included. Each program has a WAV audio file plus two XML metadata files for a total of 930 files and 82,456 megabytes. The metadata was originally in a Microsoft Access database.

3.1.2.1.4. Vincent Voice Audio Collection

This collection is a selection of audio recordings from the Vincent Voice Library at Michigan State University. It consists of 209 recordings, many of which are composed of several audio WAV files. There is an Encoded Archival Description (EAD) file associated with each recording for a total of 3,515 files and 110,186 megabytes.

3.1.2.1.5. DOQ (Digital Orthophoto Quadrangles) Data

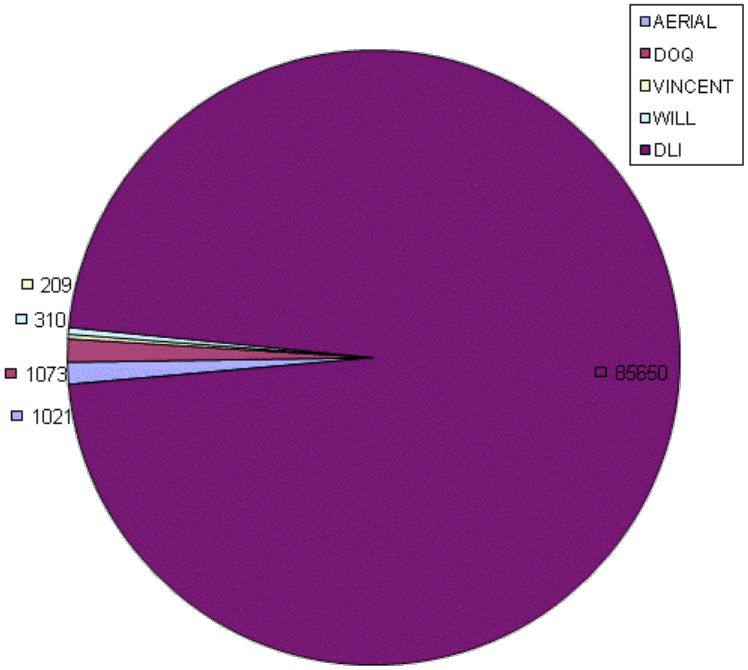
This is a collection of 1,073 high resolution, Digital Orthophoto Quadrangles of the Chicago area. Each DOQ consists of six files: the image TIFF file, the 'world file' used for geo-referencing the TIFF, an FGDC1 XML metadata file, plus a text-only version of the metadata, and the DTD for the metadata file, and an XSLT stylesheet for the metadata.

3.1.2.1.6. "The Canonical Set"

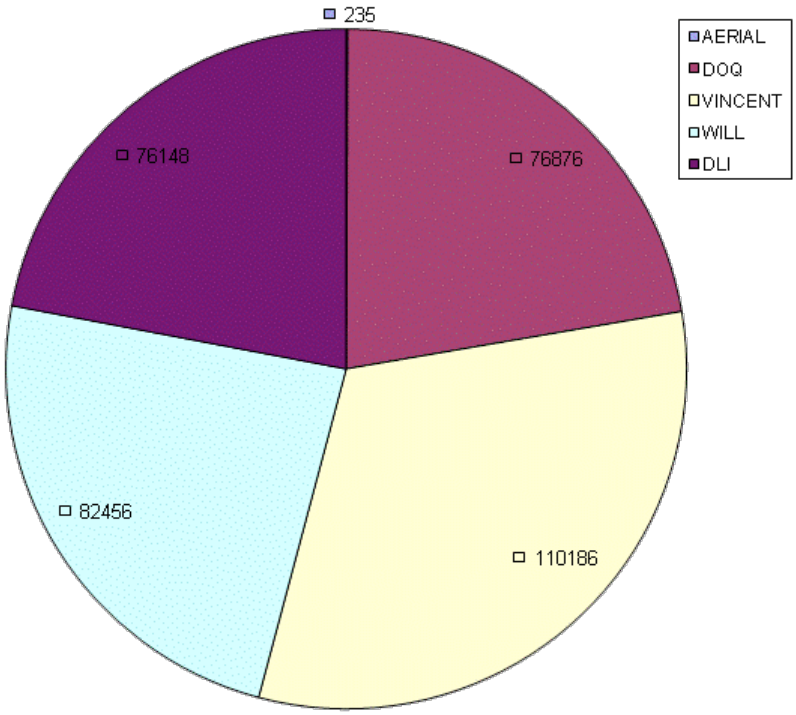
The first repository tested was DSpace. For each collection, specialized scripts and XSLTs were written to arrange the items and metadata in such a way that they could

be ingested into DSpace using the ItemImport utility. This process included moving the files associated with each "item" into a single directory, creating descriptive metadata in DSpace's idiosyncratic Qualified Dublin Core format, and creating a content manifest.

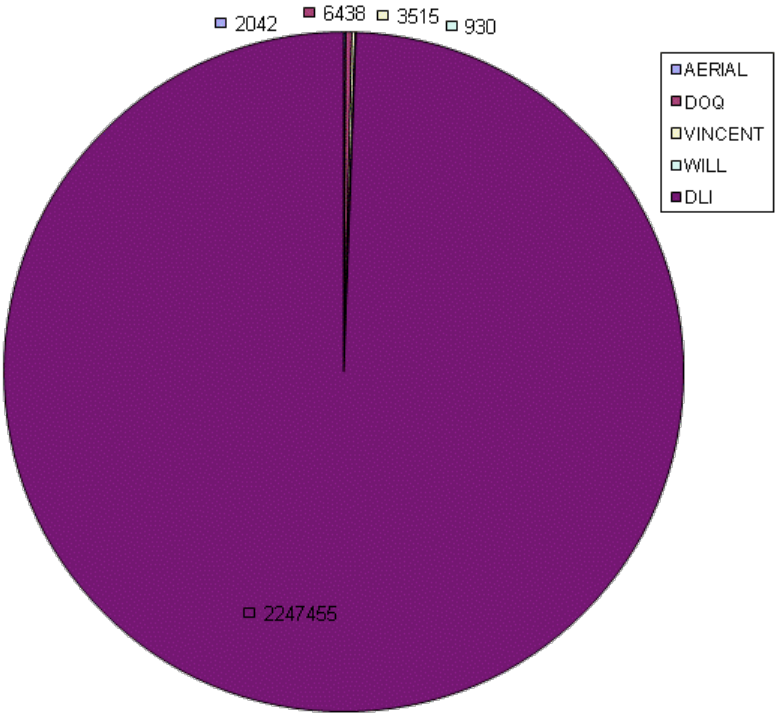
The DSpace bulk ingest package format was accepted as the baseline configuration from which all other processing would occur. The collection of all the digital packages in this format became our "canonical set." See figures below for various breakdowns of all the files in all this set:



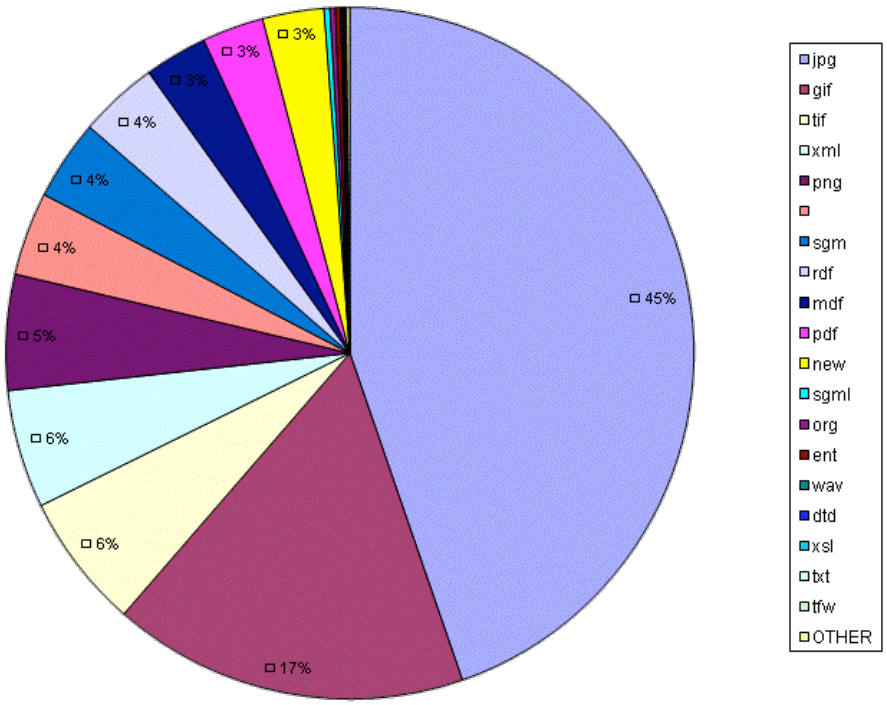
Canonical Test Set:: Number of Packages by Collection



Canonical Test Set: Total Megabytes by Collection



Canonical Test Set: Number of Files by Collection



Canonical Test Set: Number of Files by Filename Extension

3.1.3. Testing Approach and Methodology

In a nutshell, our process was to install a particular repository, do whatever was necessary to ingest our canonical dataset into the repository, do whatever was necessary to export our canonical dataset back out of the repository, and record in a narrative fashion our findings, especially in the context of digital preservation and, also, of our Annotated Trusted Digital Repository Checklist. It needs to be mentioned that the Annotated Trusted Digital Repository Checklist and our concept of what was actually required for long-term digital preservation was being continually revised in parallel with the repository evaluation process. More details of the testing approach and methodology are provided below.

First, different project staff members, consisting primarily of graduate research assistants, were assigned to each of the different repositories to be evaluated. Cooperation between evaluators was encouraged, especially when different skill sets might be needed in performing an evaluation. Some of the evaluators had library backgrounds while others had technical computer backgrounds.

The approach was fairly freeform and the evaluators had some leeway in the details, but in general they followed this rough outline:

- The first step was to become familiar with the repository to be evaluated. This could involve review of any previous evaluations or other written material about the repository, including the documentation provided by the repository itself. This step culminated in the installation of the repository on our test server.
- The next step was to develop a process for ingesting our canonical dataset into the repository. This required the evaluator to gain a good understanding of the details of the repository's supported metadata formats and supported file structures, as well as the repository's programming interfaces or batch processing tools that could be used to facilitate the ingest. The evaluator would also need to become familiar with our canonical dataset at this point, if they were not already familiar with it. The ingest process generally consisted of these steps:
    - Develop mappings between the various metadata formats represented in the canonical dataset and the metadata schemas required by the repository. These mappings could be implemented using XSLT transformations or in some cases by writing customized computer programs.
    - Package the metadata and files in a format that is digestible by the repository. This could be as simple as creating a text file manifest, or naming files according to some standard and putting them all in a certain directory structure, or as complex as creating METS or FoxML XML packages. Similar to the metadata, these packages are usually implemented using some combination of XSLT and customized computer programs.
    - Finally, the actual ingest needed to occur. Once again, this could be as simple as running or activating the repository's native ingest tool, or as complex as writing a custom ingest program that uses the repository's low-level programming interfaces. If the repository supported a native batch ingest mechanism every attempt was made to use it as is before resorting to the creation of any customized ingest programs.
- Often development of the ingest process was iterative, consisting of developing and implementing a process, testing it, and refining it until the entire canonical set could be reliably ingested.
- After the canonical dataset had been ingested, the process was reversed, and the data was exported or disseminated back out of the repository. Similar to ingest, the dissemination could be as simple as invoking native repository functions, or as complex as writing a custom program—although we

preferred to use native batch export capabilities if the repository had support for any. This process was also often iterative.
- Evaluators were encouraged to record their processes and findings about the repositories throughout this process.
- Finally, reuse of transformations and computer programs between different repositories was encouraged.

In parallel with the repository evaluation processes described above, team members also participated in the review of the Trusted Digital Repository Checklist, so that both of these tasks informed the other in an iterative fashion. The simultaneous Trusted Digital Repository Checklist review and the repository evaluations culminated in the evaluators being asked to apply our Annotated Trusted Digital Repository Checklist to their repository evaluation findings, which are provided in the appendices of this report.

Unfortunately, one of the pitfalls of employing graduate assistants on a long-term project like this is that they eventually graduate or move onto other assistantships as their educational goals progress or change. While we strongly encouraged our graduate assistants to document their work, we found in some cases that their notes were not always detailed enough for us to accurately reflect their findings as we applied their evaluations to our Annotated Trusted Digital Repository Checklist. This sometimes required us to revisit, or recreate, a test for a given repository in order to address one or more of the checklist items.

Another outcome of the repository evaluations is that as we progressed with ingest and export testing of the different repositories, one of our goals became to be able to reliably move a collection of digital objects between any two of the repositories that were being evaluated and back again (round tripping). This was the genesis for our currently developing Hub and Spoke repository interoperability architecture.

3.1.4. Repository Testing Findings: Narrative Reports, and Annotated Audit Checklist Commentary

The following open source repositories were evaluated:
- DSpace: Version 1.2.2 with later upgrade to 1.3.1
- Eprints: Version 2.3.13
- Fedora: Version 2.0, with later upgrades to 2.1, 2.1.1 and 2.2
- Greenstone: Version 2.6, with upgrade to 7.7

Overview reports were produced for each repository, containing the following sections. These are provided in **Appendix 6.5**, Repository Testing Findings: Narrative.

- Repository Overview
- Testing and Technical Environment
- Methodology
- Findings
- Conclusion

We also documented our experiences using our Annotated Checklist evaluation framework. This documentation is provided in **Appendix 6.6**.

3.1.5. Conclusion and Next Steps

Two key overall findings were typically low out-of-the-box support for interoperability and low support for emerging preservation standards. During the development of our test bed we found ourselves developing a number of different though similar customized scripts and programs for exporting digital packages from one repository system and importing those digital packages into another repository system. The repository systems themselves had very little in common that would facilitate this task. They typically supported different descriptive metadata formats, had no support for provenance metadata, offered little or no support for technical metadata, and employed different means of identifying the files constituting a package. The development of an in-house tool to facilitate data interoperability between multiple repositories without the need to develop customized mechanisms for each repository combination therefore soon emerged as a key task to support our repository evaluation activities.

At the same time, we were also coming to a more structured understanding of emerging digital preservation standards, specifically early drafts of An Audit Checklist for the Certification of Trusted Digital Repositories (RLG, 2005; Kaczmarek, Hswe, Eke, & Habing, 2006; Kaczmarek, Habing, & Eke, 2006) and the PREMIS Data Dictionary for Preservation Metadata (PREMIS Working Group, 2005). We began to see that a formally-developed interoperability architecture designed with a focus on providing additional support for retention of provenance and technical metadata could be a valuable and practical project deliverable, and one with immediate application in our own libraries and in other institutions that commonly implement multiple repository systems to manage and preserve digital collections.

These findings led our proposing the Hub and Spoke tool suite as an additional project deliverable. This work is described in detail in the next section (Section 3.2).

3.2. Hub and Spoke Architecture (HandS): Supporting Repository Interoperability and Emerging Preservation Standards

This section describes the development of the Hub and Spoke (HandS) tool suite, built to help curators of digital objects manage content in multiple repository systems while preserving valuable preservation metadata. Implementing METS and PREMIS, HandS provides a standards-based method for packaging content that allows digital objects to be moved between repositories more easily while supporting the collection of technical and provenance information crucial for long-term preservation. (Note that related project work, investigating the more fundamental semantic issues underlying the preservation of the meaning of digital objects over time, is profiled in Section 4.)

3.2.1. HandS Overview

The HandS is a suite of tools built to support moving content between repositories while generating and maintaining PREMIS-based technical and preservation metadata. It emerged out of project activities to evaluate open-source repositories (see Section 3.1) in which we found typically low out-of-the-box support for interoperability and low support for emerging preservation standards. The next section describes the impetus and rationale behind the HandS development in more detail.

3.2.2. The Need for Interoperability and Preservation Support

3.2.2.1. Institutions Commonly Rely on Multiple Repositories

There are currently many different digital repositories in widespread use, including DSpace, Greenstone, Fedora, EPrints, and CONTENTdm, along with digital archive services like those from OCLC and CDL. There are also many different sources of input into these systems, such as from web crawlers like Heritrix or packaged content from OCLC's Web Archives Workbench, as well as numerous digitization and scanning services. It is also not uncommon for several of these systems to be in use within a single institution. If curators wish to share data internally, or with other institutions or consortia, then multiple repository systems very likely will come into play. Repository interoperability issues also emerge as institutions update or replace their repository systems, and must migrate content from an existing repository system to its replacement.

3.2.2.2. Out-of-the-box Repository Interoperability is Low

Our repository evaluation experiments and our experiences with repositories in production at our own institutions show that the native ability for repositories to interoperate is typically very basic. Almost none of the systems we tested were able

to operate with one another beyond a rudimentary level, usually restricted to the OAI Protocol for Metadata Harvesting (OAI-PMH) for Dublin Core. If any OAIS concepts are implemented (and few are), such as the use of submission or dissemination information packages (SIPs and DIPs), these implementations vary greatly. In an ideal OAIS-compliant world, a DIP from one repository should be a SIP to another. However, in reality, a dissemination package produced by DSpace cannot be used for submission into Eprints. Because of these inconsistencies, achieving any real interoperability between repository systems usually entails some level of custom software development. Further, any time a new repository is added to the mix, new software will need to be developed in order to accommodate the added repository.

3.2.2.3. Support for Emerging Preservation Standards is Low

Few of the current repositories have any explicit support for preservation, such as for collecting preservation metadata as articulated by PREMIS, or activities to support preservation such as format migrations or checksum validations as outlined in the Trusted Digital Repository Checklist. For an institution that deploys several repository systems, a task as simple as performing consistent backups to off-line storage can become complicated by the fact that the systems store their underlying data differently. There may be data stored in relation databases, XML databases, RDF triple stores, and various file systems – all of which must be backed up, and may require different backup techniques.

In summary, the general lack of repository support for interoperability and for emerging repository standards at a time when libraries and other institutions commonly rely on multiple repository systems to manage, share and preserve content, is the fundamental impetus behind the development of the HandS tool suite. The key principles of interoperability and preservation, and the approaches implemented in the HandS to support them, are examined more closely in the next section, followed by a functional and technical overview of the HandS tools.

3.2.3. Hub and Spoke Key Principles

The Hub and Spoke approach is based on the two key principles of interoperability and preservation, with the understanding that interoperability is not only an end unto itself, but it is also critical for preservation.

3.2.3.1. Interoperability

To reduce the complexity of interoperability, the Hub and Spoke uses a common packaging format which is used for interchange of digital resources between different repositories. Digital packages coming from a repository are transformed into this common format before any further processing, and digital packages are transformed from this common format into the native repository format when being

placed into a repository. The idea is to reduce an N2 problem into a 2N problem as shown in Figure 5 below.

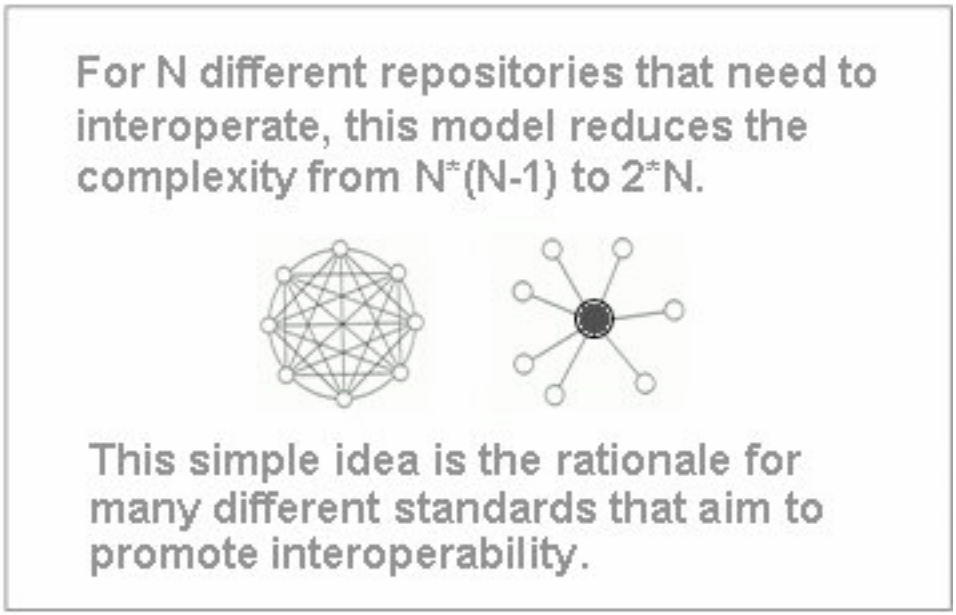


Figure 5: Interoperability Standards: a Simple Idea

3.2.3.2. Preservation

The second key principle is that the common packaging format as well as the processes that act on that packaging format should not only support interoperability, but should also promote preservation. This principle treats the common packaging format as an archival information package (AIP) in the OAIS model. The assumption being that one reason packages are being moved between repositories is for preservation.

There are several features of the Hub and Spoke architecture that promote preservation. One key preservation feature is the reliance on current best practices regarding preservation metadata, primarily informed by PREMIS. The Hub and Spoke is especially concerned with technical metadata about the files and bitstreams which comprise a digital package, and also with provenance metadata about the events that occur during the package's lifetime, including events pertaining not only to the files and bitstreams, but also to the metadata itself. The technical metadata is used to validate the files and bitstreams throughout a digital object's lifetime and are updated as required, for example when a format transformation occurs. The provenance metadata is also updated throughout an object's lifetime. The tools that implement the HandS architecture perform these

actions automatically as required during processing, but the data are always present in the packages so that other systems can also perform these actions as needed.

Another key preservation feature is the treatment of the packages themselves. For purposes of repository interoperability and also to support preservation, the Hub and Spoke framework treats the instantiations of the packages as first class digital objects. This means that when a HandS package is transformed for ingestion into a specific digital repository, not only are the metadata, files, and bitstreams that comprise the package decomposed as appropriate for the repository and uploaded, but the package itself (in our case a suite of METS files) is also treated as a digital object to be uploaded to the repository. Later when the digital package needs to be disseminated from the repository, not only are the metadata, files, and bitstreams available for download, but also the original HandS package. This allows the HandS system to compare the package as it was originally ingested to how it now appears as disseminated from the repository. This process, we feel, is critical for preservation in an environment of heterogeneous and changing repositories. Another aspect of this treatment of the Hub and Spoke packages as first class digital objects is that we can create snapshots of individual packages at points in time and also record preservation metadata data about the package snapshots. The HandS tool suite currently implements this concept as a master package which references time-stamped snapshots of the main package. The master package also records preservation metadata about the snapshots. This approach is explained in more detail in the next section, which describes the concrete implementation of the HandS packages using METS.

3.2.4. METS Profile

To realize the above principles, we wanted to utilize the prevailing digital library standards as much as possible. To that end, we adopted METS as the packaging standard, PREMIS as the preservation metadata standard, and MODS as the descriptive metadata standard. We also optionally utilize several format-specific technical metadata standards such as MIX and textMD for image and text objects respectively, among others. Our METS profile, the ECHO DEP Generic METS Profile for Preservation and Digital Repository Interoperability (Habing, 2005), is currently registered with the Library of Congress.

As already described, the primary focus of the HandS METS profile is to enable repository interoperability and to support preservation of repository content. Because of the strong focus on preservation rather than access, the HandS profile is relatively noncommittal regarding file formats or structures; instead, special attention is given to administrative and technical metadata, particularly to integrating the PREMIS data model and schema into METS. We anticipate that our file format-agnostic HandS profile may be overlaid on top of, or inherited by, other profiles that better define a particular file format or structure, providing them with

added support for preservation or interoperability. An example of this arrangement, where a format-specific METS profile is implemented as a subclass of the PREMIS-focused HandS profile, is the ECHO DEP METS Profile for Web Site Captures (Habing, 2006), also registered with the Library of Congress. Thus the ECHO DEP Generic METS Profile for Preservation and Digital Repository Interoperability is generally not concerned with rendering or making accessible any particular representation of an object, but it is concerned with preserving the object and its representations, including the history of how those have changed over periods of time. In this context, preservation refers to short-term interoperability, preserving the representations and metadata as a digital package is moved between two different repositories. It also refers to the long-term preservation of the package and its history as it exists in various repositories for long periods of time and undergoes various "preservation actions" such as fixity checks, normalizations, or format migrations.

Note that though the profile is generally agnostic about almost all aspects of a digital object's representation, such as structure or file formats, we have made some pragmatic concessions, such as mandating at least MODS for the primary descriptive metadata (dmdSec) while at the same time allowing multiple alternative descriptive metadata sections. The alternative descriptive metadata sections are used as a means to record various versions of these metadata as they have existed in different repositories or at different points in time. A potential usage scenario can be illustrated in the following migration example using our METS profile:

1. We start with a digital object whose original source descriptive metadata is in the MARCXML format. Because our profile requires MODS as the primary descriptive metadata, the MARCXML will be transformed into MODS, and the MODS will be stored in the METS document along with a provenance statement with some details about the transformation, especially identifying the source metadata format. However, because descriptive metadata are considered to be a significant part of the representation of an entity and because transformations between metadata formats are often imperfect, the original MARCXML format is also stored in the METS document as an alternate metadata format.
2. Now suppose that the digital object is to be ingested into DSpace. DSpace, however, does not have native support for the MODS or MARCXML metadata formats; therefore, as part of the ingest process, the MODS must be transformed into the idiosyncratic Dublin Core (DC) metadata format that is supported by DSpace. This metadata format is also added as another alternate descriptive metadata format to the METS document, along with a provenance statement describing how this new DC format was derived from the primary MODS format.

3. Next imagine that the object exists in DSpace for some period of time during which the descriptive metadata undergoes some revision, such as the addition of new subject terms or the addition of an abstract. Now the object is to be disseminated from DSpace for ingest into some new repository. This could trigger the addition of another alternate descriptive metadata section to the METS document. This alternate format would conform to the idiosyncratic DSpace Dublin Core format, but the provenance statement would specify that this DC format represents a newer version of the descriptive metadata than was originally ingested into DSpace.

The above scenario would produce a chain of descriptive metadata formats, such as MARCXML (original) → MODS (primary) → DC (version 1) → DC (version 2), with provenance PREMIS event statements adequate to determine the sequence of events that led to this chain. As part of this profile we also envision future processes that might reconcile later metadata revisions and merge those revisions back into a new primary MODS descriptive metadata section. The preservation of semantics during these types of migrations is one of the concerns of semantic preservation described in the final section of this report (Section 4).

Because we feel that administrative metadata are important for preservation, this profile is fairly prescriptive when it comes to the administrative metadata, which can be associated with almost all of the sections that make up a representation: structures, files and bitstreams, and descriptive metadata. Particular attention is paid to the technical and provenance metadata associated with these METS sections.

3.2.4.1. Master METS Profile

Another key idea behind our METS profile is the idea of a Master METS document. Each package in the HandS architecture consists of a single Master METS document, one or more METS Snapshot documents, plus all the files and bitstreams that are referenced from the METS Snapshots, as shown below in Figure 6.

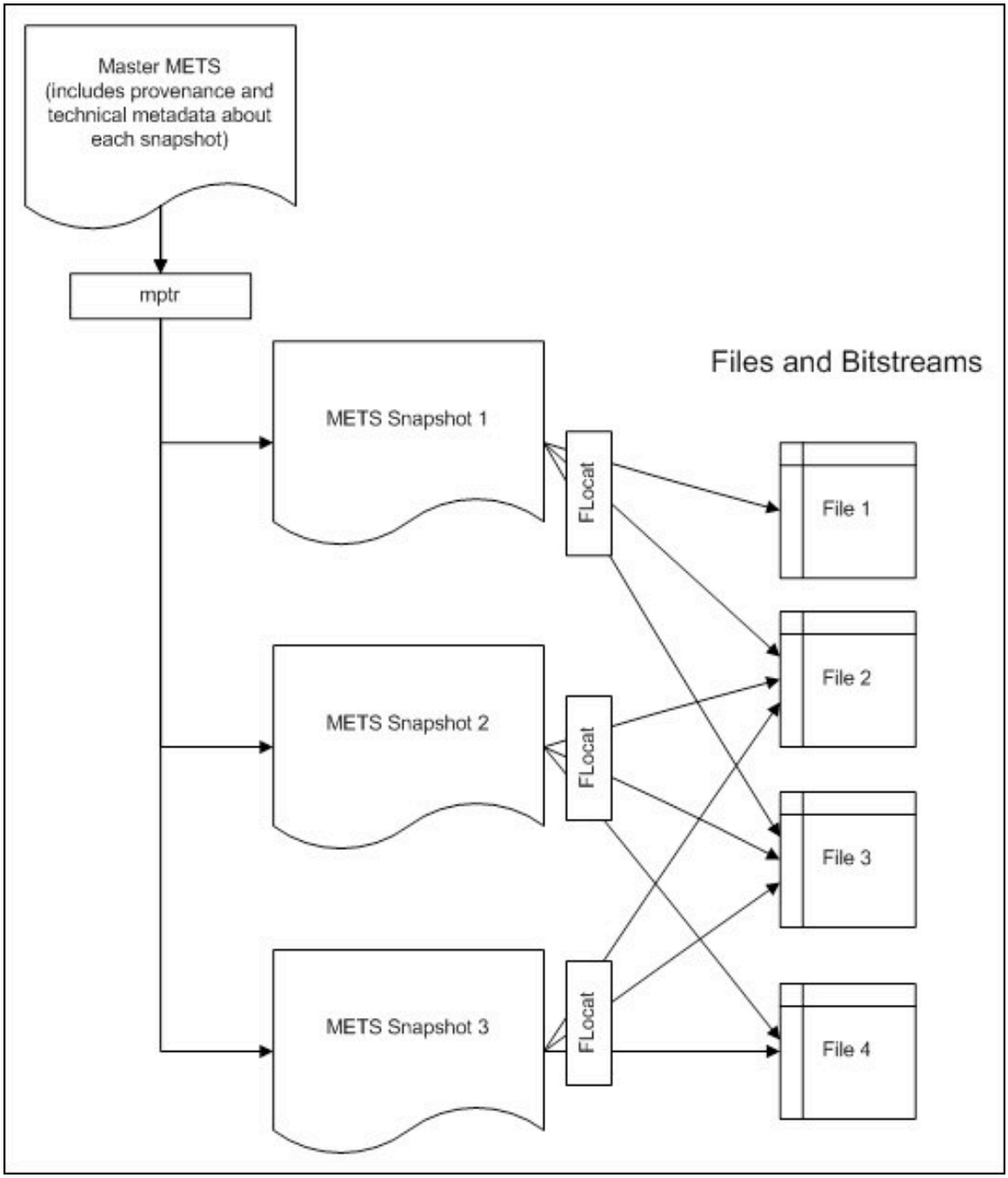


Figure 6: Master METS Showing Multiple Snapshots and Associated Files

Each Snapshot represents a version of the digital package at a point in time, usually when the package is either retrieved from or placed into a given repository. Nearly any aspect of a digital object's representation may change with time, including descriptive metadata, structure, and, as illustrated above, even the files referenced from a package may change over time, perhaps as format migrations occur. These changes are recorded as provenance statements in the METS Snapshot in which the

change is manifest. For example, METS Snapshot 2 in the above diagram would have a PREMIS event describing that File 1 was deleted from the package and File 4 was added to the package. In most cases, the HandS system can automatically detect when these changes occur and will automatically add the appropriate provenance statements or embellish the technical metadata as required. However, it may not be able to determine why the changes occurred without some sort of intelligent intervention. HandS is able to detect the changes because it has access to the previous Snapshots and can compare the Snapshot of the package as it went into a repository to the package that is retrieved from the repository. This is one of the primary reasons that the METS documents themselves are also placed into a repository along with the other files that are actually part of the package.

The METS profile implementations described above are an integral piece of the HandS architecture, used as framework for generating and maintaining PREMIS-based metadata over time to support long term preservation. The next section looks in more detail at other mechanisms of the HandS tool suite, and illustrates its overall workflow cycle.

3.2.5. HandS Workflow Cycle

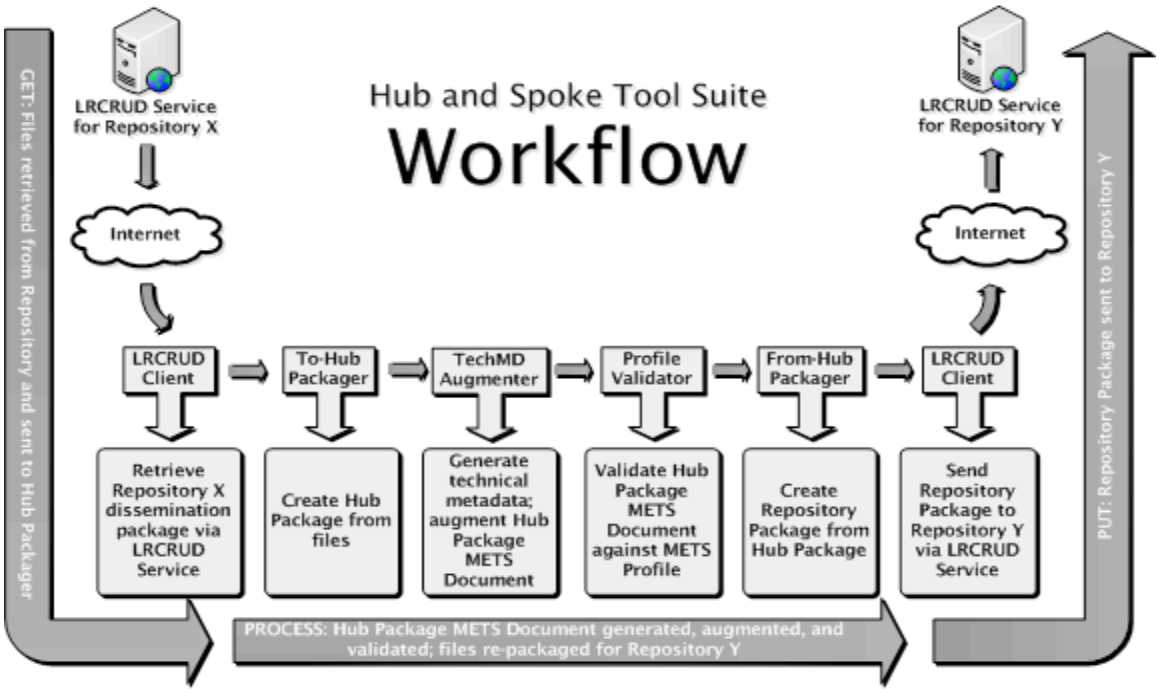


Figure 7: HandS Workflow Cycle

As described in the preceding sections, the HandS Tool Suite provides a framework for sustaining and enriching preservation metadata for digital objects as they are moved into, out of, and between digital repository systems. Digital objects or preservation packages typically refer to a set of files that represents a single intellectual entity, including metadata about the entity or about the files themselves. In the Hub and Spoke workflow cycle (see Figure 3), digital objects are retrieved, converted to a common profile, validated, enriched with metadata, transformed into a repository-compatible form, and ingested into a digital repository.

3.2.5.1. Workflow Overview: GET, PROCESS, PUT

Preservation packages may enter the HandS workflow in various ways: some may come from third-party applications like the OCLC Web Archives Workbench, others may be disseminated from a digital repository like DSpace or EPrints, and some will originate simply as directories of files on a computer file system. In any case, the set of files that make up the preservation package must first be gathered and organized for processing. Objects entering the workflow from a digital repository system must first be fetched from the repository by interacting with its native dissemination

routine, which will vary from repository to repository. This interaction with the repository system is facilitated by our "Lightweight Repository Create, Retrieve, Update, and Delete" Service — affectionately named LRCRUD. LRCRUD is made up of two modules: the LRCRUD Client, which runs on the same machine as the other HandS tools, and the LRCRUD Service, which runs alongside a digital repository system. To retrieve a package from the repository, the LRCRUD Client makes a request to the LRCRUD Service. The LRCRUD Service, in turn, communicates directly with the repository system and retrieves the package via the repository's native dissemination routine. The LRCRUD Service zips up the package and sends it over the network to the Client. Once the package has been received by the LRCRUD Client and verified, its contents are unzipped onto the local file system.

From there, the To-Hub Packager tool converts the digital object into what we call a Hub Package. A Hub Package is made up of the content files that constitute the digital object; METS documents containing descriptive, administrative, and structural metadata about the object at various points in time; and a single Master METS document that comprises chronological and structural information about the other METS documents. The Master METS file will contain a pointer to at least one, but potentially several other ECHO DEP METS documents, each of which serves as a snapshot of the Hub Package at some point in its life cycle. The ECHO DEP METS document is the heart of a Hub Package; it holds together all the files and various metadata that make up the package. When a Hub Package is created, a new ECHO DEP METS document is generated for the package. If the package already contains an older ECHO DEP METS document (generated prior to ingestion into the repository), the new METS document is compared to the older one to discover any changes or damages to the package that might have occurred while in the custody of the repository.

The ECHO DEP METS document is then enriched with technical metadata and validated against the ECHO DEP METS Profile registered with the Library of Congress (Habing, 2005)]. The TechMD Augmentor tool enriches the METS document with format-specific technical metadata found by analyzing each of the package's content files, and converting the result into PREMIS Object metadata. Once the package has been analyzed and enriched, the Profile Validator closely inspects the constituent files that make up the package, both data and metadata, and verifies there are no errors or inconsistencies.

At this point, the Hub Package is ready to be sent on to another repository. But first, it has to be converted into a form compatible with ingestion into the target repository, which again will vary from repository to repository. This final conversion is carried out by a From-Hub Packager tool, built specifically for the target repository. From there, the package is handed off from the LRCRUD Client to the LRCRUD Service for the target repository and ingested.

3.2.5.2. Workflow Example

The workflow cycle might be more easily understood by following an example preservation package as it makes its way through the process. For this example, we will use three small files that make up a single web page: an HTML file, a Cascading Style Sheet (CSS), and a JPEG image. These three files compose a single preservation package, or item, which has been submitted to a DSpace digital repository. Using the HandS tool suite, we will transfer the item from DSpace to an EPrints repository, while generating preservation and technical metadata along the way.

1. Retrieve Repository X dissemination package via LRCRUD

In our example, suppose we have an LRCRUD Service running alongside a DSpace repository on a remote server. The LRCRUD Client application sends a request to the LRCRUD Service to retrieve (GET) a package from the repository. The LRCRUD Service relays the retrieval request to DSpace using the repository's native dissemination method. The output of a repository's dissemination will typically be made up of any number of metadata streams and other supporting artifacts in addition to the item's content files. In the case of DSpace, the package will include a DSpace METS file that encompasses MODS descriptive metadata about the package and PREMIS technical metadata pertaining to each of the constituent bitstreams. In our example, the package returned by DSpace now contains four files: the HTML, CSS, and JPEG files we began with and a DSpace METS file.

The LRCRUD Service receives the DSpace dissemination and packages its contents into a zip archive, which will be transmitted over HTTP to the LRCRUD Client. The LRCRUD Service also calculates file size and a checksum value for the zip file before sending it, and transmits these values as Content-MD5 and Content-Length HTTP header fields along with the package zip file. As the LRCRUD Client receives the package zip file, it too calculates file size and checksum values, which are validated against the HTTP header fields to ensure the package was unharmed during the file transfer. Assuming the values agree, the package is unzipped and saved to disk.

2. Create Hub Package from repository dissemination files

To create a Hub Package from the repository dissemination package, the To-Hub packager needs to produce a new ECHO DEP METS document for the package. The packager begins by searching the retrieved files for any metadata included by the repository. In our example, the packager locates the DSpace METS document and retrieves its MODS descriptive metadata stream. This DSpace MODS metadata will be transformed into Aquifer MODS and inserted into the new ECHO DEP METS document's descriptive metadata section. Other repositories export metadata in different formats (e.g., Dublin Core), but in all cases the package metadata are ultimately transformed to Aquifer MODS by the To-Hub Packager.

The To-Hub Packager then creates an entry in the file section of the ECHO DEP METS document for each of the package's constituent files. In our example, the new ECHO DEP METS document will contain a file element for each of our three content files. The To-Hub Packager will also create in the new ECHO DEP METS document three PREMIS technical metadata objects to correspond to three file elements. Each will contain basic technical metadata about one of the files, including checksum values, file size, and MIME-type. Finally, any leftover descriptive and technical metadata elements from the DSpace METS document are inserted into the ECHO DEP METS document as alternate metadata so that it is never lost.

If the package contains older ECHO DEP METS documents (because it had been packaged by HandS before entering the repository), the most recent ECHO DEP METS document is compared to the just-generated ECHO DEP METS document to expose any changes the package may have undergone since it was last analyzed. These data are recorded in the new ECHO DEP METS document's provenance metadata as PREMIS events. If the package contains a Master METS document, a pointer to the new METS document is created and designated as the most-current ECHO DEP METS document for the package. If no Master METS document can be found, the To-Hub Packager creates one from scratch. Once the new METS document has been created and the Master METS document is updated, Hub Package creation is complete. Our example Hub Package now consists of a Master METS document, which points to a single ECHO DEP METS document. This ECHO DEP METS document contains descriptive metadata about the package; technical metadata about each of the three content files, along with pointers to those files and the technical metadata left by DSpace; and provenance metadata documenting the package's export from the repository and its Hub Package transformation.

3. Generate technical metadata; augment Hub Package METS Document
Using tools from the JSTOR/Harvard Object Validation Environment (JHOVE), the HandS TechMD Augmenter module analyzes each of the Hub Package's content files and generates format-specific, technical metadata for each. The JHOVE-generated metadata is transformed using format-specific XSLT stylesheets, and inserted into the technical metadata section of the ECHO DEP METS document as technical metadata. Any inconsistencies between the technical metadata currently held in the METS document and those generated by JHOVE are recorded in the provenance section of the ECHO DEP METS document as PREMIS validation events. The technical metadata stored in the ECHO DEP METS document is formatted in compliance with the following metadata preservation standards: AudioMD for audio files; TextMD for text, XML, and HTML; and MIX for images.

In our example the HTML, CSS, and JPEG files will each be analyzed by JHOVE. The JHOVE output for both the HTML and CSS files will be formatted as TextMD, and the output for the JPEG image will be formatted as MIX. Each will be inserted into the

ECHO DEP METS document in a technical metadata element corresponding to the appropriate file element. The JHOVE analysis itself is also documented and recorded in the ECHO DEP METS document as a validation event. One of the limitations of JHOVE is its small number of supported media types. In its current release JHOVE offers no support for closed formats such as Microsoft Office files. Another drawback of using JHOVE is that it only reports the MIME-type correctly for HTML or XML files if they are well formed; otherwise it reports them as plain text, causing discrepancies within the ECHO DEP METS document and validation warnings. Nevertheless, we found JHOVE to be a useful tool for analyzing files and generating technical metadata. For more on JHOVE visit http://hul.harvard.edu/jhove/.

4. Validate Hub Package METS Document against METS Profile

The Profile Validator examines the current ECHO DEP METS document for the Hub Package against the requirements of our METS profiles currently registered with the Library of Congress (Habing 2005, 2006).

Key validation points include checking to make sure that the primary descriptive metadata element contains a MODS object that conforms to the Aquifer MODS profile; that every file referenced by the file section has associated technical metadata PREMIS objects; and that all provenance metadata associated with a file contain valid PREMIS event elements. The Profile Validator also checks that the package content files referenced by the ECHO DEP METS document are accounted for, and that their checksum, file-size, and mime-type values are correct.

Our example ECHO DEP METS document passes validation for the following reasons: it contains valid Aquifer MODS in its primary descriptive metadata element; each of its file elements reference technical metadata elements containing valid and complete PREMIS object metadata; and it conforms structurally to our METS profile requirements. Once the validation has completed, the validation event itself is documented and recorded in the ECHO DEP METS as a PREMIS validation event.

5. Create Repository Package from Hub Package

Before a repository can accept a package for submission, it must first receive a description of the package's contents. The From-Hub Packager module uses descriptive metadata extracted from the Hub Package ECHO DEP METS document to generate the repository-specific metadata need for package submission. This process usually involves transforming the Aquifer MODS metadata found in the ECHO DEP METS document into a metadata format required for repository submission, and will vary from repository to repository.

In our example, we are sending the package to an EPrints repository, which means the packager will generate an EPrints-specific metadata file from the Aquifer MODS stream. The transformation event is recorded in the ECHO DEP METS document as a

PREMIS metadata-transformation event, and the newly-generated metadata is added to the METS document as alternate descriptive metadata. A Repository Package zip file is then created, consisting of the Master METS document, all the subordinate METS snapshot documents and the content data files, as well as any repository-specific metadata files.

6. Send Repository Package to Repository Y via LRCRUD
At this last step in our example, we have an EPrints-specific LRCURD Service running on a remote server with an EPrints repository. The LRCRUD Client sends a request to the LRCRUD Service to create (POST) a new package. The LRCRUD Service relays the create request to the repository and, using the repository's native methods, creates an empty record. The LRCRUD Service receives a new location identifier, or handle, corresponding to the newly created location in the repository, which it sends back to the LRCRUD Client. This location identifier is inserted into the package's ECHO DEP METS document as the primary ID for the METS document.

The LRCRUD Client then sends a request to the LRCRUD Service to update (PUT) the new package at that location. The LRCRUD Client calculates file size and checksum values for the package zip file before sending it to the Service, and it transmits these values as Content-MD5 and Content-Length HTTP header fields along with the package. As the LRCRUD Service receives the package zip file from the Client, it calculates its own file size and checksum values and validates them against the HTTP header fields to ensure the package was unharmed during the file transfer. Once the LRCRUD Service has validated the file transfer, it unzips the package and ingests each of its contents—including the package METS files—into the repository using the repository's native submission routine. The repository-specific descriptive metadata that was generated in Step 5 above is submitted to the repository as well. Once the package has been fully ingested, the LRCRUD Service returns an update response message to the LRCRUD Client confirming the successful submission, or an error if the submission failed.

Some repositories allow for certain bitstreams to be given privileged status. In such cases the Master METS and ECHO DEP METS files may receive special status; but in all cases the METS files are preserved along with the other package content files and are treated as first class objects with regard to the repository. That way, when the package is retrieved from the repository, all the metadata pertaining to the state of the package before it was submitted to the repository is not lost.

3.2.5.3. Workflow Recap

Through the workflow process described above, HandS provides tools to facilitate moving digital objects between multiple repositories while generating and maintaining important PREMIS-based technical and provenance preservation metadata. Digital objects are retrieved, converted to a common profile, validated,

enriched with metadata, transformed into a repository-compatible form, and ingested into the target repository.

3.2.6. HandS Technical Implementation

The key technical components of the HandS implementation are the Hub and Spoke METS profile Java classes, providing an extensible Java API to our METS profile with Apache XMLBeans; the To- and From-Hub Packager modules, facilitating interoperability through pluggable interfaces; and the Lightweight Repository CRUD Service (LRCRUD), supporting the dissemination and submission of objects by defining a protocol for transmitting digital objects to and from repository systems over HTTP.

3.2.6.1. Hub and Spoke METS Profile API

The core of the HandS Tool Suite is our METS Profile API, a Java code representation of a METS XML document compiled from our METS profile. The bulk of our METS classes were created with Apache XMLBeans (http://xmlbeans.apache.org/), a tool for generating Java classes from XML schema files (XSD files). With XMLBeans, we are able to compile XML schema documents to produce a Java code structure, allowing us to work with XML data through our own Java classes and methods. To create our METS profile API, we combine methods from XMLBeans-generated classes from the METS, MODS, and PREMIS schemas, along with format-specific preservation metadata schemas like MIX, TextMD, and AudioMD. We also layer custom-built convenience methods on top of the XMLBeans-generated methods to facilitate additional manipulation of the METS document in a fashion unique to our METS profile.

A new HandS Profile Java object can be created from scratch given a set of content files and accompanying metadata, or by instantiating an existing XML METS document that conforms to our profile. Once instantiated, the underlying METS document object can be operated upon programmatically through API calls. In working with the API, we are assured that any manipulation of the METS document will always be consistent with our METS profile. For instance, to add a new file to the preservation package, a call is made to the to the addFile() method, which in turn triggers calls to other methods that ensure the METS object remains consistent with our profile — such as adding a new PREMIS Object techMD section associated with the new file, and generating checksum, MIME-type, and file size values. At any time the METS object can be validated against our profile, or re-serialized as XML and saved to the file system.

3.2.6.2. To- and From-Hub Packagers

To facilitate repository interoperability, the HandS Tool Suite includes a set of packager classes for transforming a collection of preservation items into a Hub Package, and for transforming a Hub Package into a form required for submission

into a given repository. For items entering the Hub and Spoke from a digital repository, the repository-specific To-Hub packager takes the native repository dissemination, unpacks it, and instantiates a new HandS METS Profile object from its contents. Going the other way, a repository-specific From-Hub packager prepares a Hub Package for submission into the particular repository.

Currently, we have To-Hub packagers for processing items coming from DSpace, EPrints, OCLC's Web Archives Workbench, or from a directory of files. Our current list of From-Hub packager targets includes DSpace, EPrints, and the Library of Congress archive standard Bagit. To- and From-Hub packager modules for the Fedora repository are currently in development. We have employed a pluggable architecture for creating packager modules. Base To- and From-Hub classes are implemented in Java as abstract classes with the intention that they will be overridden and extended by other programmers needing to tailor the HandS Tools to their specific repository or archiving standard. This modular architecture allows other developers to create packager plug-ins for their own repository systems without having to recompile or re-factor the existing HandS codebase.

3.2.6.3. Lightweight Repository CRUD Service (LRCRUD)

The Lightweight Repository CRUD service specification defines dissemination and submission web-service interfaces to digital repository systems for use with the ECHO DEP Hub and Spoke Tool Suite. The LRCRUD specification defines a protocol for transmitting digital objects to and from repository systems over HTTP. It enables users to obtain objects in a format expected by the HandS processing scripts and supplies digital objects to repositories in a format expected by their native ingestion mechanisms. The specification is implementation-agnostic: it simply defines the parameters and responses required to enable a service implementation to communicate with the LRCRUD client application. This allows LRCRUD implementers to choose the most appropriate environment and programming languages for interacting with their chosen repository. The HandS Tool Suite currently has LRCRUD implementations for DSpace, EPrints, and Fedora.

The LRCRUD Service follows Representational State Transfer (REST) conventions. It exposes CRUD actions on repository content over the HTTP protocol. As mentioned above, CRUD is an acronym for Create Retrieve Update and Delete – the basic operations that applications should implement when acting upon persistent storage like relational database management systems, file systems, and the like. The LRCRUD client communicates with the LRCRUD service via HTTP methods, status codes, and headers. The list below shows how the CRUD actions are mapped to the HTTP methods:

- Create == POST
- Retrieve == GET
- Update == PUT

- Delete == DELETE

In most cases the LRCRUD service will reside on the same host as the repository it serves so that it has access to the repository's API.

LRCRUD is essentially a "dumb" packager; it is simply a way to supply files to the remote repository in any format/configuration that it can natively ingest. In this it is similar to protocols like the Simple Web Service Offering Repository Deposit, or SWORD (Allinson, François, & Lewis, 2008), which are being adopted by repositories -- and which may make the submission function of LRCRUD ultimately unnecessary.

It may be beneficial to present some descriptive step-by-step examples in order to clarify the functions of the LRCURD components within the Hub and Spoke Tool Suite. These examples describe in detail the interactions between the client and the service.

3.2.6.4. LRCRUD Functions -- Examples

3.2.6.4.1. Dissemination

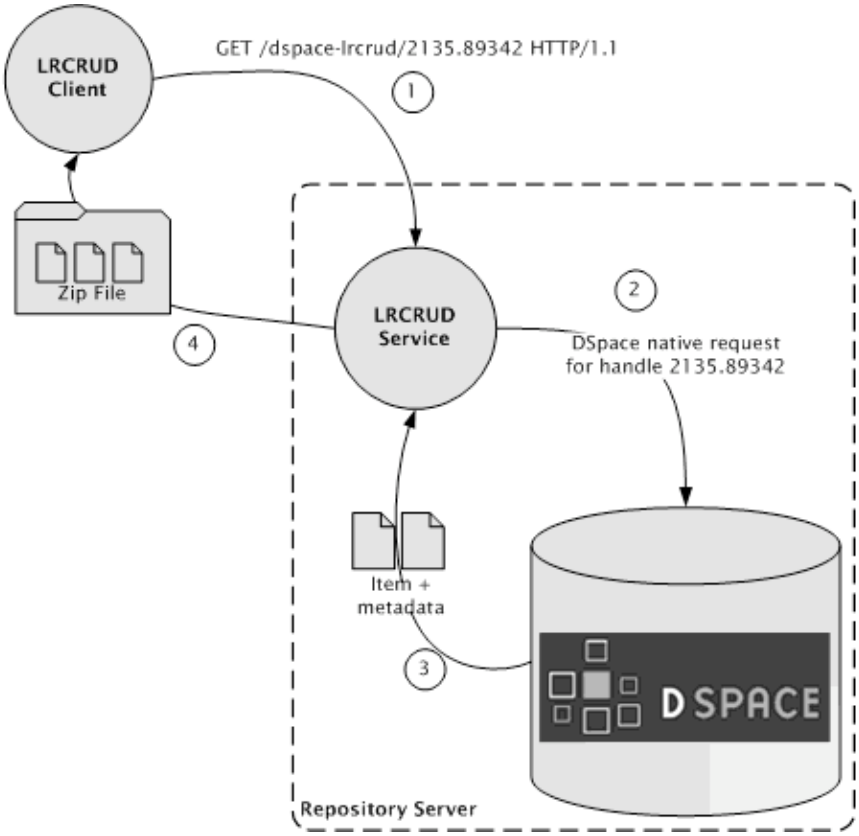


Figure 8: LRCRUD Dissemination

Dissemination (see Figure 8) is the act of retrieving an item from a repository, whereby the item is defined as an intellectual entity comprising any number of content streams, metadata streams, and other supporting artifacts. Items disseminated from a repository using the LRCRUD service are most likely bound for processing and transformation by the HandS Tool Suite to-hub packager. The packager creates METS files conformant to the HandS profile, extracts and augments technical metadata, and records provenance information.

Described below are the four major steps in negotiating dissemination:

1. The LRCRUD client submits an HTTP GET request to the LRCRUD service. The GET request provides the ID of the item desired via the LRCRUD service URL syntax.
2. The service calls the repository's native dissemination routine for the ID indicated.
3. The service receives the output from the dissemination and adds the entire content into a zip-file.
4. The service returns the zip-file containing the "package" to the client.

3.2.6.4.2. Submission

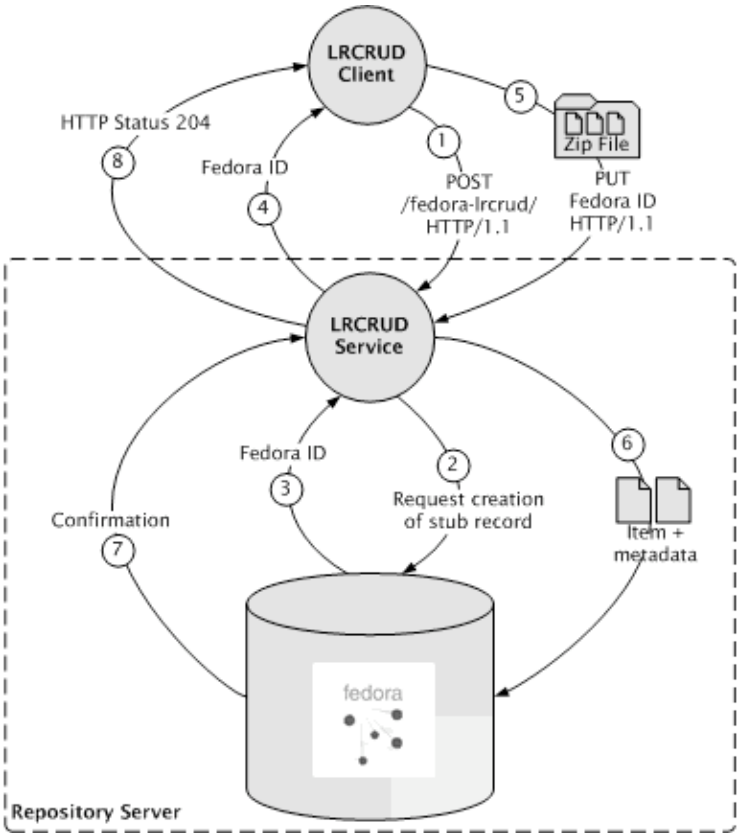


Figure 9: LRCRUD Submission

Submission (see Figure 9) is the act of either 1) adding an item to a repository for the first time; or 2) updating an item already in the repository. Described below is the process of adding an item to a repository for the first time. This is a two-stage process; the first stage reserves an identifier in the system, while the second actually places content in the repository.

Stage 1 - Create stub record to reserve an identifier: It is critical to note that the package itself is not uploaded as part of the POST request; rather, the POST request creates only a stub or placeholder record. The reason that the actual package is not uploaded as part of the POST is that the identifier assigned to the package by the repository needs to be embedded in the METS file which is part of the package. The typical sequence of operations to ingest a new package is to use POST to create a new placeholder record and get the identifier for that record. That identifier is then used to update provenance and other metadata that is part of the package, and then the placeholder record is updated or overwritten with the actual package using the PUT action.

The major steps in this process are:

1. The LRCRUD client issues a POST request to the LRCRUD service specifying the ID of "where" to create the record (e.g. in a specific collection) if needed.
2. The service calls the repository's native item or ID creation routine.
3. The repository supplies the service with the ID for the newly-created record.
4. The service responds to the client with an HTTP 201 "Created" message and returns the ID in the Location: header.

Stage 2 – Upload and ingest the item: In this stage, the item is uploaded and placed in the repository. This is the exact process for updating an existing item:

1. The LRCRUD client issues a PUT request to the LRCRUD service to replace the package identified by the supplied URI. The entity body of the request will contain a zip-file containing the "package" to be ingested.
2. The service unpacks the files and calls the repository's native ingestion routine.
3. The service responds to the client with an HTTP 204 "No Content" message indicating that the request was successful.

3.2.7. Lessons Learned

Below, in no particular order, are several key lessons learned during the course of developing the HandS architecture.

3.2.7.1. Merging metadata is potentially risky.

After expending much effort exploring how we might merge different versions of metadata files so as to maintain a single master metadata file, we reached the conclusion that this was very difficult problem, and potentially dangerous in terms of data loss. This realization led us to our current architecture, which skirts the issue of merging metadata into a single file by maintaining multiple Snapshot METS files all referenced from a common Master METS file.

3.2.7.2. METS supports multiple metadata formats well.

Combining PREMIS, MODS, and other XML-based technical metadata formats into a single METS document worked well for this particular project. The general structure of METS seemed to lend itself to constructing preservation packages. Our conceptual model, which was directly influenced by the METS and PREMIS structures, consisted at a high level of the intellectual entity having one or more representations. These representations and all their component parts were the primary focus of the preservation efforts. The METS file itself is treated as the abstract parent representation of the intellectual entity. However, there are also one or more concrete representations consisting of each structMap within the METS file. These representations consist of the relationships embodied in the structMap (and possibly the related structLink sections); the files and bitstreams referenced from the structMap; and the associated descriptive metadata (dmdSec), which could be referenced via the structMap or via individual files or bitstreams. All remaining parts of the METS document, primarily the header (metsHdr) and administrative metadata (amdSec) sections, are not considered part of the intellectual entity's representations but are, instead, metadata about these representations -- mostly concerned with preservation and thus having a strong focus on technical and provenance metadata. There were pragmatic challenges in getting these disparate metadata standards to work together, however, and the next paragraph conveys one such example.

3.2.7.3. Implementing PREMIS in METS requires high-level structural decisions.

Embedding PREMIS metadata within a METS package was not an intuitive undertaking. There were several reasons for this. Among these were various overlaps in the metadata fields supported by each standard. When faced with these overlaps our general approach was to provide the metadata in both places. Although this approach introduced duplication and the opportunity for inconsistencies into the metadata, we feel the added flexibility in processing

compensated for these shortcomings. Moreover, the HandS tool suite validation steps ensured that these types of inconsistencies were not present. Decisions were also required as to where to embed the PREMIS entities within the METS file. While these entities are clearly all administrative metadata, they do not always fit neatly within one of the four subgroups, techMD, digiprovMD, sourceMD, or rightsMD, provided by METS. Refer to the ECHO DEP METS profiles (Habing, 2005) for details. Project staff also participated in a working group chaired by Rebecca Guenther at the Library of Congress to address this issue. The working group produced a report, *Guidelines for using PREMIS with METS for exchange* (Guenther, 2008).

3.2.8. Next Steps: the Hub and Spoke

Development of the Hub and Spoke tool suite is ongoing. The latest versions of the source code can be downloaded from the project's SourceForge web site: http://sourceforge.net/projects/echodep/. Recent developments include the addition of a From-Spoke for the BagIt specification (Boyko, Kunze, Littman, & Madden, 2008) and modifications to support version 1.5 of DSpace. Work is continuing apace on both From- and To-Spokes for the Fedora repository with particular attention being paid to how our METS profile might be accurately mapped to a Fedora content model, reducing the need for potentially lossy mappings as have thus far been required for other repository systems. The project is also looking at other potential repositories, such as LOCKSS or CONTENTdm, for Spoke development. In addition to developing new Spokes, we are also monitoring developments with the next version of JHOVE, as well as with the Global Digital Format Registry (GDFR), to explore how these tools might be used to enhance the format-specific technical metadata we are currently generating for different file types.

3.2.8.1. Supporting Preservation Now and in the Future

The Hub and Spoke (HandS) framework enhances the interoperability and preservation features of existing open-source repository systems. It provides a suite of tools to facilitate moving digital objects between repositories more easily while supporting the collection of technical and provenance information crucial for long-term preservation. It is intended to support curators' efforts today to manage content in multiple repository systems and to preserve valuable preservation data in accordance with emerging digital preservation standards.

In the long term, however, we see the need for the next generation of digital repositories to do more in order to support our ability to preserve the meaning of the digital objects maintained in repositories. Current repository systems preserve the structures of digital objects, from which meaning or semantics must be inferred. Learning from real-world data migration examples from the HandS efforts, GSLIS and NCSA researchers are working to model how semantic inference capability may help next-generation archives preserve the meaning (not just the structures) of

digital objects and head off longer-term preservation risks. Specifically, we are developing automated reasoning techniques targeted at identifying, and eventually correcting, problematic metadata descriptions. This work is profiled separately in the next section (Section 4).

3.2.9. Conclusion

With digital preservation still in its infancy, many changes to emerging standards, strategies, and methodologies can be expected in the coming years. The Hub and Spoke framework provides a model that attempts to incorporate current technologies and best practices from the field to support digital preservation in current repository environments. It implements METS and PREMIS to provide a standards-based method for packaging content that allows digital objects to be moved between repositories more easily while supporting the collection of technical and provenance information crucial for long-term preservation. HandS is intended to help curators of digital objects today by providing improved support for preservation and interoperability to existing repository systems. Ultimately, in order to meaningfully preserve our digital content over time, we will need the next generation of preservation tools to support automatic inference of meaning, or semantics, from changed—and thus potentially ambiguous—information structures.

4. Preserving Meaning, Not Just Objects: Semantics and Digital Preservation

A key goal of the ECHO DEPository project is to investigate both practical solutions for supporting digital preservation activities today, and the more fundamental research questions underlying the development of the next-generation of digital preservation systems. Earlier in this report, we reviewed two areas of activity that aim to support on-the-ground preservation efforts in existing technical and organizational environments: the Web Archives Workbench, a suite of tools to help curators collect and manage web-based digital resources; and the HandS tools suite, which aims to enhance existing repositories' support for interoperability and emerging preservation standards.

In the longer term, however, we recognize that successful digital preservation activities will require a more precise and complete account of the meaning of relationships within and among digital objects. This section describes project efforts to identify the core underlying semantic issues affecting long-term digital preservation, and to model how semantic inference may help next-generation archives head off long-tern preservation risks.

4.1. Introduction: The Need for a Semantics of Preservation Approach

4.1.1. The Preservation Semantics Problem

Like any information management activity, digital preservation efforts are guided by human understanding. Decisions about documenting a file format, emulating an environment, or migrating from one system to another are made with an understanding of how levels of digital expression cascade and interrelate: voltage, bit, octet, pointer, integer, grapheme, pixel, polygon, color, pitch, text string, tree, image, tuple, file, and so on. The complexity of these relationships poses few serious problems for human beings -- in fact, the problems lie precisely in the ease with which our minds interpret those relationships. Long-term preservation is distributed not only over time but also across the responsibilities of many different people. It is directed at collections much too large to allow thoughtful attention to individual resources. We must therefore build into our tools a more careful and precise encoding of the knowledge that guides our effortless mental deductions. The preservation hazards that result from current descriptive practice and our experiments with automated tools to ameliorate those risks are described in the sections that follow.

4.1.2. Our Goal

Our goal is to understand better the semantic problems arising in digital preservation, and how we might apply that understanding to the development of resources and tools. Specifically, we are experimenting with automated inferences about entities, their properties, the relationships within and between them, and how these facts are expressed in metadata descriptions. Enriching that metadata with new deduced assertions is one step in heading off digital preservation risks. We are working toward a deductive system for reasoning about anomalous or incomplete metadata. The aim is not to automatically deduce all missing information or to correct malformed records, but to call human attention to descriptions that are problematic or suspicious.

Our work begins with an analysis of the kinds of semantic problems posed by current descriptive practice and metadata schemas, informed by analyses of real-world data migration examples. We have applied the understanding gained in this analysis to the development of a draft metadata ontology (discussed in Section C), which moves us toward a more formal understanding of how descriptive information about archived digital resources is structured. This metadata ontology is key to a proof-of-concept experimental system composed of the RDF repository Tupelo and the BECHAMEL reasoning software.

4.2. The Problems: Understanding Semantic Preservation

4.2.1. Problems Posed by Descriptive Practice and Structures

In many preservation efforts metadata description may seem straightforward, but crucial information -– including facts that seem obvious at first glance -- is left unstated, and must be inferred by human readers. (An example is provided next.) As discussed above, this situation may not be risky when people are available to reason about individual records, but a human-based manual approach does not scale over large collection sizes or over time. The sheer volume of digital information means we increasingly rely on automated machine processing of records. But software tools execute transactions using only knowledge that has been explicitly represented for them. Our aim therefore is to make those unstated facts available in a form that software can use.

This work begins with an investigation of the kinds of semantic problems posed by current information structures and implementations. These problems break down into three basic categories:

- Semantic problems relating to descriptive practice
- Semantic problems relating to encoding standards

- Semantic problems relating to metadata schema design

4.2.1.1. Semantic problems relating to descriptive practice.

Some of the problems we face are a result of how resources are described using metadata, while other problems arise by way of how those descriptions are expressed, and what happens to them over time as they are migrated from one system to another. One semantic problem of particular interest to us is what Renear et al (2002) describe as "ontological variation in reference." Essentially, metadata can fail to make critical distinctions in what, precisely, it is describing. The problem is illustrated in the metadata example below, which shows properties asserted at a number of different levels of abstraction.

```
1.  <rdf:Description rdf:about="info:fedora/changeme:97">
2.  <dc:subject>Counties: Peoria</dc:subject>
3.  <j.1:disseminates rdf:resource="info:fedora/changeme:97/AERIAL.2135.85771.1"/>
4.  <dc:publisher>U.S. Dept. of Agriculture, Agricultural Adjustment Agency, North Central
5.  Division, Washington, D.C.</dc:publisher>
6.  <dc:creator>Aerial Photographs</dc:creator>
7.  <j.1:hasDatastream rdf:resource="info:fedora/changeme:97/AERIAL.2135.85771.2"/>
8.  <dc:title>Peoria County, Illinois</dc:title>
9.  <dc:type>image</dc:type>
10. <dc:creator>United States. Agricultural Adjustment Agency.</dc:creator>
11. <j.1:hasDatastream rdf:resource="info:fedora/changeme:97/DC"/>
12. <dc:format>image/jpeg</dc:format>
13. <dc:contributor>Scanning, indexing, and description sponsored by the Illinois State Library
14. and the University of Illinois at Urbana-Champaign Library. Geo-referencing sponsored and
15. performed by the Geographic Modeling Systems Laboratory, University of Illinois at
16. Urbana-Champaign.</dc:contributor>
17. <j.0:state rdf:resource="info:fedora/fedora-system:def/model#Active"/>
18. <j.1:disseminates rdf:resource="info:fedora/changeme:97/DC"/>
19. <j.0:createdDate rdf:datatype="http://www.w3.org/2001/
20. XMLSchema#dateTime">2006-11-17
21. T20:40:24.149</j.0:createdDate>
22. <j.1:hasDatastream rdf:resource="info:fedora/changeme:97/AERIAL.2135.85771.1"/>
23. <dc:subject>Railroads: Atchison, Topeka, & Santa Fe</dc:subject>
24. <dc:rights>Copyright 1997-2003 the University of Illinois Board of
25. Trustees. Images cannot be re-distributed in this form for any commercial
26. purpose.</dc:rights>
27. <dc:subject>Highways: IL 30</dc:subject>
28. <dc:date>Scanned and Processed: 1998-06-01</dc:date>
29. <dc:language>en_US</dc:language>
```

Figure 10: Example of Multiple Levels of Abstraction in Metadata Description

We see in this example properties of the image itself (like its title and subject matter in lines 8 and 23) described alongside properties of the file which encodes the image (its MIME classification in lines 2 and 12), properties of the metadata description (its creation date in line 28), and properties of the repository software object that

expresses the metadata description (e.g., that it disseminates resources, and has particular data streams associated with it; lines 1, 3, 7, 11, 18, and 22).

The main preservation risk proceeding from this mixing of levels is the inability to distinguish, without semantic information absent in the description, the level at which a particular property applies. For example, what is it exactly that has a MIME classification image/jpeg? Is it the Fedora record or is it one or both of the datastreams? A human reader can easily resolve that kind of ambiguity without conscious effort, but preservation transactions (such as migration), which are typically executed through software, cannot. The preservation aim here is presumably to preserve access to the image. That aim may or may not depend on preserving the jpeg file expressing the image, and preserving the Fedora object that expresses the metadata is almost certainly not a requirement.

This example therefore illustrates the problem of mixed levels of description. We need to clarify and enrich metadata descriptions by linking their assertions explicitly to the appropriate entities, or else draw the attention of human analysts to records that cannot be disambiguated automatically.

4.2.1.2. Semantic problems relating to encoding standards.

In addition to problems of descriptive practice, we face semantic problems stemming from limitations of the encoding technologies in which metadata descriptions are expressed. These problems generally fall into one of the following two categories:

4.2.1.2.1. Syntactic overloading in conventional markup.

Familiar encoding technologies for metadata descriptions, such as those based on XML, work well most of the time, but they have certain fundamental problems, such as those as stemming from the use of multiple competing semantic relationships and of unstructured data. Specifically:

- **Competing semantic relationships:** Preservation metadata formats typically overload a simple syntax with multiple competing semantic interpretations. Typical examples include XML applications where a small number of syntactic relationships (e.g., the parent/child relationship between elements) represent any number of semantic relationships (whole/part, property name/value, etc.) that are context dependent. Often a precise interpretation of these semantics can be found only in the execution of application software that consumes the file -- and, presumably, in the mind of the programmer who wrote the application.
- **Unstructured data:** The information in resource descriptions may only be incompletely available for machine processing and verification. Crucial

contextual data may exist only as natural language annotations or as unstructured information in the content of metadata fields.

The metadata example presented in Figure 10 does not exhibit problems of syntactic overloading, because it conforms to a standard serialization of the RDF abstract model in which properties and relationships are explicitly identified. But the second problem is evident in how much information in this description is expressed in natural language text and annotations. (Note, for example, the *dc:date* element (line 28) in which the documented event (scanning and processing) is prepended to the date string.)

4.2.1.2.2. Problems with object models.

Other potential semantic problems stemming from limitations of metadata encoding technologies concern the object models of repository systems themselves. Modeling decisions in repository design can create descriptive artifacts that leave their mark even after record migration. For example, a repository may mingle information about repository objects with the information that the repository objects are meant to preserve, creating problems when those records are further processed and contextual information is no longer available to help interpret the records and make further preservation decisions. A good illustration of this issue can be seen by revisiting the metadata description in Figure 10, which was serialized from triples that were extracted from the RDF database backing a Fedora repository installation. In Figure 10, notice that in RDF terms this entire metadata description is "about" an object identified as *info:fedora/changeme:97* (line 1). This repository software object is the only resource identified by an *rdf:type* arc, and is therefore the only entity with an object class identification. Barring any explicit type identification in a resource description, Fedora objects seem to be the only kind of thing that the Fedora repository knows about. Expressed in that form, we cannot preserve any information except Fedora records, and those records assert no explicit preservation targets. A system like Fedora can preserve objects within the context of its own transactions, but the implicit knowledge directing such operations depends on the interpretation of programmers, with all the problems discussed so far.

On the other hand, it is not a design flaw of Fedora that its metadata record is centered internally on the Fedora digital object. Preservation ontology is properly a matter of descriptive practice, not software engineering. In fairness, our metadata example comes from a migration scenario in which RDF triples are extracted from Fedora's RDF store directly, rather than through a conventional export process. But this example serves to remind us that object modeling in a system such as Fedora plays the same role to the same ends as with other kinds of software: efficient source code management by and for the system developers. Object modeling decisions are not intended and cannot be expected to address the weaknesses of

resource analysis and description. For long-term preservation, therefore, it is important to reduce ambiguous or implicit semantics in repository object models. That can mean either modifying those models or, as we have attempted, providing tools and techniques for migrating from repository object models to models that include better representations of preservation targets.

4.2.1.3. Semantic problems with published metadata schemas.

Finally, in addition to semantic problems relating to descriptive practice to encoding standards, we see semantic problems stemming from limitations of published metadata schemas themselves. Published schemas formalize element sets on which the property/value ascriptions are based. Each of these metadata schemes not only expresses its unique view of the universe but is itself grounded in basic ontological assumptions. A variety of ambiguities can still arise, as illustrated below, drawing again on our running example from Figure 10.

We need to begin by understanding the logical parts of the metadata record and their relationships to one another. A metadata record describes some entity -- an instance of a class like the class of books, images, or audio recordings. Metadata descriptions list properties of that entity, each of which has a value. For instance the "author" property of the book might take as its value the name of the author. Membership in a class requires that the instance respect defined class constraints (movies, for example, have running times, but books do not).

Consider the metadata statement *dc:type>image</dc:type>* (line 9) from our original example in Figure 10. We easily recognize that the word "image" points us to an instance of a class, just as the name of an author points us to a particular person. A human reader would never conclude that a book was authored by a name or by a string expressing a name. Similarly, the word "image" licenses our inference that the property value for *dc:type* is a class of entities in the world rather than, for example, a quantity (such as 14 centimeters) or a quality (like monochrome). In this case we are cued to the existence of not just any entity, but to the very target of our preservation efforts -- something much more important to us in the long run than the digital file or the bit sequence that only expresses this image contingently.

Computer software cannot make those kinds of meaningful distinctions without help. One kind of help would be a constraint on the range of allowable property values, but the Dublin Core element schema enforces no such constraint: *dc:type* can take any value that indicates the "nature or genre of the resource." (DCMI Namespace for the Dublin Core Metadata Element Set, Version 1.1, 2008)

A second kind of help would be a value string that, through its machine-readable structure or notation, indentifies a class. In an RDF expression this would be a URI linked by an *rdf:type* property to some class declaration. The DCMI Type Vocabulary

has this structure (http://dublincore.org/2008/01/14/dctype.rdf), and interestingly, the scope note for *dc:type* in the DC Elements RDF schema recommends the use of that vocabulary (http://dublincore.org/2008/01/14/dcelements.rdf). Had the author of our metadata description used the URI *dcmitype:Image*, instead of the word "image," we would be one step closer to identifying the abstract image as an entity. The word "image," although it contains the same sequence of letters, is not linked in a standardized way to the declaration of a class. Assigning a DCMI type resource to the *dc:type* element simplifies the inference that an image exists, and that one or more of the metadata statements in that description are ascribing properties of an image – semantically, a significant step. But as our running example stands, the schema's flexibility invites ambiguity, and additional information is necessary to connect the literal value "image," with a formalized class such as *dcmitype:Image*.

4.2.2. Understanding the Semantic Preservation Problem: Summary

We have seen in this section that descriptive practice, encoding standards, and published specifications may all complicate digital object preservation. Imprecise resource descriptions can make it impossible to determine the level at which a particular property applies. The flexibility offered by encoding standards brings risks as well as benefits. We've also seen how object modeling decisions and semantically underspecified metadata schemas can lead to incorrect or ambiguous usage. In the next section, we move from understanding the core semantic problems associated with descriptive practice and structures to looking at the resources and tools being developed by the ECHO DEPository project to identify semantic ambiguity in real-world metadata descriptions, and highlight potential preservation risks.

4.3. Toward More Capable Archives and Repositories

4.3.1. Recap: The need for automated inference capability

Digital resource preservation efforts are distributed not only over time but typically across the responsibilities of people who may never consult with one another. Transactions like migration between systems are executed over large collections where close attention to individual records is too expensive, but where correct treatment of a resource often depends on knowledge that is incompletely or imprecisely represented in preservation metadata. Such ambiguities present few problems for human beings: our flexible minds make correct inferences without conscious effort. But the data to support those inferences are not expressed in a form that can guide the execution of our programs and utilities. We therefore need tools and methods that support the discovery and correction of preservation risks.

The next section describes our experiments in developing these methods and tools. Specifically, we look at our development of an ontology of metadata descriptions, the Resource Description Vocabulary, and how this may be applied to identify semantic ambiguity in metadata descriptions using the reasoning tool BECHAMEL.

4.3.2. BECHAMEL and Building a Metadata Ontology

BECHAMEL is a tool for expressing and testing semantic models of digital resources. (Dubin at al, 2003) It has been developed by researchers at the University of Illinois, the World Wide Web Consortium, and the University of Bergen. A BECHAMEL application can, for example, translate the bibliographic metadata for a journal article from one standard format into another by constructing a model of the author's affiliation with an institution. (Renear & Dubin, 2003) In our recent and current experiments the input to BECHAMEL are metadata descriptions retrieved from an RDF repository (Tupelo), together with schemas defined in the OWL Web Ontology Language (OWL, 2004). New facts deduced from those inputs are added back to the repository as annotations to the description. A technical overview of this approach is presented in the following sections.

4.3.3. Overcoming Semantic Problems in Metadata Encoding: A Resource and Description Vocabulary

Our aim is to enrich metadata with new assertions inferred from existing resource descriptions. Toward that aim we have identified classes, properties, and relationships for overcoming encoding problems, and we have expressed these in a schema. This vocabulary does not represent classes or properties for specific types of resources. Instead, it offers an ontology of metadata descriptions themselves. Simply stated, the vocabulary includes terms that can be used to describe records, metadata descriptions, and relationships between them and preservation targets. (The Resource and Description Vocabulary is provided in **Appendix 6.7**.)

More specifically, the vocabulary is divided into the following sections:

- W3C standard classes and properties
    - These include classes and properties such as *rdfs:Resource, rdf:Statement,* and *owl:ObjectProperty*.
- Alternate reification classes
    - Conventional use of the RDF reification vocabulary is based on an understanding that triples stand in a type/instance relationship with "tokens" appearing in RDF documents (RDF Semantics, 2004). But this interpretation, intended to support provenance documentation, presents puzzles for understanding how a serialized expression can stand in direct relationships with resources referred to by an abstract triple. (For those analysts who may be concerned with abusing the official account of RDF reification, the vocabulary includes separate

classes for generalized statements, RDF statements, and abstract triples.)

- Indication relationships
  - This section includes a group of hierarchically organized relationships, based on recommendations in Piotr Kaminiski's 2002 thesis. The relationships include indication, representation, denotation, identification, description, depiction, ascription, expression, and encoding.
- Classes based on the DCMI Abstract Model
  - In the Dublin Core Abstract Model, the term "metadata element" is used synonymously with the term "property." But our classes, though based on that model, represent metadata elements as specialized names of properties, rather than as properties themselves. Classes in this section include metadata element, metadata element set, metadata statement, and metadata description.
- Markup structures
  - A third alternative to reifying RDF statements under the official W3 interpretation, or through use of alternate classes, is to reify the notation expressing the RDF. This section of the vocabulary includes classes for XML elements, XML documents, XML schemas, XML attributes, and URIs.

In summary, the Resource Description Vocabulary is an ontology of metadata descriptions themselves. Its aim is to provide a semantically sound framework for overcoming the encoding problems described in Section 4.2 of this report. The next section walks us through a demonstration of how this ontology, as used by BECHAMEL, can help to highlight potential preservation risks.

4.3.4. Resolving Semantic Ambiguity: an Inference Example

In Section 4.2 of this paper we discussed problems of descriptive practice, encoding standards, and schema design. Now we present an illustration of how our inferencing software responds to those problems. In the example below, an ambiguous metadata statement from the record shown in Figure 10 is identified and associated with the implied preservation target it describes.

Figure 11 below shows one RDF statement extracted from Figure 10, our running example:

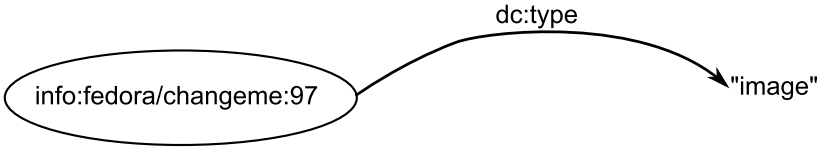


Figure 11: A fragment of the record shown in Figure 10.

This RDF statement view shows the string value "image" assigned to the Dublin Core type property for the Fedora Object identified as *info:fedora/changeme:97* – as we discussed earlier when viewing the original markup record (Figure 10). The main issue is one of clearly identifying the ***target*** of our preservation efforts: an image in this case. Summarizing the concerns discussed earlier:

- The Fedora object is an amorphous resource, which seems to share properties of the image itself, the image content, and the bitstream encoding the image. The Fedora object cannot, therefore, be our preservation target.
- According to the formal schema definition, the Dublin Core Type property indicates the "the nature or genre of a resource," but need not identify the existence of any particular concrete object or abstract entity. As already seen, this vagueness in the formal schema opens the door to the use of values (such as the literal string "image") that are clear to human readers but which pose problems for machine processing.
- Although the word "image" invites a human reader to infer that our preservation target is an image, that information is not explicit enough to support automated processing. The inference depends not only on word meaning but also on the tacit background knowledge that the property value must in this case be a class (rather than, for example, a quality, quantity, or name).

To recap then, this image (Figure 11) illustrates the relationships between the Fedora object, the DC element "type", and the value "image" ambiguously expressed in the original record (Figure 10). In the next step, we begin to clarify these relationships.

Figure 12 below shows the first inference stage:

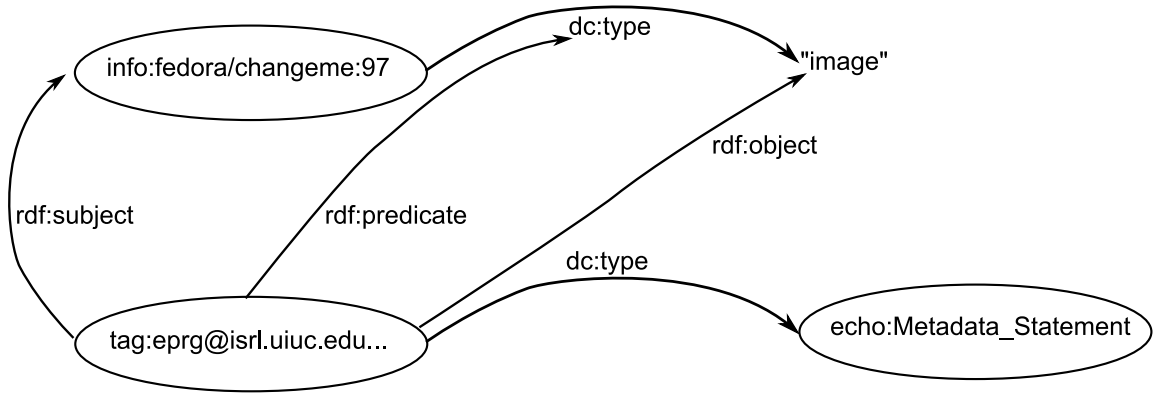


Figure 12: BECHAMEL has identified the fragment as a metadata statement.

This RDF statement shows that the original *dc:type* arc has been identified as a metadata statement, and a new TAG URI has been generated to denote that statement. Although this stage of the processing began with conventional RDF reification, our assignment of *rdf:subject*, *rdf:predicate*, and *rdf:object* properties to our new *Metadata_Statement* instance is a departure from orthodox RDF semantics. This first stage of inference processing has identified the metadata statement. In the next stage we take this a step further to identify the preservation target.

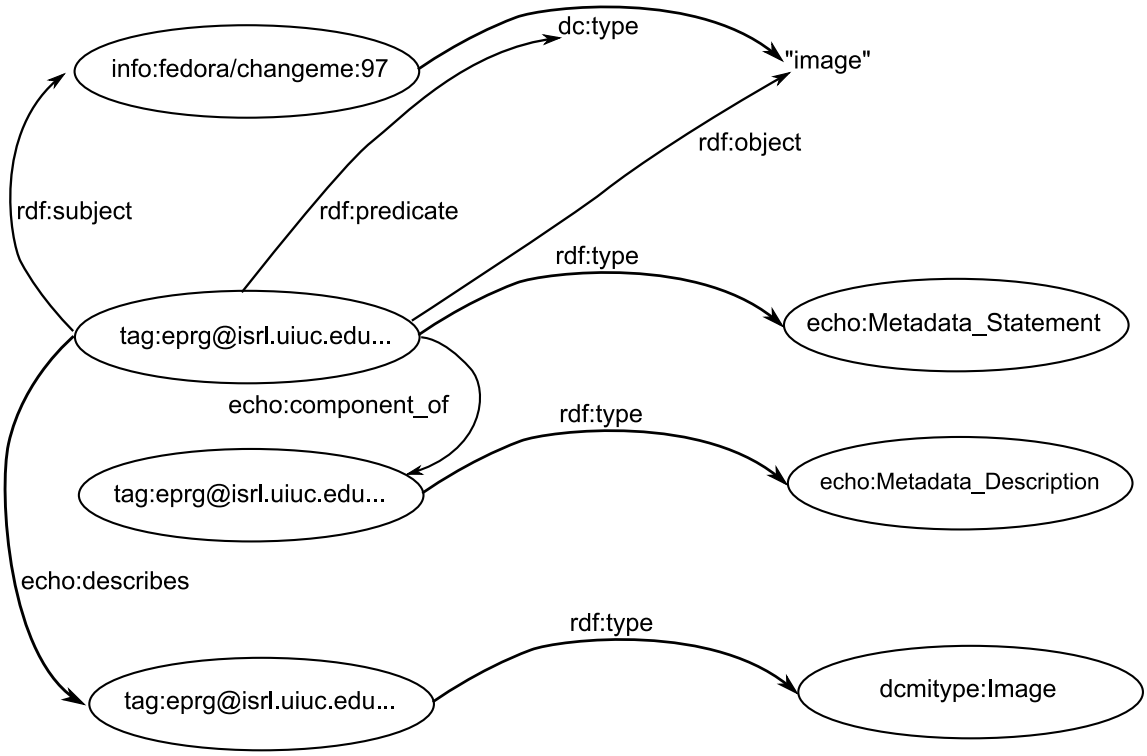


Figure 13: BECHAMEL has identified the metadata statement as a description of an image.

Figure 13 shows the identification of the preservation target. The system infers that this metadata statement must be describing an abstract image that has both a class identity and an object identity distinct from the JPEG file, the bitstream encoding that file, the geography depicted in the image, and the Fedora object that serves as the locus for property attributions at all those levels. In addition, the metadata statement is identified as belonging to a metadata description. Identifying the preservation target should simplify the validation of later preservation transactions, making it easier to verify that essential properties persist across migrations and through translations from one format to another.

4.3.5. Automated Inference as a Preservation Service

The ontology and inferences that it supports allow us, even in cases where metadata records are terse and incomplete, to recover important distinctions, such as the

distinction between a person and the metadata record describing that person. This knowledge is expressed in a portable syntax (RDF/OWL) with explicitly-defined semantics, so it can be maintained without having to modify the original record or transform it into another syntax (either of which could introduce further preservation risks). Indeed, BECHAMEL's ability to read and write from RDF databases (using Tupelo) means that it can read metadata records, apply rules and infer new assertions, and write those assertions back to the RDF database without altering the original records in any way. The "open world" of RDF/OWL means that automated inference can become a part of the preservation process without requiring that we redesign and reimplement institutional repositories to accommodate it. Instead, inference is a kind of service that can be used alongside those tools to head off preservation risks and fill gaps in representation.

The next section looks more closely at the architecture and proof-of-concept implementation of an archive that augments an institutional repository with inference capabilities and services.

4.4. System Architecture

We respond to the practice, standardization, and technical problems previously outlined in two ways:

- First, we design our systems for a world where metadata will vary greatly in their completeness, expressivity, and consistency. Preservation risks will arise, and we build tools with the aim of ameliorating those problems.
- Second, we propose an architecture for repositories that we hope will support more effective resource description and encoding: one that includes capabilities and services that will be needed in the next generation of digital content management systems.

4.4.1. Architecture: Overview

The proposed architecture augments typical institutional repository architectures with two new capabilities:

- The ability to manage not just bitstreams and associated metadata, but also associated semantics, expressed in standard RDF and OWL syntax.
- Automated services for detecting and/or correcting semantic ambiguity in metadata descriptions.

4.4.1.1. Architecture: the Tupelo model

Tupelo is a middleware component providing semantic content management for distributed, heterogeneous applications. By middleware, we mean that Tupelo provides abstractions (known as contexts) that encapsulate different storage and retrieval technologies for data and metadata, including file systems, web services,

relational databases, and RDF stores. By way of these contexts, applications can exchange RDF statements and access raw octet streams associated with them. Tupelo can therefore serve the same role as a content management system (CMS) or institutional repository. But Tupelo differs from these systems in making only minimal assumptions about the structure of the information it manages, allowing applications to encode that structure as explicit RDF statements. RDF's open-world assumption and use of Uniform Resource Identifiers means that Tupelo can assemble descriptions from multiple, independent sources, even if those sources are not otherwise coordinated.

Tupelo has originally been designed to support science applications where data is produced, processed, and transformed by multiple people and software components. Such applications require preservation of workflow traces and the tracking of relationships between raw input and output results across distributed systems. These same challenges arise in digital preservation, where critical transformations may occur outside of the control of a repository system, or within metadata whose semantics are known at one stage of the process and unknown at another. Such transformations are distributed, heterogeneous processes, and tying a digital artifact to the process in which it participated requires portable, globally-scoped identifiers that can be managed independently of the process itself. RDF usage enforces the global scope of identifiers by using URIs to identify nodes.

4.4.1.2. Connecting BECHAMEL to Tupelo

Our BECHAMEL client application retrieves an XML-serialized subgraph of the repository contents from Tupelo via Tupelo's HTTP-based client/server protocol, which is based on extending Nokia's proposed URIQA protocol (http://sw.nokia.com/uriqa/URIQA.html) The subgraph is submitted to BECHAMEL, together with supporting OWL Ontologies and standardized RDFS vocabularies (e.g., Dublin Core). New RDF statements and annotations emerging from BECHAMEL's execution (see the inference example in Figures 2-4) are then delivered back to the Tupelo server.

4.4.1.3. Observations on Implementation

Like the characteristics of the Tupelo architecture, we predict that inferential capabilities (such as those illustrated earlier in this section) will be basic services provided by and for future digital repositories. But the functional components of those repositories will be loosely coupled and distributed. Interpretive services are, furthermore, needed right away for systems based on current Content Management System technologies, and to aid in reforming descriptive practice as it stands today. For all these reasons, we have sought in our implementation to make the interpretation component a structurally distinct layer, communicating with the Tupelo middleware via general-purpose client/server protocols such as HTTP. While we assume the resource descriptions and inferred knowledge will conform to

the RDF abstract model, we have chosen to deliver them in conventional serialized forms, such as RDF/XML.

As with any similar project, a variety of engineering challenges require further experimentation and improvement. For example, at the BECHAMEL application layer, all RDF statements are expressed as if part of a single global graph, whether retrieved from Tupelo, parsed from an RDFS vocabulary, inferred by BECHAMEL itself, or drawn from any other source. But obviously, only a finite amount of input knowledge can be efficiently shared over the network between client and server. Our interpretive rules are themselves separate from the strategies for selecting, retrieving, and storing RDF statements, but pragmatically they cannot be totally independent of each other.

4.5. Lessons Learned and Next Steps

Our research contribution can be seen from one perspective as the technical groundwork for a future generation of improved automated digital preservation systems and methods. But one can also understand our findings as opportunities to apply human intelligence more effectively with existing tools and standards. It might never occur to a digital librarian that his preservation methods are being executed without clearly identifiable targets, or that a simple change (such as *dcmitype:Image* instead of "image") could dramatically reduce the work required to correct that problem. The exercise of encoding semantic knowledge with enough clarity and precision for a computer reveals complexities that our remarkable human minds would otherwise allow us to ignore. With the aid of that insight, much progress could be made in reforming the practices that prompt our development and research.

4.6. Conclusion

Institutional repositories and other current efforts for preserving digital artifacts face challenges resulting from underspecified metadata schemas, ambiguous usage, and metadata models that relate more to repository implementation than to issues of meaning. These entail very real risks to the integrity and usefulness of preserved digital artifacts as they are stored, managed, and retrieved. Descriptive practices that seem correct may introduce inconsistencies that are undetectable without manual inspection of each record -- an unreasonable requirement for collections of even moderate size.

Improved metadata standards and repository metadata models are part of the solution to these problems, but we also see a role for automation in detecting and mitigating preservation risks. Our experimental archiving technologies, BECHAMEL

and Tupelo, demonstrate that we can locate and correct ambiguous metadata expressions in the context of transactions such as import and export. As best practices evolve for digital preservation, we see reasoning capabilities like those demonstrated by BECHAMEL becoming an integral component of digital preservation systems, allowing curators to transform large collections with greater confidence that records will faithfully represent the information they are intended to preserve. Complementing interoperability models like ECHO DEPository's Hub and Spoke tool suite, we believe the techniques described here point to a new generation of preservation tools, and reveal ways to use existing tools with more success.

5. A Note from the PIs

When the University of Illinois Library and the Graduate School of Library and Information Science submitted the proposal for the ECHO DEP project to the Library of Congress in 2003, the digital preservation landscape was radically different. The number of web archiving tools was still small; far fewer institutions than now had instances of repository software applications in their libraries; the problem space of interoperability between repository platforms was just gaining ground; and techniques for migrating the semantic content of documents over time and through various encoding schemes were still on the horizon. The accomplishments of ECHO DEP Phase 1 projects, in the form of technical frameworks and software applications, as well as of published research and enduring partnerships, have contributed to the redesign of this landscape for the richer and more sustainable.

For example, our efforts at enabling repository interoperability have resulted in the registration of the ECHO DEP Generic METS Profile for Preservation and Digital Repository Interoperability with the Library of Congress. Because of our work in this area, institutions such as Harvard University, the Arizona State Library, and the Georgia Institute of Technology have contacted us to learn more about the technical architecture issues involved in our framework. These contacts bespeak knowledge sharing and community building toward a public good – interactions that are integral to the development of a networked approach to digital content stewardship.

Another beneficial outcome has been the partnerships themselves, initially established during Phase 1, such as with OCLC; Illinois and OCLC are collaborating again in ECHO DEP Phase 2, this time on a named-entity extraction and recognition tool development project that seeks to automate creation and extraction of metadata for preservation purposes and contexts. Indeed, the work of starting and sustaining cross-organizational collaboration for a project's period of performance should not be overlooked. As our teams have learned in the process, effective collaboration entails – but is not limited to – laying a foundation for a communication infrastructure that draws on an array of tools, such as wikis and virtual meeting applications; nurturing a healthy balance between encouragement of new directions in research and development *and* meeting the deliverables to which the project is committed; and understanding from the start that the outcome of our efforts will only be as meaningful and successful as the collaborations themselves are rich and productive.

The University of Illinois is grateful to the Library of Congress for funding its digital preservation research activities under NDIIPP. The work achieved during Phase 1 has afforded us a greater understanding of the challenges surrounding preservation strategies, which we hope the NDIIPP community at large will continue to learn from and draw upon in future stewardship endeavors.

6. References

6.1. Archiving the Web: the Web Archives Workbench

6.1.1. Resources

Cobb, J., Pearce-Moses, R. & Surface, T. (2005). ECHO DEPository Project. In Archiving 2005: final program and proceedings, April 26, 2005, Washington, D.C., (175-178). Springfield, VA: The Society for Imaging Science and Technology, 2005. Retrieve July 5, 2008, from http://www.ndiipp.uiuc.edu/pdfs/IST2005paper_final.pdf/.

The ISO Reference Model for Open Distributed Processing – An Introduction

ECHO Dep Generic METS Profile for Preservation and Digital Repository Interoperability. (2005). Retrieved August 27, 2008, from http://www.loc.gov/standards/mets/profiles/00000015.html.

ECHO Dep METS Profile for Web Site Captures (2006). Retrieved August 27, 2008, from http://www.loc.gov/standards/mets/profiles/00000016.html.

The ECHO DEPository: An NDIIPP-Partner Project of the University of Illinois at Urbana-Champaign with OCLC and the Library of Congress. (n.d.). Retrieved July 5, 2008, from http://www.ndiipp.uiuc.edu/.

"The ISO Reference Model for Open Distributed Processing – An Introduction." (1996). Retrieved August 27, 2008, from http://www.enterprise-architecture.info/Images/Documents/RM-ODP2.pdf.

OCLC Digital Management Services. (2008). Retrieved July 5, 2008, from http://www.oclc.org/us/en/services/collection/default.htm.

The National Digital Information and Infrastructure Preservation Program. (n.d.). Retrieved July 5, 2008, from http://www.digitalpreservation.gov/.

Pearce-Moses, R. & Kaczmarek, J. (2005). An Arizona Model for Preservation and Access of Web Documents. DttP: Documents to the People. 33(1), 17-24. Retrieved July 5, 2008, from http://www.ndiipp.uiuc.edu/pdfs/azmodel.pdf/.

Rani, S., Goodkind, J., Cobb, J., Habing, T., Eke, J., Urban, R. & Pearce-Moses, R. (2006). Technical architecture overview: tools for acquisition, packaging, and ingest of web objects into multiple repositories (poster). Opening information horizons: 6th ACM/IEEE-CS Joint Conference on Digital Libraries : June 11-15, 2006, Chapel Hill, NC, USA : JCDL 2006 / sponsored by ACM SIG on Information Retrieval, ACM SIG on Hypertext, Hypermedia and the Web, IEEE Technical Committee for Digital Libraries, (360-360). New York: ACM, 2006.

Web Archives Workbench. (2008). Retrieved July 5, 2008 from http://sourceforge.net/projects/webarchivwkbnch/.

Web archiving. (2008, August 21). In Wikipedia, the free encyclopedia. Retrieved August 27, 2008, from http://en.wikipedia.org/wiki/Web_archiving.

6.2. Repository Evaluation and Interoperability

6.2.1. Repository Evaluation

DLI Full-Text Journal Collection. Retrieved April 7, 2009, from http://forseti.grainger.uiuc.edu/pubs/tocdli.asp.

Historical Aerial Photo Image Database. Retrieved April 7, 2009, from http://images.library.uiuc.edu/projects/aerial_photos/.

Illinois Digital Orthophoto Quarter Quadrangle Data. Retrieved April 7, 2009, from http://www.isgs.illinois.edu/nsdihome/webdocs/doq05/.

RLG. (2005). *An audit checklist for the certification of trusted digital repositories*. Mountain View, CA: RLG. Retrieved September 10, 2008, from http://worldcat.org/arcviewer/1/OCC/2007/08/08/0000070511/viewer/file2416.pdf.

Vincent Voice Audio Library at Michigan State University Libraries. Retrieved April 7, 2009, from http://vvl.lib.msu.edu/showfindingaid.cfm?findaidid=CoolidgeC.

6.2.2. HandS Tools Suite

Allinson, F., François, S., & Lewis, S. (January, 2008). SWORD: Simple Web-service Offering Repository Deposit. *Ariadne*, (54). Retrieved September 11, 2008, from http://www.ariadne.ac.uk/issue54/allinson-et-al/.

Boyko, A., Kunze, J., Littman, J., & Madden, L. (2008). *The BagIt File Package Format (V0.95)*. Retrieved September 15, 2008, from http://www.cdlib.org/inside/diglib/bagit/bagitspec.html.

Consultative Committee for Space Data Standards. (2002). *Reference Model for an Open Archival Information System (OAIS).* CCSDS 650.0-B-1. Blue Book. Retrieved September 12, 2008, from http://public.ccsds.org/publications/archive/650x0b1.pdf.

DLF Aquifer Metadata Working Group. (2006). *Digital Library Federation / Aquifer Implementation Guidelines for Shareable MODS Records*. Retrieved September 10, 2008, from http://wiki.dlib.indiana.edu/confluence/download/attachments/24288/DLFMODS_ImplementationGuidelines_Version1-2.pdf?version=1.

Global Digital Format Registry (GDFR) Information Site. (n.d.). Retrieved September 15, 2008, from http://www.gdfr.info/.

Guenther, R. (2008). *Guidelines for using PREMIS with METS for exchange*. Retrieved September 11, 2008, from http://www.loc.gov/standards/premis/guidelines-premismets.pdf.

Guenther, R. (2008). Battle of the Buzzwords: Flexibility vs. Interoperability When Implementing PREMIS in METS. *D-Lib Magazine*, 14 (7/8). Retrieved September 12, 2008, from http://www.dlib.org/dlib/july08/guenther/07guenther.html.

Habing, T.G. (2005). *ECHO Dep generic METS profile for preservation and digital repository interoperability*. Retrieved September 10, 2008, from http://www.loc.gov/standards/mets/profiles/00000015.xml.

Habing, T. G. (2006). *ECHO Dep METS profile for web site captures*. Retrieved September 10, 2008, from http://www.loc.gov/standards/mets/profiles/00000016.xml.

Habing, T. G. (2007). *Lightweight repository CRUD Service (LRCRUDS)*. Retrieved September 10, 2008, from http://dli.grainger.uiuc.edu/echodep/hands/LRCRUDS.htm.

JHOVE: JSTOR/Harvard Object Validation Environment. (2007). Retrieved September 11, 2008, from http://hul.harvard.edu/jhove/.

Kaczmarek, J., Habing, T. G., & Eke, J. (2006). Repository software evaluation using the audit checklist for certification of trusted digital repositories. In *Proceedings of the 6th ACM/IEEE-CS joint conference on digital libraries 2006, Chapel Hill, NC, USA June 11 - 15, 2006*. New York: Association for Computing Machinery. Retrieved September 10, 2008, from http://doi.acm.org/10.1145/1141753.1141774.

Kaczmarek, J., Hswe, P., Eke, J., & Habing, T. G. (2006). Using the 'Audit checklist for the certification of a trusted digital repository' as a framework for evaluating

repository software applications. *D-Lib Magazine*, 12(12). Retrieved September 10, 2008, from http://www.dlib.org/dlib/december06/kaczmarek/12kaczmarek.html.

METS: Metadata encoding & transmission standard, official web site (2008). Retrieved September 10, 2008, from http://www.loc.gov/standards/mets/.

MIX: NISO metadata for images in XML schema, technical metadata for digital still images standard, official web site. (2008). Retrieved September 10, 2008, from http://www.loc.gov/standards/mix/.

MODS: Metadata object description schema, official web site. (2008). Retrieved September 10, 2008, from http://www.loc.gov/standards/mods/.

OAI-PMH: Open Archives Initiative – Protocol for Metadata Harvesting. (2008). Retrieved September 12, 2008, from http://www.openarchives.org/pmh/.

PREMIS: Preservation metadata maintenance activity, official web site. (2008). Retrieved September 10, 2008, from http://www.loc.gov/standards/premis/.

PREMIS Working Group. (2005). *Data dictionary for preservation metadata*. Dublin, OH: OCLC and RLG. Retreived September 10, 2005, from http://www.oclc.org/research/projects/pmwg/premis-final.pdf.

RLG. (2005). *An audit checklist for the certification of trusted digital repositories*. Mountain View, CA: RLG. Retrieved September 10, 2008, from http://worldcat.org/arcviewer/1/OCC/2007/08/08/0000070511/viewer/file2416.pdf.

JISC. (2008). *SWORD*. Retrieved September 11, 2008, from http://www.ukoln.ac.uk/repositories/digirep/index/SWORD.

textMD: Technical Metadata for Text, Official Web Site. (2008). Retrieved September 10, 2008, from http://www.loc.gov/standards/textMD/.

The ECHO DEPository project. (n.d.). Retrieved September 9, 2008, from http://ndiipp.uiuc.edu/.

The Apache Software Foundation. (2008). *Welcome to XMLBeans*. Retrieved September 11, 2008, from http://xmlbeans.apache.org/.

UIUC Echodep Hub and Spoke Framework Tool Suite. (n.d.). Retrieved September 12, 2008, from http://dli.grainger.uiuc.edu/echodep/hands/

6.3. Preserving Meaning, Not Just Objects: Semantics and Digital Preservation

DCMI Namespace for the Dublin Core Metadata Element Set, Version 1.1. (2008). Retrieved October 2, 2008, from http://dublincore.org/2008/01/14/dcelements.rdf

DCMI Type Schema. (2008). Retrieved October 3, 2008, from http://dublincore.org/2008/01/14/dctype.rdf

Dubin, D., Sperberg-McQueen, C. M., Renear, A., and Huitfeldt, C. (2003). A logic programming environment for document semantics and inference. *Literary and Linguistic Computing*, 18(2):225–233.

Habing, T., Ingram, W., Cordial, M., Manaster, R. and Eke. J. (2008). Developments in digital preservation at the University of Illinois: the Hub and Spoke architecture for supporting repository interoperability and emerging preservation standards. *Library Trends, 57*(4), [page nos.].

Kaczmarek, J., Hswe, P., Hauser, L., and Eke. J. (2008). The Web Archives Workbench: taking an archival approach to the preservation of Web content. *Library Trends, 57*(4), [page nos].

Kaminski. P. (2002). *Integrating information on the semantic web using partially ordered multi hypersets*. Unpublished master's thesis. University of Waterloo. Retrieved September 12, 2008, from http://www.ideanest.com/braque/Thesis-web.pdf.

OWL Web Ontology Language. (2004). Retrieved October 2, 2008, from http://www.w3.org/TR/owl-features/

RDF Semantics. (2004). Retrieved October 2, 2008, from http://www.w3.org/TR/rdf-mt/

Renear, A., Dubin, D., Sperberg-McQueen, C.M., and Huitfeldt, C. (2002). Towards a semantics for XML markup. In E. Munson, R. Furuta, and J.I. Maletic (eds.) *Proceedings of the 2002 ACM Symposium on Document Engineering* (119-126). New York: ACM.

Renear, A. and Dubin, D. (2003). Towards identity conditions for digital documents. In S. Sutton, editor, *Proceedings of the 2003 Dublin Core Conference*. University of Washington, Seattle, WA.

Tupelo. (2008). Retrieved October 2, 2008, from http://dlt.ncsa.uiuc.edu/wiki/index.php/Main_Page.

URIQA. The URI Query Agent Model: A Semantic Web Enabler. (2003-2008). Retrieved October 2, 2008, from http://sw.nokia.com/uriqa/URIQA.html.

7. Appendices

7.1. Web Archives User Guide

7.2. Web Archives Workbench Implementation Guide

7.3. Annotated Trusted Digital Repository Checklist

7.4. Using the Audit Checklist for the Certification of a Trusted Digital Repository as a Framework for Evaluating Repository Software Applications (DLib article)

7.5. Repository Testing Findings: Narrative

7.6. Repository Findings Commentary Using the Annotated Trusted Digital Repository Checklist

7.7. Resource Description Vocabulary: An Ontology of Metadata Descriptions

7.8. Sustained Access to Ejournals: Context Value, and Future Prospectus