ISS – LIBRARY OF CONGRESS ARCHIVE INGEST AND HANDLING TEST (AIHT)

FINAL REPORT OF THE STANFORD DIGITAL REPOSITORY

Stanford University Libraries
& Academic Information Resources
May 2005
(Minor errors corrected December 2006)

ISS – LIBRARY OF CONGRESS ARCHIVE INGEST AND HANDLING TEST (AIHT)

FINAL REPORT OF THE STANFORD DIGITAL REPOSITORY

Stanford University Libraries
& Academic Information Resources

❑ *INTRODUCTION*

The Archive Ingest and Handling Test (AIHT) was a project devised and seed-funded by the Library of Congress to generate practical data points and knowledge from the experiences had by institutions building digital preservation infrastructure with a small real-world digital archive. As expressed in section 2.3.1 of the AIHT Statement of Work (SOW) 26504–01, 12/19/2003:

> *The Archive Ingest and Handling Test (AIHT) is designed to test the feasibility of transferring digital archives in toto from one institution to another. Phases of the test will assess the process of digital ingest, document useful practices, maximize automated handling of digital material, and identify areas that require further research or development.*
>
> *This project will not test issues of general public search or access, rights clearance or management, long-term viability of various storage media, security, evidentiary provenance, or terabyte-scale ingest.*

Key team members within Stanford University Libraries & Academic Information Resources (SULAIR) for the AIHT test were:

- Richard Anderson, Stanford Digital Repository Software Developer
- Catherine Aster, Digital Services Group Project Manager
- Connie Brooks, Head, SULAIR Preservation
- Hannah Frost, Media Preservation Librarian
- Nancy Hoebelheinrich, SULAIR Metadata Coordinator
- Keith Johnson, Stanford Digital Repository Project Manager, Project Manager, 05/2004–03/2005
- Michael Keller, University Librarian; Director of Academic Information Resources; Publisher of HighWire Press; Publisher of Stanford University Press, Principal Investigator (PI)
- Jerry Persons, SULAIR Chief Information Architect, Project Manager, 02/2004–04/2004

Additional valuable administrative support was provided by Stanford University's Office of Sponsored Research, SULAIR Strategic Projects and Finance offices—in particular Catherine

Boxwell, Senior Contracts Officer, Andrew Herkovic, Strategic Projects Director, and Susan Horsfall, Budget Officer.

As specified in Stanford's AIHT proposal, AIHT project management was initially the responsibility of SULAIR Chief Information Architect, Jerry Persons. This responsibility was transitioned to Keith Johnson, Stanford Digital Repository Project Manager, in May 2004.

As enlarged upon below, the project proved to be valuable for all participants, and validated both the approach taken by the Library of Congress and the decision by Stanford to request participation in it. The project yielded both tangible and intangible benefits; in particular, the strong pattern of effective collaboration that developed among the participants achieved some real synergy and suggests future productive shared efforts in digital preservation.

❑ *BACKGROUND*

Previous to the AIHT project, the Stanford Digital Repository (SDR) process and workflow design was focused primarily on preserving specific corpora of content from the SULAIR's Digital Library projects that comprised large, well-defined, highly normative collections of digital objects. The SDR team worked to define profiles for instantiation of these objects, so that those objects would reliably conform with SDR preservation policy. In addition, it was possible to pre-establish agreements with the content creators that allowed the SDR to reject any objects not conforming to the agreed-upon profiles. This, while being a very efficient way to handle internal projects, is a luxury not possible in many other workflows.

The scope of envisioned SDR services is much broader than those initial services, so it was clear that the SDR would need to develop capability to offer valuable and scalable preservation services for arbitrarily created digital objects and heterogeneous collections. The GMU 911 Archive (the AIHT test-set) is an excellent example of a heterogeneous collection of digital content objects created with the best intentions of preserving digital historical and cultural artifacts, but constructed without consideration of a digital repository's preservation recommendations or policies. The AIHT test provided SULAIR with the opportunity to further its development work in this important area, and to do so in a rich, practical and collaborative environment that specifically focused on the partnership aspects of preservation—a critical SULAIR strategic goal as evidenced by its leadership role in the Digital Library Federation Aquifer project.

Before Stanford's participation in the AIHT project, an internal group—the Technical Assessment Group (or TAG team)—performed significant research on methodologies for assessment of arbitrarily created digital objects. The team developed a questionnaire to embody its assessment methodology. Its purpose was to facilitate the following:

> *A mechanism was needed to raise questions and to record the answers, a tool to gather vital information about an object, or groups of like objects, and to explicitly express the intent or will of the "content owner," the person who best knows the information resource, its creation, and its value. The tool needs to accomplish these functions within*

the context of what is more or less known to be technically possible with respect to digital preservation, and then assess the interplay of these factors in order to set reasonable expectations for both content owners and repository managers about the prognosis for maintaining accessibility to the information encoded in the digital objects over time.

The full questionnaire provided a methodology, framework, and mechanism for assessing object classes for digital preservation services; SULAIR attached it to its AIHT RFP response with the intent that it constitute the foundation work for the SDR's preservation assessment of heterogeneous collections.

❑ *INVESTIGATION*

Initial attempts to extend and adapt the TAG Team Questionnaire to perform assessment on heterogeneous collections quickly lead the AIHT Team to several issues:

- The questionnaire's mechanism doesn't scale well since it relies solely on human resource.
- The questionnaire and its underlying methodology can capture only the knowledge of the depositor, which admittedly is essential to documenting intended use and significant properties of the digital object classes. However, the SDR cannot rely on the depositor's knowledge to include or describe accurately the true technical state of digital objects in the collection.
- The questionnaire therefore attempted to gather two distinct classes of information: use intent and value, which can only be obtained from a person, and technical state, which is more appropriately obtained from an empirical process.

The AIHT team also looked closely at the 911 collection in the context of our manual organization, assessing, and describing activities. Our realizations and observations led our team towards synthesis of a generic workflow for heterogeneous collections (such as the AIHT test-set) with some new automated components improving efficiency and accuracy. The envisioned pre-ingest workflow (following) provided the team's paradigm for the AIHT project:

1. Depositor brings a heterogeneous collection to the SDR.
2. Before starting a dialog, the repository runs an automated tool to:
    a. Discover and describe the physical structure of the collection;
    b. Inspect each file in the collection for format, then analyze for validity and other preservation factors;
    c. Compare results of this item-level analysis to current repository policy, then assign an "object complexity" or "preservation risk" status to each file;
    d. Summarize the results. To the extent possible, use visual representations of the information, e.g., through Grokker, for most efficient communication; and
    e. Capture the initial state of the collection for evidentiary purposes and to enable exact recreation of the original presentation for future digital archaeology.
3. Informed by the empirical results from the tool, repository staff familiarize themselves with the collection and determine preservation and metadata services options.

4. Repository staff initiate a dialog with the depositor to make preservation and metadata decisions; the results of the automated tool facilitate the efficiency and accuracy of the dialog and therefore the quality of the decisions. Depending on the nature of the specific collection, the dialog could include:
    - Review of predominant file formats (or object classes) within the collection tied to repository preservation service policy, metadata services, and costs for those services related to the different object types. Investigation of any relationship between object class groups and curatorial intent.
    - Verification of physical and logical collection structure. Investigation of potential relationships between the collection's physical structure and logical structure. Determination of existence and significance of sub-collections.
    - Interrogation of the collection for decisions or actions by file type, sub-collection, preservation risk, curatorial intent or value, extant metadata, or some combination of the above.
    - Prioritization of curatorial intent, decisions about processing actions at the point of ingestion, definition of stewardship commitment by the repository, clarification of access policies, and explication of long-term preservation funding options and commitments.
5. Repository negotiates contract agreement detailing and committing results of the dialog.
6. Repository employs automated tools to perform agreed-upon format transformations and other ingest-time preservation tasks. Alternatively, manual tools could be employed if resources and funding exist, or depositor could take responsibility for transformations.
7. Repository staff use browsing tools to identify, analyze, describe, and mark up significant sub-collection nodes in the collection.
8. Repository staff evaluate any user-supplied metadata, then use automated tools to map, duplicate, and enhance (when feasible) these metadata to normalized standard types and structure for ingest.
9. Automated tools package collection content and metadata into "normative" Submission Information Package (SIP).
10. Automated ingest begins: repository technology verifies and ingests SIP into repository, transforming it into Archive Information Package (AIP) components.

Significantly, the idealized workflow assumes the existence and cooperation of a knowledgeable depositor. However with the AIHT test collection, and presumably often in the real world, this assumption is false. In that case, the repository must perform the role of depositor, which makes automated inspection, browsing, and organizing tools critically important.

Two factors also influenced our development strategy for the AIHT test. First, at the time the test began, production AIP specification was still pending for the SDR, though METS had been chosen as the overall packaging standard. For this reason, we decided to focus on generating normative METS-based transfer forms (i.e., SIPs) instead of AIPs. This decision later proved problematic for our partners in Phase III, which fact suggested some important design distinctions between SIPs and AIPs. Secondly, SDR prototype development already employed database technology, so the team knew its strengths and weaknesses for metadata handling and content packaging. Given METS as our packaging standard, we were motivated to experiment with stress-testing pure XML tools to see to what extent the complexity of database environments could be avoided in the ingest and preservation processes.

❑ STRATEGY AND WORK PLAN

Informed by our idealized model and previous exercises in ingest automation, we created an initial strategy and work plan. Its most significant features included:

1. Ingest Phase:
   - Build an automated file system mapper for our ingest toolkit: a recursive, directory-walking, item-level description, analysis, and assessment tool. Ensure that it contain a pluggable framework which could drive current, as well as future, external file inspection tools. (Eventually we dubbed this tool the "Empirical Walker.")
   - Limit the scope of our development environment to java and XML-oriented tools. This aligned with our developing preservation environment decisions, our learning needs, and our desire to keep the preservation environment as simple as possible.
   - Align the collection packaging as closely as possible with our evolving AIP specification. (The closer the SIP is in structure to the AIP, the easier automation and scaling of the assessment and ingestion process should be. Therefore, the tool should output METS directly.) Devise a METS structure as a normative SIP specification for heterogeneous collections.
   - Codify format-assessment and preservation policies into machine-actionable form, then build or integrate tools to automate the assessment process. Plug these tools into the Empirical Walker.
   - As time allows, build other envisioned pieces of the toolset, especially those facilitating the dialog with a depositor. These could include a visualization environment to summarize output of the Empirical Walker, a toolkit for metadata transformation and mapping, or a documentation toolkit for the decisions and commitments made in a dialog with the depositor.
2. Transfer Phase:
   - Establish and implement mechanism for transfer to other repository(ies). Since our Phase I activities would result in a normalized SIP, it follows that we would simply transfer that SIP to the other repository.
3. Re-Ingest Phase:
   - Deploy the Empirical Walker to discover whether it was easier or more valuable for our tools to create a normalized SIP from another repository's SIP than from the original raw collection.
   - If not, investigate whether the Empirical Walker could be extended or adapted to extract more value from another repository's SIP.
4. Transformation/Migration Phase:
   - Attempt migration of complex objects — specifically websites – in concordance with our original RFP Response.
   - Extend the Empirical Walker functionality or create a new tool to transform objects *in situ* in METS content packages.

❑ *THE TEST*

AIHT team members wrote in-depth, detailed descriptions of the methodologies, activities and products of each phase of the test; these are attached as appendices. The purpose of this section is to give an overview of each phase to provide context to the reader in navigation of the appendices.

Phase I-Ingest

Phase I occupied the majority of Stanford's time and effort in the test. Our activities divided roughly into three areas: methodology development, data model and metadata format development, and programming.

As specified in the AIHT SOW, Stanford's first task was to verify successful receipt of the Archive from the Library of Congress by validating their Transfer Manifest (TM) against the files we received. We developed the first iteration of the Empirical Walker to create our own TM to compare to the Library's. This methodology not only fit our overall strategy, but also allowed us to proceed with development in the absence of the LC TM and complete significant work before it became available on July 28, 2004. We achieved success generating our own TM and comparing it against the LC TM, verified that all files in the test-set were successfully transferred, and posted our results on August 20, 2004. Though most of this process was automated, some manual work was also necessitated by the differences in organization between the two files; this argues for stricter definition of the TM format for successful automated verification. A complete, detailed report on this work is attached as Appendix A: *Report on AIHT Transfer Metadata (Archive Manifest).*

Concurrently, the team began extending and automating the format-identification and risk-assessment portions of the TAG Team Questionnaire. We identified two key external efforts in the digital preservation community to build upon: for format identification and analysis, JHOVE (The JStor Harvard Object Verification Environment)[1]; and for risk-assessment methodology the work of the Library of Congress Office of Strategic Initiatives as presented by Carl Fleischauer and Caroline Arms at the Digital Library Federation Fall Forum 2003.[2] In Phase I we completed the following tasks:

- Utilized JHOVE to provide automated file format identification and analysis for the collection.
- Developed a "Format Scoring Matrix" based on the Arms and Fleischauer work as the first instantiation of SDR preservation risk policy.
- Codified the matrix into machine-actionable XML form.
- Developed algorithms to apply the preservation risk policy automatically to a collection.
- Developed a packaging data model and METS structure to express the results of our automated inspection and analysis, thereby describing the collection.
- Examined the descriptive metadata structure and content manually.

[1] <http://hul.harvard.edu/jhove/>
[2] <http://www.diglib.org/forums/fall2003/fallforum03.htm#p1>

- Employed automated tools to transform the descriptive metadata to individual XML files and link them to their content files.
- Incorporated all of the above into the Empirical Walker.

The final Empirical Walker tool recursively traverses a file tree, calculates checksums, determines file format type, analyzes file contents, and performs preservation status assignment and risk assessment based on simple configuration files. Input comes both from the command line and from configuration files codifying preservation policy, METS output structure, and other parameters. Output is to a master METS XML file, linked to supplemental individual metadata files, also in XML format. This output represents a draft iteration of a normative SIP for heterogeneous collections.

Running the Empirical Walker on the AIHT collection yielded the following pleasantly surprising preservation policy results:

Assigning Policy Status to AIHT files:		
Policy Status	**file count**	**%**
preferred	25102	43.7
approved	29929	52.1
acceptable	1342	2.3
minimal	0	0
unknown	1077	1.9

Full details of Stanford's activities and results for Phase I are attached in appendices as follows:

- Appendix A: Report on AIHT Transfer Metadata (Archive Manifest)-*Describes all activities leading to verification of the Library of Congress Transfer Manifest*
- Appendix B: Preservation Assessment of Digital Objects-*Describes development of automated preservation-risk assessment methodologies. Includes Format Scoring Matrix and Workflow Diagram*
- Appendix C: Preservation Assessment of AIHT Files by SDR Method-*Summarizes results of SDR preservation assessment of AIHT files*
- Appendix D: Empirical Walker Documentation-*Documents Empirical Walker technology in a narrative fashion. JavaDoc documentation provided electronically to the Library Of Congress along with source*
- Appendix H: TAG Team Questionnaire-*A version of the original questionnaire including information on which questions were successfully automated in the AIHT Test (21 of 41 or ~ 50%)*

Phase II-Export, Transfer, and Import

Testing this phase of the project, Stanford focused on generating a SIP rather than importing the contents of that SIP into our preservation environment. Our "export" consisted of delivering our SIP as a Dissemination Information Package (DIP).

AIHT partners chose to exchange DIPs via an internet-based "drop-box" server hosted at Old Dominion University rather than via shipped media. Stanford was pleased that this method worked for multi-gigabyte archives, despite concerns that the time required bodes poorly for scaling. Full details of the transfer and the structure and contents of Stanford's export appear in Appendix E: *Stanford University AIHT Export Files.*

Stanford chose to import the SIP from Johns Hopkins University (JHU) because we thought it posed an interesting and, we hoped, instructive challenge. This version had a structure significantly different from our own, and thus proved difficult for us to assimilate. For instance, there was no single document that corresponds to our notion of a master metadata file. Instead the metadata was located in 59,552 separate METS documents (one per original data file or directory).

JHU provided program code that we could have used to facilitate our import process. As it happened, we did not take advantage of this code and instead performed two separate, but complementary, actions:

- We treated their package as an unknown, analyzing the entire content with the Empirical Walker as if it were data. This allowed us to be in a position to recreate the JHU export at any future time.
- We wrote a separate program that read each of the JHU item.xml METS documents. The file path information in these files was used to create a master METS structMap representing the original archive's directory hierarchy and filenames.

Empirical Walker revealed scaling issues when importing the JHU archive; since that archive contained a METS file for each content file, it had roughly twice the number of files as the original ingest test-set. Specifically, the Empirical Walker caused out-of-memory errors on our server configuration. We also discovered that the JHU archive contained four fewer content files than the original archive. Perhaps JHU has already explained the missing files, but if not, we believe this discrepancy may be connected to errors in the original test-set metadata as delivered in database format. Full details on Stanford's import experience are attached as Appendix F: *Report on Stanford's Ingest of the Johns Hopkins Export.*

Phase III-Preservation Transformation

Stanford's original goal in this phase was to explore the transformation of complete web pages or web sites into alternative formats, such as TIFF or PDF. As we entered this phase, the Library of Congress encouraged us to reevaluate our goal in light of the compressed timeframe. The Library also encouraged us to continue developing our automated risk-assessment work. We decided to narrow our focus to transformations of HTML files into XHTML format, and to devote the majority of our effort to the evaluative process of identifying which files of a given format type were the best candidates for transformation. We, therefore, enhanced Empirical Walker (and its input rule set), so that the program's preservation-risk assessment procedures would better support the automation of file transformations. The revised version flagged files for transformation based on the analysis of JHOVE output for specific risk factors, and wrote out a list of those flagged files as part of its summary.

To perform the transformations, we wrote a program that utilized the JTidy implementation of HTML Tidy. It stored output files in a single directory separate from the original collection, and produced a summary file listing the original filename, transformation filename, log filename, and transformation success or failure for each transformation attempt.

We successfully transformed the vast majority of HTML files to XHTML using HTML Tidy. Only 121 out of 17003 files had errors severe enough to prevent transform. To check for any differences in rendering, we visually compared a limited number of before and after files. We observed no problems.

A fully detailed report on Stanford's experience in Phase III is attached as Appendix G: *Report on Stanford Mass Transformations of AIHT HTML Docs to XHTML Format.*

Interesting Unexpected Events

As requested by the Library of Congress, following are selected unanticipated events the Stanford team found interesting.

Stanford's design for content packaging in the AIHT test specifically maintained file names as they were received, providing some evidence of collection and sub-collection. At the time we believed that this preserved as much simplicity as possible in the package and reduced the effort required to unpack. We discovered in the transform phase, however, that this approach made updating the archive much more difficult; this led us to believe that our AIP packaging might need to be fundamentally different from our SIP packaging despite our wish to keep them closely aligned. On further reflection, we realized that we were easily able to maintain filenames in our SIP only because the originating filesystem (ext3) was very similar to our ingest cache filesystem (ufs). It therefore turns out that a more complex approach, i.e., moving filenames to metadata records for packaging, might turn out to be generally simpler for most repository operations.

Gigabyte-scale archives transfer successfully over the Internet. However, it still took hours, which does not bode well for scaling repository transfer or replication across the current Internet—even between well-connected higher-education institutions.

We invested substantial effort transforming the archive's descriptive metadata from custom relational database records to individual XML files. While other AIHT partners were mapping the existing descriptive metadata to standard schema (e.g., JHU mapped to MODS), our practice was to capture the metadata content raw as delivered, and convert it into a form supported by general XML tools and easily referenced by our SIP, the intended result being a richer representation of the original metadata,. However, our encoding the metadata in XML led to difficulties for our receiving archive partner's ingest process. It may have been much more efficient for our partner if we had mapped the metadata in a more compatible manner. This suggests that any future transmission packaging standards should include the explicit facility to wrap "raw" metadata for evidentiary purposes.

Our manual inspection of the archive led us to anticipate that it would have a high preservation risk in aggregate, yet over 97% of the collection's files had at least "acceptable" preservation assessment or better. That still left over one thousand files identified at high risk, and our process today is only capable of assessing individual files, not complex objects, which might have dropped the number. Nevertheless we were expecting fewer files to be acceptable.

Limiting our toolset to java, XML tools, and direct expression of METS resulted in requiring more computer resources than we had hoped. There is reason to believe the resource consumption probably stemmed from programming issues rather than from a fundamentally un-scalable approach.

The further we got into the test, the more we realized how the community work we had to build on was critically important, especially JHOVE. Without JHOVE we could not have taken the approach we did, and in fact JHOVE was a dependency for all test partners.

Stanford's experience during the negotiation phase demonstrated that the Library of Congress and its contractor ISS had a steep learning curve ahead to master effective collaboration with academic libraries; this led us to expect a rocky road for the duration of the project. The fact that they both navigated that curve within the first half of the project—that by the second half all partners were working together enthusiastically and efficiently—was a felicitous outcome.

Lessons Learned

Stanford developed the Empirical Walker with an intentionally modular approach that would make it valuable as a framework to drive other tools. However, we did not initially appreciate how far we should push the modular approach. The Empirical Walker framework itself is essentially monolithic, and this ultimately frustrated later efforts to utilize pieces of its functionality separately or in a different order. What we learned is that the more modular the development approach, the more flexible the toolkit will be, and that workflows requiring a flexible approach—such as the reactive ingest of a heterogeneous collection—benefit from this. We are now exploiting this lesson in designs for repository tools.

Stanford invested considerable effort debating the motivations for and the ramifications of content packaging decisions. For instance, where should the metadata exist in a package, in one file or in many, and in what structure? How should the content structure relate to the metadata structure, and how should it be containerized? We hoped to develop a generically-useful archival packaging framework. In the process, we reached several general conclusions:

- *Contemporary* technology and *contemporary* understanding of content use drives choices in content packaging—these choices typically optimize for some current constraint. For instance, .tar containers are broadly supported, simple, and easily transmitted, yet do not support random, low-latency granular access. METS has a shallower learning curve and more human intelligibility than MPEG21-DIDL, yet MPEG21-DIDL may currently be more easily machine-processable. These factors will change, over time. Therefore, Stanford should be prepared not only to migrate files over time, but continuously to evaluate the efficacy of packaging methods—even archival ones—over time. We should also recognize that the constraints for archiving may be different enough from the constraints of actual use of digital objects (for certain classes of content or tools) to necessitate that the archival packaging be substantially different from the use packaging.

- The AIHT team entered the test with a bias towards maximizing human-intelligibility of archival packages. This was driven by the assumption that simplicity would facilitate future digital archaeology. Only time will prove or disprove this assumption; it is nonetheless probably a worthy aspiration. However, our unexpected difficulty maintaining filenames (referenced in Interesting Unexpected Events, above) demonstrates that this aspiration should not seriously compromise repackaging or other machine processing over time. We now view Digital Archaeology as an extreme measure to be avoided at all costs, but not eliminated from possibility through our decisions. This argues for optimizing our package designs more for machine processing than for human-intelligibility.
- Following that logic, we concluded that file identifiers, and probably all identifiers in a package, should be deliberately free of semantic meaning. All semantics should instead go explicitly into metadata, and package identifiers should be optimized for uniqueness, machine identification efficiency, and automated processing.
- Whenever possible, digital collections should be packaged for transfer to a repository *in situ* in their original technology environment using natively running tools. AIHT demonstrated the cost perils inherent in breaking that rule: mixing together Linux, Windows, Unix, ext3, ntfs, ufs, tar, gtar, etc. in the packaging and transfer mechanics of the archive caused nothing but wasted time and headaches. Combining this conclusion with the previous two, we realized that ideally we should have tools that package digital collections in their native computing environment and move all environmentally stored semantics explicitly into package metadata. Examples are filenames, extensions (and their meaning in the specific environment), permissions and other filesystem metadata, and proprietary environmentally-implemented relationships such as links, shortcuts, or aliases.

Practical tests are valuable. We believe this is one of the most significant and general lessons stemming from this project. We obtained more benefit from this process than we had expected at the start. The freedom and encouragement to experiment with different approaches while being forced to work with practical, real-world content enabled Stanford to generate directly-applicable results—applicable both for its digital preservation efforts and, we are confident, for the digital preservation community at large.

Partnership creates value. The practical nature of the AIHT test by itself was valuable, but the successful partnership during the project greatly reinforced and magnified that value. Stanford reached noteworthy conclusions and developed useful tools during the test; we would have accomplished far less without benefit of the discussions with, the shared experiences of, and the tools provided by our partners. Heterogeneity can also create value. Stanford understood that deliberately complementary partners were chosen. Our experience in the test validates this choice—the diversity of the group brought richness to our interactions that would have been unlikely in its absence. Stanford University thanks our test partners—Harvard University, The Johns Hopkins University, Old Dominion University, The Library of Congress—for their extremely fruitful collaborative approach. We especially thank the Library of Congress for devising the test and establishing the environment that allowed these partnerships to flourish.

❑ *FUTURE DEVELOPMENT POSSIBILITIES*

As requested by the Library of Congress, following are several suggestions for future development funding.

One clear follow-on project idea inspired by the AIHT is the collaborative development of one or several packaging standards for the transfer of digital collections, with the goal of providing easier transfer of collections and building shareable tools for interoperability. The different packaging approaches adopted by the AIHT partners would inform this collaboration. The intention would be creation of one set of standards for the digital preservation community—and maybe the digital content community at large—which would form the basis for collaborative maintenance and tool-building. The Stanford AIHT team developed a hypothesis, neither proven nor debunked during the test, that a heterogeneous collection transmission package standard could be used almost universally for digital content transmission. If truly heterogeneous collections can be packaged consistently, then homogeneous collections can presumably be packaged using the same standard, as could simple objects ("collections of one"). Even if the standard were itself relatively complex, it could dramatically reduce the overall complexity of creating and maintaining interoperable digital preservation modules and systems. It would be useful to the community to test this hypothesis.

Following on the previous idea, and inspired by lessons learned, we suggest a project be funded to develop and maintain *in situ* heterogeneous collection packaging (or harvesting) tools for contemporary computing architectures. If a reasonably universal collection transmission package standard is possible, then it should also be possible to create relatively lightweight tools to enable digital-preservation-quality *in situ* collection harvesting. AIHT reinforced the critical importance of scaling and automation to digital preservation. Providing that automation outside of the digital preservation community could dramatically reduce the cost of collecting at-risk information.

Finally, we suggest further collaborative development of automated ingest and preservation tools such as the Empirical Walker. During the AIHT, Stanford successfully created one piece of an envisioned automated ingest-facilitating toolkit through a combination of internal development and external tool integration. To round out the toolkit and assure its continued utility as digital preservation matures, it is clear that collaborative, interoperable development is both necessary and effective.

APPENDICES

Report on AIHT Transfer Metadata (Archive Manifest) by Stanford University Libraries

Richard Anderson
(rnanders@stanford.edu)

2004-08-20

Introduction

The *Archive Ingest and Handling Test (AIHT) Statement of Work (SOW)* contains the following test requirement:

2.4.1.1. The archive will be delivered to participants accompanied only by the metadata that existed at the moment of delivery to the Library of Congress. The Library will not conduct any verification or post-processing of the archive other than generating a list of the archives contents in Transfer Metadata (TM) format, a simple format developed by the Library of Congress that shall serve as the manifest for the contents of the archive at the time of its delivery.
2.4.1.2. The TM format is analogous to a digital bill of lading, and represents the minimum metadata. It is required for confirmation of delivery of a particular group of digital material. The format for this file is attached in appendix A.
2.4.1.3. A TM file will accompany each data transfer from the Library to ISS. Upon receipt of the archive, participants shall produce a TM file of the material and transmit back to the Library.
2.4.1.4. Every subsequent export of data for the AIHT shall be accompanied by a TM manifest.

The purpose of this document is to describe Stanford's procedure for examining the Library of Congress TM manifest (TMD file), generating a local TM manifest from the untarred archive, and comparing the two manifests to verify that we have received all files and that all files have the same checksum as calculated by LC.

The first section below describes where the LC TMD file was obtained, a peek at its contents and structure, and some comments on aspects of the TMD file that provide challenges when comparing against a locally produced TMD file.

The remainder of the document describes Stanford's programming efforts toward generating a TMD file, a peek at the Stanford TMD file, and the methodology used to compare manifests.

Note that the TMD files produced by LC and Stanford are in XML format, rather than the CSV format specified in Appendix A of the SOW.

The TMD file supplied by Library of Congress

Announcement of TMD file availability

Date: Wed, 28 Jul 2004 11:52:27 -0400
From: "David Hafken" <dhaf@loc.gov>
To: <aiht@mail.iss-loc.com>
Subject: [aiht] TMD File available

I've uploaded the TMD file (compressed in Windows ZIP format) for the GMU 9/11 Archive to Sharepoint. It is in the "Shared Documents" folder in the "Documents and Lists" section. You might be able to get to it using this link:
http://www.iss-loc.com/aiht/Shared%20Documents/archiveManifest.zip

The format of the file is simple, and consists of two sections: one that provides some metadata that describes the overall collection, and another that specifies each individual file by its full path and md5 checksum value.

[Note that I attempted download of this file, but could not obtain a valid file from the Sharepoint server. Instead I received a copy as email attachment from David Hafken.]

Content of the LC TMD file

Here is a very abbreviated peek at the contents of the LC TMD file (archiveManifest.xml):

```
<?xml version="1.0" encoding="iso-8859-1"?>
<tmdFile>
      <collection>
            <title>The September 11 Digital Archive</title>
            <sentTo>Library of Congress</sentTo>
            <sentFrom>George Mason University</sentFrom>
            <sentDate>February 20, 2004</sentDate>
            <archivePackage type="GNUTar" name="911da.tar" os="linux" filesystem="ext3
                  <checksum type="md5">2be822d4d0d8098e752048c5a80c683a</checksum>
            </archivePackage>
      </collection>
      <items>
            <item type="file" count="1">

      <fileAbsolutePath>/websites/chnm/september11/REPOSITORY/CONTRIBUTORS/.bkdate
                  </fileAbsolutePath>
                  <checksum type="md5">a51d72f5663358875c808b111aafb26c</checksum>
            </item>
            <item type="file" count="2">
                  <fileAbsolutePath>/websites/chnm/september11/REPOSITORY/CONTRIBUTORS
                        99_photos/winmail.dat
                  </fileAbsolutePath>
                  <checksum type="md5">cf78d7751191ab89af1713b7f6f40669</checksum>
            </item>
            ...
      </items>
</tmdFile>
```

Structure of the LC TMD file

Using Altova XMLSpy, I generated the following W3C XML Schema from the LC TMD file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v2004 rel. 4 U (http://www.xmlspy.com) by Richard Anderson (Stanford
University) -->
<!--W3C Schema generated by XMLSPY v2004 rel. 4 U (http://www.xmlspy.com)-->
<xs:schema elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="archivePackage">
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="checksum"/>
			</xs:sequence>
			<xs:attribute name="type" type="xs:string" use="required"/>
			<xs:attribute name="name" type="xs:string" use="required"/>
			<xs:attribute name="os" type="xs:string" use="required"/>
			<xs:attribute name="filesystem" type="xs:string" use="required"/>
		</xs:complexType>
	</xs:element>
	<xs:element name="checksum">
		<xs:complexType>
			<xs:simpleContent>
				<xs:extension base="xs:hexBinary">
					<xs:attribute name="type" type="xs:string" use="required"/>
				</xs:extension>
			</xs:simpleContent>
		</xs:complexType>
	</xs:element>
	<xs:element name="collection">
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="title"/>
				<xs:element ref="sentTo"/>
				<xs:element ref="sentFrom"/>
				<xs:element ref="sentDate"/>
				<xs:element ref="archivePackage"/>
			</xs:sequence>
		</xs:complexType>
	</xs:element>
	<xs:element name="fileAbsolutePath" type="xs:string"/>
	<xs:element name="item">
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="fileAbsolutePath"/>
				<xs:element ref="checksum"/>
			</xs:sequence>
```

```
                <xs:attribute name="type" type="xs:string" use="required"/>
                <xs:attribute name="count" type="xs:int" use="required"/>
            </xs:complexType>
        </xs:element>
        <xs:element name="items">
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="item" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="sentDate" type="xs:string"/>
        <xs:element name="sentFrom" type="xs:string"/>
        <xs:element name="sentTo" type="xs:string"/>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="tmdFile">
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="collection"/>
                    <xs:element ref="items"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:schema>
```

Observations about the LC TMD file

- The content of the <collection> element is hard coded into the program used by LC to generate the archiveManifest.xml file.

- The `<archivePackage> element specifies name="911da.tar"`
  and MD5 `checksum = 2be822d4d0d8098e752048c5a80c683a`
  The file provided on the hard disk is a gziped version of this tar file, named `"911da.tar.gz", and has a MD5 checksum = 9d59240371a07248a9198571854b4d5e.`
  I have therefore ungzipped the .gz file to create the .tar file in order to compare checksums.

- Every <fileAbsolutePath> element begins with /websites/chnm/september11/REPOSITORY/. Relative paths would probably be preferable from my viewpoint, because
    1. Having the same string in every path is redundant
    2. Relative paths would be easier for a human to read
    3. When we untarred our copy of the archive it ended up having a different "absolute path" than was present in the original archive location.

- Some of the <fileAbsolutePath> elements in the LC TMD file are not in alphabetical sequence. Here is example containing two filenames out of sequence:

```
/CONTRIBUTORS/alfonso_gatto/4-capt.1000355732terrorist_attack_vigil_rvg101.jpg
/CONTRIBUTORS/alfonso_gatto/4-front.jpg
/CONTRIBUTORS/alfonso_gatto/4-capt.1000356917terrorist_attacks_vigil_rvg103.jpg
```

```
/CONTRIBUTORS/alfonso_gatto/4-capt.1000357234terrorist_attacks_vigil_rvg104.jpg
/CONTRIBUTORS/alfonso_gatto/4-capt.1000386032britain_us_attacks_reax_lmn106.jpg
/CONTRIBUTORS/alfonso_gatto/4-capt.1000423542terrorist_attacks_nyr166.jpg
/CONTRIBUTORS/alfonso_gatto/4-capt.1000473628terrorist_attacks_escape_ny203.jpg
/CONTRIBUTORS/alfonso_gatto/4-capt.1000473659terrorist_attacks_escape_ny204.jpg
/CONTRIBUTORS/alfonso_gatto/4-capt.1000473705terrorist_attacks_escape_ny205.jpg
/CONTRIBUTORS/alfonso_gatto/4-capt.1000473738terrorist_attacks_escape_ny206.jpg
/CONTRIBUTORS/alfonso_gatto/4-imdf12092001142510a.jpg
/CONTRIBUTORS/alfonso_gatto/4-capt.1000473950terrorist_attacks_escape_ny202.jpg
/CONTRIBUTORS/alfonso_gatto/4-capt.1000499961canada_us_attacks_remembrance_ott106.jpg
```

Stanford's TMD file Generator

The FileWalker

Stanford chose to create a Java application for this purpose. A single program recursively walks the file tree, calculates the checksums, and generates the XML output.

- The basic filewalker was adapted from code that uses the java.io.File object for recursive descent into a file hierarchy. Basic logic is:

```
static void process(File parentFile) {
  if (parentFile.isDirectory()) {
      // Do directory specific stuff
      String[] childFilenames = parentFile.list();
      if (childFilenames.length > 0) {
          Arrays.sort(childFilenames);
          for (int i = 0; i < childFilenames.length; i++) {
              File childFile = new File(parentFile, childFilenames[i]);
              // recurse here!
              process(childFile);
          }
      }
      else {
          // Empty Directory
      }
  }
  else {
      // Do file specific stuff
      // (e.g. compute checksum)
  }
}
```

- java.security.MessageDigest is a standard Java library that provides a procedure for generating a MD5 calculation:

```
static String Md5HashString(File currentFile) throws
        NoSuchAlgorithmException {
    try {
        // Calculate MD5 Hash value of data
        MessageDigest md5 = MessageDigest.getInstance("MD5");
        FileInputStream fis = new FileInputStream(currentFile);
        byte[] buffer = new byte[1024];
        int read;
        while ( (read = fis.read(buffer)) != -1) {
            md5.update(buffer, 0, read);
        }
        fis.close();
```

```
            byte[] digest = md5.digest();

            // Convert byte array into a 32 byte hex string
            StringBuffer hexBuffer = new StringBuffer(digest.length * 2);
            for (int i = 0; i < digest.length; i++) {
                int j = digest[i] & 0xff;
                // append a zero before a one digit hex number to make it two digits.
                if (j < 16) {
                    hexBuffer.append("0");
                }
                hexBuffer.append(Integer.toHexString(j));
            }
            String hexString = hexBuffer.toString();
            return hexString;
        }
        catch (IOException ie) {
            System.out.println("IOException: " + ie);
            return " ";
        }
    }
```

- Note that an earlier version of the above function read an entire file into memory before generating the checksum. This worked fine for small files, but produced java.lang.OutOfMemoryError when attempting to read a very large file (e.g. > 500MB). A better approach is to read in one chunk of the file at a time and calculate the checksum incrementally. With this change the program successfully ran in 91MB of memory.

- The Castor Open Source data binding framework (http://www.castor.org/) was chosen as the toolkit for generating the XML output. The Borland JBuilder X Developer environment that I use for programming can generate Castor classes for each element type using an XSD schema file as a starting point. These Castor classes "know" about the XML element hierarchy, make it easy to assign values to elements and attributes, and include a "marshal" function for serializing the data. This output mechanism (marshalling) produces a XML file with no line breaks between elements.

```
            TmdFile myTmdFile = new TmdFile();
            Collection myCollection = new Collection();
            myTmdFile.setCollection(myCollection);
            myCollection.setTitle("The September 11 Digital Archive");
            myCollection.setSentTo("Library of Congress");
            myCollection.setSentFrom("Stanford University");
            myCollection.setSentDate("August 19, 2004");
            ArchivePackage myArchivePackage = new ArchivePackage();
            myCollection.setArchivePackage(myArchivePackage);
            myArchivePackage.setType("GNUTar");
            myArchivePackage.setName("911da.tar");
            myArchivePackage.setOs("solaris 8");
            myArchivePackage.setFilesystem("ufs");
            Checksum myChecksum = new Checksum();
            myArchivePackage.setChecksum(myChecksum);
            myChecksum.setType("md5");
            myChecksum.setContent("2be822d4d0d8098e752048c5a80c683a");
            Items myItems = new Items();
            myTmdFile.setItems(myItems);
        For each file:
            Item myItem = new Item();
            itemcount++;
            myItem.setCount(itemcount);
            myItem.setType("file");
            myItem.setFileAbsolutePath(currentFile.getCanonicalPath());
            Checksum myChecksum = new Checksum();
            myItem.setChecksum(myChecksum);
            myChecksum.setType("md5");
            myChecksum.setContent(hexString);
            _Items.addItem(myItem);
        Finally:
            Marshaller.marshal(myTmdFile, writer);
```

Stanford's TMD file

Stanfords TMD file will be uploaded to the Sharepoint website as a separate document. Here is what the first section looks like when viewed in an XML editor:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<tmdFile>
   <collection>
      <title>The September 11 Digital Archive</title>
      <sentTo>Library of Congress</sentTo>
      <sentFrom>Stanford University</sentFrom>
      <sentDate>August 19, 2004</sentDate>
      <archivePackage filesystem="ufs" name="911da.tar" os="solaris 8"
                      type="GNUTar">
         <checksum type="md5">2be822d4d0d8098e752048c5a80c683a</checksum>
      </archivePackage>
   </collection>
   <items>
      <item count="1" type="file">
            <fileAbsolutePath>/websites/chnm/september11/REPOSITORY/CONTRIBUTORS/.b
      kdate
            </fileAbsolutePath>
         <checksum type="md5">a51d72f5663358875c808b111aafb26c</checksum>
      </item>
      <item count="2" type="file">
            <fileAbsolutePath>/websites/chnm/september11/REPOSITORY/CONTRIBUTORS/11
      99_photos/winmail.dat</fileAbsolutePath>
         <checksum type="md5">cf78d7751191ab89af1713b7f6f40669</checksum>
      </item>
      . . .
    </items>
</tmdFile>
```

Comparison of Stanford and LC manifest files

A direct differences comparison between Stanford and LC manifest files was not possible. What I did therefore was to write a SAX-based parser that extracted the path and checksum information from each item element into a flat text file. I ran this against both files. I edited Stanford's flat file so that the filepath prefix on each line begins with `>/websites/chnm/…` Next I sorted both flat files (using unix sort) and did a diff of the two sorted files, which demonstrated that the two files had the same basic content.

Times required for analysis

Hardware = Sun Fire 4800 with Sun StorEdge T3 Array and SUN StorEdge 6320 disk array (7 TB)
four 750 MHz processors , 4 GB memory + 4 GB virtual memory

The command "gunzip -c < 911da.tar.gz > 911da.tar" required 18 minutes to complete

It took 28.25 minutes to calculate the checksum of the 911da.tar file

It took 31 minutes to traverse the file tree and compute checksums of all content files.

APPENDIX B

Stanford University Libraries and Academic Information Resources Preservation Assessment of Digital Objects

Hannah Frost

Part 1: The TAG Team Questionnaire

Background

In the early stages of developing a digital preservation program at Stanford, it was clear that the Stanford Digital Repository (SDR) would need to offer a range of services to its prospective clients. While the initial streams of content to be preserved were known to be highly normative and predictable, indications of the varied nature of the materials, limited resources, and enormous volume of files to be preserved in the future necessitated a tiered approach to repository services, such as metadata encoding, pre-ingestion transformations, long-term format migration and delivery, in addition to bit preservation.

A team of individuals -- the Technical Assessment Group, or TAG Team -- came together to develop a framework for categorizing digital objects to be preserved. The thinking was that such a framework would not only prevent the SDR from becoming an undifferentiated heap of content, but that it would also further the development of administrative principles and policies around which the SDR infrastructure and service model would grow.

In approaching the task of developing tiered services for categories of digital objects, the following questions immediately arose:

- How will inevitable change affect the nature of the digital objects stored in the SDR? What may become lost in the process of migrations and transformations?
- What attributes, if any, of an object are crucial to its on-going use and value as an information resource?What are the underlying technical characteristics of an object that may prevent those attributes from being preserved?
- What external (non-technical) factors, if any, may have a bearing on the extent of services appropriate for a digital object or collection?

A mechanism was needed to raise questions and to record the answers, a tool to gather vital information about an object, or groups of like objects, and to explicitly express the intent or will of the “content owner”, the person who best knows the information resource, its creation, and its value. The tool needs to accomplish these functions within the context of what is more or less known to be technically possible with respect to digital preservation, and then assess the interplay of these factors in order to set reasonable expectations for both content owners and repository managers about the prognosis for maintaining accessibility to the information encoded in the digital objects over time.

The Questionnaire

The group spent several months identifying the key issues, defining terms, reviewing the literature[1], studying formats and metadata elements, and consulting with colleagues. With the results of these efforts, we drafted a questionnaire that incorporated the laundry list of factors exposed in our research and discussions that may impede digital preservation, from a technical perspective, or otherwise may impact long-term management of the objects, from a collection administration perspective. The questionnaire also was designed to serve a secondary role in determining the degree of metadata (primarily administrative, including technical, some structural, and other preservation metadata) that may be necessary to adequately document some objects for long-term management. It focused on assessing the types of normative objects to be initially ingested in the SDR (i.e., TIFFs, ASCII text files, PDFs), but it was expected that the framework that the questionnaire represented could be extended in the future to other object types (e.g., audio, video, web, complex, etc.).

In order to manage the results of the survey, we decided upon the notion of a scale of complexity, a horizontal spectrum on which the relative "preservability" of an object can be gauged based upon its technical risk factors. For each risk factor revealed in or exhibited by an object, one point is scored. The more points scored for a given object, the increasingly complex its preservation is expected to be. The group felt that weighting the scores based on the questions was beyond our immediate technical expertise, and that in fact it might not be practical due to the unpredictability of the technological landscape in which digital preservation activities take place. Answers to questions aimed at exposing external, non-technical factors (such as circumstances of origin and acquisition, retention expectations, uniqueness/rarity, etc.) were not scored at all, because it was felt that additional input from curators as well as experience from further collections case studies was required before these factors could be carefully evaluated for their applicability.

Given that at the time of the questionnaire's completion, the SDR was still very much in prototype and had no dedicated management, development or production staff, the practical implementation of the questionnaire remained an open question. Under these circumstances, it was conceived that the completion of the questionnaire would be an iterative process mediated by a "repository liaison" (i.e., someone with technical expertise and who is involved in the production end of digital library projects) in consultation with the content owner. The score, if not the entire questioning process itself, would inform the negotiation of a service agreement. A web-based form was created (access limited to Stanford only), with an Oracle table as a backend to store the answers and scores, in order to demonstrate the concept.

The questionnaire is outlined in the attached document, `TAGQuestionnaireAIHT.pdf`. It is organized into the following sections:

Non-Technical Factors:	All formats
Technical Factors:	All formats
	Image
	PDF
	Text

Part II: Toward Automation: The AIHT Project

From the completion of the TAG Team questionnaire in April 2003, its implementation has continued to remain an open question. The questionnaire has served a role as a theoretical framework which is influencing the structure of developing SDR services. As the Stanford team has explained, with the AIHT project we intended to continue developing capabilities for assessing the preservability of information encoded in files preserved in the SDR through extending the concepts the questionnaire represents, and we have achieved that goal.

Shifts in the Approach

In order to apply the questionnaire's concepts to a large, heterogeneous collection like the GMU 9/11 Archive, it was obviously necessary to automate most, if not all, of the file assessment process. The work of two other organizations working in the digital preservation community contributed to Stanford's ability to automate preservation assessment of digital objects.

One break-through that supported our ability to automate was the availability of JHOVE, the tool created by Harvard that provides automated verification and identification, not to mention extraction of technical metadata, of a number of key file formats. With the possibility that JHOVE could expose technical vulnerabilities of specific files automatically, it was possible to abandon the instance-level approach which had been embedded in the questionnaire, in favor of a broader approach where an object is assessed more generally along the lines of its format type. A broader approach was key to automating the assessment of digital files; it follows that a more general framework of the factors that impact the preservability of information within formats was necessary. For this, we turned to the work of the Library of Congress Office of Strategic Initiatives as presented by Carl Fleischauer and Caroline Arms at the Digital Library Federation Fall Forum 2003 (see: http://memory.loc.gov/ammem/techdocs/digform/ and http://www.diglib.org/forums/fall2003/fallforum03.htm#p1).

The presentation titled "Digital Formats: Factors for Sustainability, Functionality, and Quality" describes the results of their work to "provide information to help LC staff develop strategies and practices for incoming [digital] content . . . by identifying preferred formats" (Arms and Fleischauer 2003). Two types of factors, *sustainability* and *quality & functionality* factors, emerged as the primary forces which have a bearing on whether or not a format can be considered preferable to others. Of interest is the table in which the factors making up the anticipated sustainability of a file format – disclosure, adoption, transparency, self-documentation, external dependencies, patents, and technology protection measures – are measured and analyzed against a handful of specific formats. This approach relates to Stanford's work, because it effectively generalizes and categorizes much of the spirit and some of the substance of the TAG Team questionnaire. We adopted it, in large part, and developed a matrix of our own for the analysis of predominant formats. It is this matrix that now serves as the basis underlying SDR preservation assessment activities and developing policies.

The Format Scoring Matrix

The Format Scoring Matrix is contained and described in greater detail in the attached document, `FormatScoringMatrix.pdf`. Having evolved over the course of the AIHT project, the matrix at this stage is in its most formal and developed state. While it appears to serve as a reasonable measure of a format's sustainability based on current knowledge, testing is required and additional revisions are likely. Several anticipated developments include the inevitable need to break out the various types of marked-up text for a more granular analysis of the distinctions between them. A close analysis of datasets is also called for to determine how the matrix can accommodate them. Further research and experimentation is necessary with respect to the final analysis of formats against the set of sustainability factors. Finally, the as-of-yet unexplored impact of relationships between highly complex compound objects will need to be accommodated in the overall SDR preservation assessment process. In the very near future, Stanford will be closely examining formats used for geospatial data, formats inherently more complex than those already addressed in our process. The matrix, indeed all rules which frame file assessment, will always be evolving.

It is worth noting that not all of the factors identified by the Library of Congress were adopted for use in Stanford's matrix. Excluded from the format analysis are two sustainability factors: "impact of patents" and "technology protection mechanisms". As Arms and Fleischauer acknowledge in their discussion, the topic of patent impact is a tricky one and needs further exploration. Even among some "standard" formats, the impact of patents can be felt. For the time being, the SDR assumes that the degree to which a format is encumbered by patents, or a similar formal claim on technological invention or innovation, directly affects the other sustainability factor of external dependencies and perhaps indirectly affects the factor of adoption.

With respect to technology protection mechanisms, they only have a bearing on the extent of services that the SDR can offer if, and only if, a specific file's internal technology protection mechanism is enabled. This state should be revealed and accounted for during routine file analysis processes prior to ingest. Therefore the potential for a file to have internal protections does not serve a purpose in the Format Scoring Matrix.

Also, the *quality and functionality* factors were not explicitly adopted in Stanford's Format Scoring Matrix. Such factors "pertain to current and future usefulness, e.g., for scholarship or repurposing". Specific characteristics indicative of a format's potential for quality and functionality include support for high resolution, color management, document structure and navigation, etc. (Arms and Fleischauer 2003). At Stanford, characteristics of quality and functionality are considered separately in the determination of SDR "preferred" formats (see below).

Part III: The SDR Preservation Assessment Process, version .01

Preservation assessment of files to be stored in and managed by the SDR occurs in several steps in the course of preliminary preparation of files for ingestion, a process carried out by a file-traversing program developed at Stanford called "The Empirical Walker." The steps can be generally described as follows (refer also to the diagram depicted in WorkflowSDRDigiProv.pdf):

1. Initial File Identification and MIME Type Assignment: The Empirical Walker is run on a specified collection of digital objects. As it traverses the directory(s) of files, it identifies file extensions and maps them to corresponding MIME types.

2. File Validation: A file validator (e.g., JHOVE) is invoked for those files for which there exists an applicable tool or module. Output is stored temporarily for subsequent processing.

3. Preservation Assessment Process Initiated: The Empirical Walker assigns a default Format Status by checking a registry of values, numbers 0-5, derived from the Format Scoring Matrix. Default Format Scores are grouped and matched with corresponding Preservation Quality Status levels. As the name implies, the Preservation Quality Status provides a relative qualitative measure as a result from the format assessment and serves as a useful gauge in subsequent file analysis.

   The complete results of this analysis are stored as preservation metadata in the METS Digital Provenance Section.

4. File Analysis: Analysis has two primary goals. The first goal is to examine the output of the file validation process in search of:
   a. invalid or not well-formed files;
   b. technical characteristics which could pose potential complications in preservation activities (such as a TIFF that is compressed);
   c. technical characteristics which indicate that the file has reduced preservation-risk associated with its default format status (such as a PDF which meets the anticipated PDF-A profile).

   A set of rules guides the Empirical Walker to flag any output pertaining to a specific file that could alter its preservation prospects, which up to this point are based solely on a general assessment of file formats articulated in the File Scoring Matrix.[2]

   The second goal of supplemental analysis is to identify those files that may benefit from transformation or normalization. The Empirical Walker determines if the extension represents a MIME type or format that has particular characteristics at risk of loss in future digital preservation activities and therefore could be transformed pre-ingest by means of reformatting or normalization in order to improve its preservation prospects. For instance, a Photoshop document (*.psd), which has a low format status (4), could be reformatted as a PNG or TIFF with little to no loss, and thus earn a higher preservation assessment as a result. Similarly a MS Word document could be reformatted as plain text for enhanced preservation of the textual content over time.

All file analysis results, including the benefits and risks resulting from any optional transformations, such as changes to the file's content, functionality, or look-and-feel, are reported to repository staff and/or the content owners (see below).

5. Preservation Policy Status Assignment: As a result of the file analysis steps, a suggested policy status level is assigned to the file. This value guides the SDR staff and content owners in the negotiation of an on-going preservation service agreement to be applied to the files or collection of files. There are five policy status levels: preferred, approved, acceptable, minimal, unknown.

   A class of preferred formats (as opposed simply to "approved") is necessary for business reasons because it focuses the number of formats for which the SDR is committed to providing full support services. Not all formats with a Format Score of zero automatically earn the status of a preferred format; the capacity of a format for enhanced quality and functionality must be factored in. Similarly, a Format Score of zero is not required to earn the "preferred" status; a format that is both highly suited to a specific purpose within the digital library context *and* free of risk factors does not always exist.

SDR Preferred Formats

Plain text	ASCII
	UTF-8
Marked-up text	XML 1.0
Image	TIFF 5.0 and above (uncompressed)
Page-Viewer	PDF* (any version)
Audio	WAVE (linear pulse code modulation)
Video	*To Be Determined*

*Despite the PDF format's lack of transparency and external dependencies, factors that give it a "medium", not high, Preservation Quality score, PDF is currently Stanford's preferred page viewing format, because it is the *de facto* standard with extremely wide market penetration.

For those files that do not qualify as preferred, another status level is assigned according to the Preservation Quality Status of the file. The following table outlines the correlation between scores and statuses.

Default Format Score	**Preservation Quality Status**	**Policy Status**
0-1	High	Approved
2-3	Medium	Acceptable
4	Low	Minimal
5	Low	Unknown

Written policies and specific services associated with the various policy levels remain under development.

6. Reporting: At the culmination of the file analysis, a report is issued to repository staff and/or content owners/depositors. The report includes the results and scores of the various analytical tests to which the files have been subjected. Those files with traits, exposed in supplemental analysis, which may require a modification of policy status or other subsequent action are indicated in the report. Possible outcomes reflected in the report include:
   a. *Change in status*: a file has been demoted or promoted from its default Preservation Quality status, influencing the type and extent of preservation services it qualifies for;
   b. *Transformation options*: possible target formats for an original file format are outlined, detailing any potential loss or gain associated with such action with respect to a file's contents, formatting, functionality, etc.;
   c. *Red flags*: a file has a particular technical trait or quality that is noteworthy but requires additional human input to determine an appropriate action, if any.

Preliminary Conclusions

While several parts of this process remain to be built and tested in full, we believe this multi-stage preservation assessment process provides an advantage of flexibility due to its modular design. Twenty-one of forty-one possible technical risk factors from the questionnaire have been identified or exposed in an automated fashion by way of this assessment process (they are indicated as such in `TAGQuestionnaireAIHT.pdf`). With the expanded capability of file validation tools in the future, it is expected that this number will increase over time. Through the incorporation of reporting to content owners results from the analysis steps, it will be possible to seek input from the repository client about the potential presence of specific characteristics of documents (such as links or other embedded interactivity and multimedia in proprietary file formats) that are at risk of loss over time but are not easily identified through automation, to manage the client's expectations for long-term preservation of content more generally, and to inform the client about transformation options or other repository services that may be appropriate for the files in question. The goal is for the reporting aspect to create a key role for the content owner by shifting some of the weight that the preservation assessment and decision-making process entails from SDR staff to the content owner. This goal is in keeping with our overall strategy to automate as much of the pre-ingestion file preparation process as possible and to push service-oriented tools to the repository clients wherever possible.

[1] Key sources consulted which informed the overall development of the questionnaire include: Bennett, John C. *JISC/NPO Studies on the Preservation of Electronic Materials: A Framework of Data Types and Formats, and Issues Affecting the Long Term Preservation of Digital Material*, British Library Research and Innovation Report No. 50, 1997, <http://www.ukoln.ac.uk/services/elib/papers/supporting/pdf/rept011.pdf>; Harvard Office for Information Systems *DRS Policy Guide*, 5 October 2001 <http://hul.harvard.edu/ois/systems/drs/policyguide.html>; and LeFurgy, William G. "Levels of Service for Digital Repositories," *D-Lib Magazine,* May 2002, Volume 8 Number 5, <http://www.dlib.org/dlib/may02/lefurgy/05lefurgy.html>.

[2] Evaluation of errors in HTML documents revealed by JHOVE is still on-going. Until complete testing is carried out, the file assessment process simply flags those HTML documents that are invalid or not well-formed; details or recommended outcomes have not yet been incorporated into the supplemental assessment and reporting process.

STANFORD DIGITAL REPOSITORY -- FORMAT SCORING MATRIX

The Format Scoring Matrix (below) provides a scale for measuring the long-term sustainability of a digital file format or group of like formats. It is based in large part on the work of the Library of Congress Office of Strategic Initiatives. Formats are evaluated against the five factors listed in the left columns (definitions for the factors appear on page 2). The format scores positively for a given factor if that factor is considered not to impede future digital preservation efforts; a negative score is assigned when the factor is expected to hinder preservation. The accumulation of negative scores results in an increasingly lower status for a format. Therefore, a higher score indicates that a format's prospects for sustainability are poorer than one with a low score. If no match on the matrix is made, or if the MIME type is determined to be application/octet-stream, a file undergoign assessment receives a score of 5. The resulting Format Score is matched to a corresponding Preservation Quality Value (see table to right).

Preservation Quality Values:

Neg. Count	**Status**
0	high
1	
2	medium
3	
4	low
5	

	Text		**Image/Graphic**							**Page-viewer**
	Plain (ASCII, UTF-8)	Marked-up (html, css, xml, sgml)	JPEG	GIF	TIFF	JPEG2000 (lossless comp)	BMP	PNG	Photoshop	PDF
Adoption	+	+	+	+	+	-	+	-	+	+
Disclosure	+	+	+	+	+	+	+	+	-	+
Transparency	+	+	+	-	+	+	+	+	-	-
Self-Documentation	+	+	-	-	+	+	+	+	-	+
External Dependencies	+	-	+	+	+	+	+	+	-	-
Negative Count	0	1	1	2	0	1	0	1	4	2

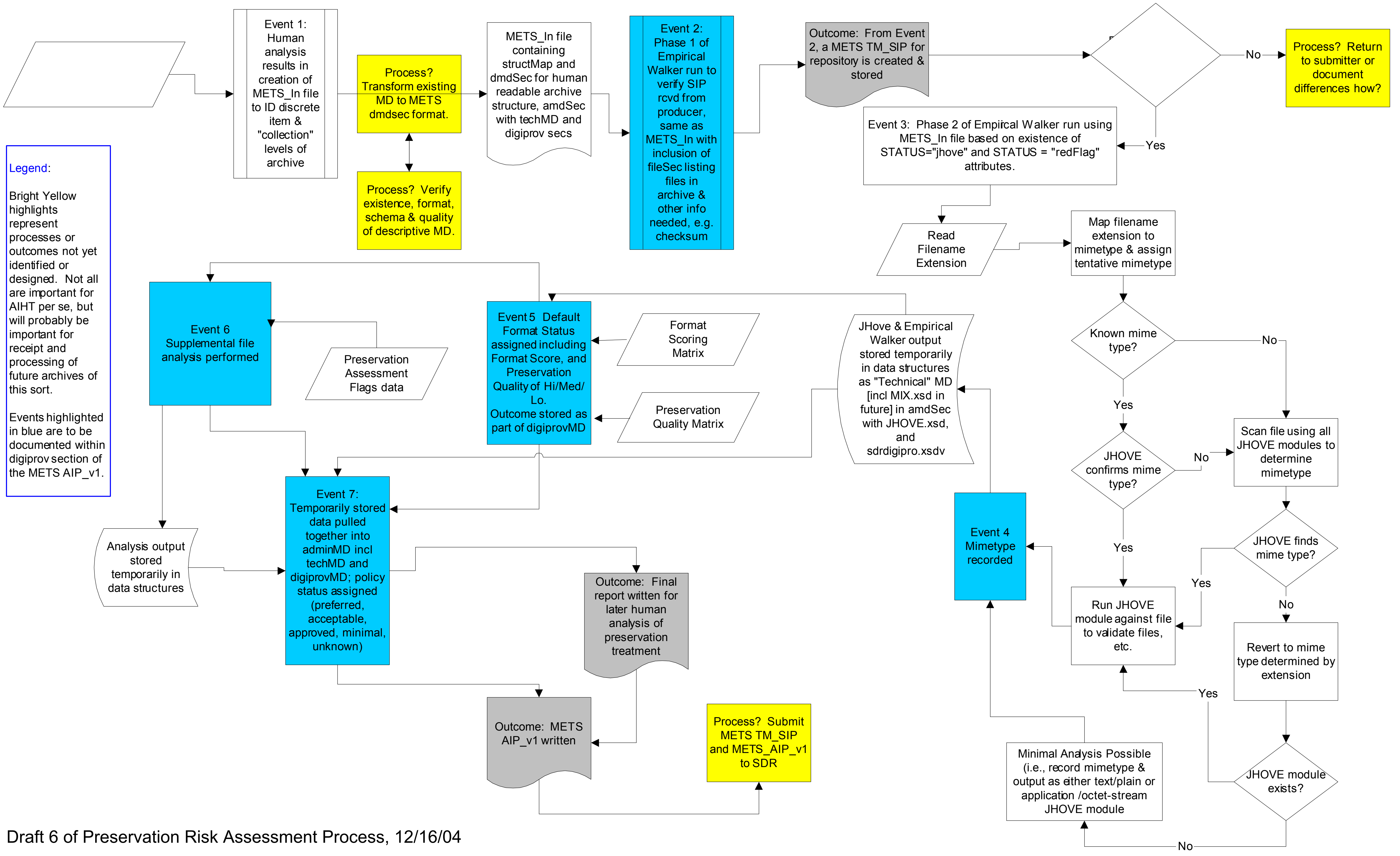
	Office Docs	**Audio**			**Video**				**Animation**
	Word, Excel, PowerPoint	WAVE (LPCM)	AIFF	MP3	MPEG	Real	QT	Windows Media	Flash
Adoption	+	+	+	+	+	+	+	+	+
Disclosure	-	+	+	+	+	-	-	-	-
Transparency	-	+	+	-	-	-	-	-	-
Self-Documentation	-	+	+	+	+	-	-	-	-
External Dependencies	-	+	+	+	+	-	-	-	-
Negative Count	4	0	0	1	1	4	4	4	4

The matrix has been formatted to fit on a single printed page.

12/16/2004

The following definitions for sustainability factors -- "factors that influence feasibility and cost of preserving content in the face of future change" -- have been excerpted from "Digital Formats: Factors for Sustainability, Functionality, and Quality", a presentation by Caroline Arms and Carl Fleischhauer, Office of Strategic Initiatives, Library of Congress, at the Digital Library Federation Forum in Fall 2003 (available online at: http://memory.loc.gov/ammem/techdocs/digform/ and http://www.diglib.org/forums/fall2003/fallforum03.htm#p1)

Disclosure	Degree to which complete specifications and tools for validating technical integrity exist and are accessible to those creating and sustaining digital content. A spectrum of disclosure levels can be observed for digital formats. What is most significant is not approval by a recognized standards body, but the existence of complete documentation. Preservation of content in a given digital format over the long term is not feasible without an understanding of how the information is represented (encoded) as bits and bytes in digital files.
Adoption	Degree to which the format is already used by the primary creators, disseminators, or users of information resources. This includes use as a master format, for delivery to end users, and as a means of interchange between systems. If a format is widely adopted, it is less likely to become obsolete rapidly, and tools for migration and emulation are more likely to emerge from industry without specific investment by archival institutions.
Transparency	Degree to which the digital representation is open to direct analysis with basic tools, such as human readability using a text-only editor. Digital formats in which the underlying information is represented simply and directly will be easier to migrate to new formats and more susceptible to digital archaeology; easier development of rendering software for new technical environments.
Self-documentation	Self-documenting digital objects contain basic descriptive, technical, and other administrative metadata. Self-documenting digital objects are likely to be easier to sustain over the long term and to transfer reliably from one archival system to another, including a successor system. LC wants to take advantage of the trend towards embedded metadata for business reasons. Some metadata will be extracted to support discovery and collection management.
External Dependencies	Degree to which a particular format depends on particular hardware, operating system, or software for rendering or use and the predicted complexity of dealing with those dependencies in future technical environments. Some interactive digital content is designed for use with specific hardware, such as a joystick. Scientific datasets built from sensor data may require specialized software for analysis and visualization. External dependencies will make content more difficult and costly to sustain than static content. The specialized software required by some scientific datasets may itself be very difficult to sustain.



Event 1: Human analysis results in creation of METS_In file to ID discrete item & "collection" levels of archive
Process? Transform existing MD to METS dmdsec format.
Process? Verify existence, format, schema & quality of descriptive MD.
METS_In file containing structMap and dmdSec for human readable archive structure, amdSec with techMD and digiprov secs
Event 2: Phase 1 of Empirical Walker run to verify SIP rcvd from producer, same as METS_In with inclusion of fileSec listing files in archive & other info needed, e.g. checksum
Outcome: From Event 2, a METS TM_SIP for repository is created & stored
No
Process? Return to submitter or document differences how?
Yes
Event 3: Phase 2 of Empirical Walker run using METS_In file based on existence of STATUS="jhove" and STATUS = "redFlag" attributes.
Read Filename Extension
Map filename extension to mimetype & assign tentative mimetype
Known mime type?
No
Yes
JHOVE confirms mime type?
No
Scan file using all JHOVE modules to determine mimetype
JHOVE finds mime type?
Yes
No
Revert to mime type determined by extension
JHOVE module exists?
Yes
No
Run JHOVE module against file to validate files, etc.
Event 4 Mimetype recorded
Minimal Analysis Possible (i.e., record mimetype & output as either text/plain or application /octet-stream JHOVE module
JHove & Empirical Walker output stored temporarily in data structures as "Technical" MD [incl MIX.xsd in future] in amdSec with JHOVE.xsd, and sdrdigipro.xsdv
Format Scoring Matrix
Preservation Quality Matrix
Event 5 Default Format Status assigned including Format Score, and Preservation Quality of Hi/Med/Lo. Outcome stored as part of digiprovMD
Event 6 Supplemental file analysis performed
Preservation Assessment Flags data
Analysis output stored temporarily in data structures
Event 7: Temporarily stored data pulled together into adminMD incl techMD and digiprovMD; policy status assigned (preferred, acceptable, approved, minimal, unknown)
Outcome: Final report written for later human analysis of preservation treatment
Outcome: METS AIP_v1 written
Process? Submit METS TM_SIP and METS_AIP_v1 to SDR
Legend:
Bright Yellow highlights represent processes or outcomes not yet identified or designed. Not all are important for AIHT per se, but will probably be important for receipt and processing of future archives of this sort.
Events highlighted in blue are to be documented within digiprov section of the METS AIP_v1.

Draft 6 of Preservation Risk Assessment Process, 12/16/04

PRESERVATION ASSESSMENT OF AIHT FILES BY SDR METHOD

Format Assessment				File Counts Determined by	
MIME Type	Score	Quality	Policy	File Extension	Empirical Walker
image/bmp	0	high	approved	28	
image/gif	2	medium	acceptable	1302	1342
image/jpeg	1	high	approved	12758	12764
image/png	1	high	approved	9	
image/tiff	0	high	preferred	1531	1537
text/xml	1	high	preferred	1	6
text/css	1	high	approved	35	
text/html	1	high	approved	16680	17003
text/plain	0	high	preferred	19767	19879
audio/x-ms-wma	4	low	minimal	15	
audio/x-aiff	0	high	approved	162	162
audio/x-mpeg	1	high	approved	39	
audio/x-wave	0	high	preferred	2016	2016
audio/x-realmedia	4	low	minimal	15	
video/quicktime	4	low	minimal	36	
video/x-msvideo	4	low	minimal	31	
video/x-ms-wmv	4	low	minimal	29	
video/x-mpeg	1	high	approved	4	
application/pdf	2	medium	preferred	1664	1664
application/octet-stream	5	low	minimal	47	1077
application/vnd.ms-word	4	low	minimal	78	
application/vnd.adobe-photoshop	4	low	minimal	208	
application/vnd.ms-powerpoint	4	low	minimal	20	
application/vnd.ms-excel	4	low	minimal	8	
application/x-msaccess	4	low	minimal	2	
application/x-shockwave-flash	4	low	minimal	57	
other	5	low	minimal	908	
total files				57450	57450

Empirical Walker Summary Information		
	File Count	Percentage
Score		
0	23594	41.1%
1	29773	51.8%
2	3006	5.2%
3	0	0.0%
4	0	0.0%
5	1077	1.9%
Quality		
high	53367	92.9%
medium	3006	5.2%
low	1077	1.9%
Policy		
preferred	25102	43.7%
approved	29929	52.1%
acceptable	1342	2.3%
minimal	0	0.0%
unknown	1077	1.9%

Empirical Walker Documentation *APPENDIX D*

Richard Anderson
Stanford Digital Repository
Digital Services Group
Stanford University Libraries and Academic Information Resources
April 25, 2005

Introduction

Empirical Walker (EW) is a program written in Java (using Borland JBuilder-X Developer edition). This program recursively traverses a file tree, calculates checksums, determines file format type, analyzes file contents, and performs preservation status assignment and risk assessment. Primary output is to a master METS xml file. Supplemental output is to individual metadata files, also in XML format.

This program utilizes API mechanisms for invoking the functionality of Harvard METS Toolkit, JHOVE, and XmlBeans Java classes. See the Dependencies section of this document for references and discussion of these libraries.

Empirical Walker was developed during Stanford's participation in the Library of Congress' *Archive Ingest and Handling Test* , a part of the *National Digital Information Infrastructure and Preservation Program.* http://www.digitalpreservation.gov

Objectives of Collection and File-level Analysis

What immediately follows is a more detailed overview of the analytical functions provided by the Empirical Walker.

The theoretical framework behind the preservation aspects of this analysis are discussed in other project documentation. See SDRPresAssessDescrip.pdf and supporting documentation written by Hannah Frost.

Transfer Manifest Generation

The transfer manifest is analogous to a digital bill of lading, and represents minimal metadata for a repository submission. It is required for confirmation of delivery of a particular group of digital material. The minimal data required for a transfer manifest is a list of the files being transmitted along with associated fixity information, such as a MD5 checksum. (One tool commonly used to generate such a manifest is the md5sum unix utility program. http://www.gnu.org/software/textutils/manual/textutils/html_node/textutils_21.html#SEC21)

Empirical Walker includes a MD5 checksum calculator that generates the checksum incrementally, avoiding the need to read the entire contents of the data file into memory at once.

The checksum information can be compared against the transfer manifest supplied by the originating entity. This is not always a straightforward operation, as the file paths used by the originator may be expressed as absolute, rather than relative paths. And the sort sequence of the filenames cannot be assumed to be in the same order as in the list generated locally.

In the AIHT project, Empirical Walker was used to generate a local manifest in METS format, then a separate SAX-based parser was used to extract the path and checksum information from each item element into a flat text file. The originator's transfer manifest was also transformed into a flat file. One or both of these files were edited so that the filepath were expressed as equivalent relative paths. Next both files were sorted (using unix sort) and a diff was performed , which demonstrated that the two files had the same basic content.

File Format Identification or Verification

Initially, a file's MIME type is tentatively assigned based on the filename extension to MIME type mapping specified in the extension-mimetype.txt configuration file. JHOVE is then used to confirm this tentative identification.

If no tentative MIME type could be assigned, or if JHOVE fails to confirm the tentative MIME type, then JHOVE is used in a scanning mode in an attempt to discover the format type. If this scan returns a MIME type other than text/plain or application/octet-stream (the most generic types), then the JHOVE results are used for format assignment.

If a JHOVE scan returns a MIME type of text/plain or application/octet-stream, but the filename extension to MIME type mapping yields a more specific assignment, then the filename extension mapping is used for format assignment.

If no more specific format assignment is possible, then the JHOVE text/plain or application/octet-stream results are used for format assignment.

Regardless of the outcome, the output from parsing of the file's contents (using the JHOVE module for the assigned format type) is written to file and also stored temporarily in memory for later analysis.

Format Status Assignment

The format type assigned in the previous step is used by Empirical Walker to assign default values for *format score*, *preservation quality*, and *policy status*. Collectively these values are referred to as "*format status*" The configuration file sdrFormatStatusDefaults.xml is used as the source of these default values. .

The *format score* (ranging from 0-5) is calculated based on a format scoring matrix. In this matrix each format type is assigned true/false values for adoption, disclosure, transparency, self-documentation, and freedom from external dependencies. The format score is the count of false values for these attributes. Thus the lower the score, the better the prospects for long term preservability.

Associated with each *format score* is a corresponding value of *preservation quality*, which can have a value of High, Medium, or Low. P*reservation quality* provides a verbal and less granular relative measure of format status.

Format Score	**Preservation Quality**	**Policy Status**
0-1	High	Approved
2-3	Medium	Acceptable
4	Low	Minimal
5	Low	Unknown

Based on a combination of format score and local format preferences, a value for *policy status* is also assigned to the file. There are five *policy status* levels: preferred, approved, acceptable, minimal, unknown. A class of preferred formats (as opposed simply to "approved") is necessary for business reasons because it focuses attention on the specific formats for which the repository is committed to providing full support services. Not all formats with a Format Score of zero automatically earn the status of a preferred format; the capacity of a format for enhanced quality and functionality must also be factored in. Similarly, a Format Score of zero is not required to earn the "preferred" status; a format that is both highly suited to a specific purpose within the digital library context *and* free of risk factors does not always exist.

Plain text	ASCII
	UTF-8
Marked-up text	XML 1.0
Image	TIFF 5.0 and above (uncompressed)
Page-Viewer	PDF* (any version)
Audio	WAVE (linear pulse code modulation)
Video	*TBD*

Preservation Risk Assessment

Preservation risk assessment is performed by comparing JHOVE's file parsing output against a set of rules documented in PreservationAssessmentFlags.xml.

This analysis has two primary goals. The first goal is to examine the output of the file validation process in search of:

a. invalid or not well-formed files;
b. technical characteristics which could pose potential complications in preservation activities (such as a TIFF that is compressed);
c. technical characteristics which indicate that the file has reduced preservation-risk associated with its default format status (such as a PDF which meets the anticipated PDF-A profile).

The second goal of supplemental analysis is to identify those files that may benefit from transformation or normalization. The Empirical Walker determines if the extension represents a MIME type or format that has particular characteristics at risk of loss in future digital preservation activities and therefore could be transformed pre-ingest by means of reformatting or normalization in order to improve its preservation prospects. For instance, a Photoshop document (*.psd), which has a low format status (4), could be reformatted as a PNG or TIFF with little to no loss, and thus earn a higher preservation assessment as a result. Similarly a MS Word document could be reformatted as plain text for enhanced preservation of the textual content over time.

For the AIHT project we elected to perform transformations based on the analysis of JHOVE analysis of HTML files. A file was flagged for transformation if JHOVE identified it as being well formed and valid. We also flagged files where a JHOVE error message was equal to any of the following values:

- Unrecognized or missing DOCTYPE declaration; validation continuing as HTML 3.2
- Close tag without matching open tag
- Tag illegal in context
- Parse error
- Undefined attribute for element
- Unknown tag
- TokenMgrError

Inputs and Outputs

Runtime Parameters

Command-line arguments to the program specify the directories in which the configuration files, data object files, and session-specific input-output files are located:

- -c {config file directory}
- -o {data object directory}
- -s {session file directory}

Config File Directory

The config file directory is the location where the following configuration files should be located:

- extension-mimetype.txt = file containing filename extension to MIME type mappings
- jhove.conf = JHOVE configuration info
- sdrFormatStatusDefaults.xml = defaults data for use in format status assignments
- PreservationAssessmentFlags.xml = preservation assessment 'red flags'

If the -c option is not specified, then the value defaults to {User Home Directory}\empiricalwalker.

Data Object Directory

The data object directory is the location where the files or collections of files to be analyzed are located. If the -o option is not specified, then the value defaults to the current working directory.

The files and/or directories analyzed are assumed to be located in or below a single base directory. File references within the METS output document are therefore expressed as paths that are relative to that base directory (rather than being expressed as absolute paths).

Session File Directory

The session file directory is the location where the following METS input and output files are located:

- MetsIn.xml = input data file containing specification of how to perform analysis
- MetsOut.xml = output data file containing METS document that records metadata including results of the analysis

- DigiprovOut-n.xml = A file containing a summary of format identification, format status assignment, and preservation risk analysis results for collection number n.
- METADATA directory = top level of a directory structure that parallels the directory structure of the collection. This hierarchy contains descriptive and administrative metadata files that are referenced from within the METS document.

If the session file directory is not specified, then the value defaults to the same directory as the data object directory.

Configuration files

extension-mimetype.txt

This file contains filename extension to MIME type mappings, expressed in the name=value format commonly used in a properties file. Here is an excerpt from this file:

```
aiff=audio/x-aiff
asp=text/html
avi=video/x-msvideo
bmp=image/bmp
css=text/css
doc=application/vnd.ms-word
eml=text/plain
gif=image/gif
htm=text/html
html=text/html
jpeg=image/jpeg
jpg=image/jpeg
```

The mapping used was derived from a table supplied by Artesia as part of its TEAMS application, and expanded or modified based on consultation with a variety of format registries. Such registries include:

- DLF Global File Format Registry: http://hul.harvard.edu/gdfr/
- PRONOM: http://www.nationalarchives.gov.uk/PRONOM/
- The file extension source: http://filext.com/
- The programmers file format collection: http://www.wotsit.org/
- "Every File Format in the World": http://whatis.techtarget.com/fileFormatA/

The MIME type thus determined is used in combination with JHOVE to identify and/or verify the format type.

Note that the current version of EW uses a mapping that reflects our judgment of the most commonly associated MIME type for a given filename extension. Future versions of the program should be enhanced to support multiple mappings, and this table will likely be incorporated into the sdrFormatStatusDefaults.xml file.

If filenames include extensions such as gif?1000245094, EW will ignore the '?' and the string that follows, using the extension of gif to determine MIME type.

jhove.conf

This is the same configuration file that would be used for standalone use of JHOVE. Consult JHOVE documentation for the structure and meaning of this file's contents. (See JHOVE references in Dependencies section of this document).

Note that the sequence of JHOVE modules listed in this file can affects the outcome of a file format identification scan. "The order in which format modules are defined is important; when performing a format identification operation, JHOVE will search for a matching module in the order in which the modules are defined in the configuration file. In general, the modules for more generic formats should come later in the list."

sdrFormatStatusDefaults.xml

Default data used in format status assignment. See "Format Status Assignment" discussion above. See also FileFormatStatus.xsd in the XML schema section below.

PreservationAssessmentFlags.xml

Records the criteria used for preservation risk assessment. See "Preservation Risk Assessment" section above. See also PreservationAssessmentFlags.xsd in the XML schema section below.

Session Input File

MetsIn.xml

This input data file contains specification of which objects to analyze and how to perform the analysis.

The mets and metsHdr element of this document are copied into the METS output file.

The structMap element is used to specify which files or directories to analyze. File pathnames are expressed as relative to the base object directory as specified with the –o option flag when invoking Empirical Walker.

The dmdSec section(s) are used to specify what type of descriptive metadata to include in the output. If the STATUS attribute contains the string "mods", then a dmdSec element will be created in the METS output that describes the directory hierarchy for the collection. If the STATUS attribute contains the string "raw", then for each data content file a dmdSec element will be created that points to an external XML file containing raw descriptive metadata.

The amdSec section(s) are used to specify what type of administrative metadata to include in the output. If this section contains a techMD element with a STATUS="jhove" attribute, then for each data content file a amdSec/techMD element will be created that points to an external XML file containing jhove output. If this section contains a digiprovMD element with a STATUS="redflag" attribute, then for each data content file a amdSec/digiprovMD element will be created that points to an external XML file containing the outcome of preservation status assignment and risk assessment. Note that the redflag analysis is dependent on the JHOVE analysis having first been performed.

ID/IDREF identifiers are used in the dmdSec, amdSec, and structMap elements to specify which data content files (or collections) require which types of metadata outputs.

For examples of output, see the "Output Files" section below.

Example MetsIn.xml:
```
<?xml version="1.0" encoding="UTF-8"?>
<mets xmlns="http://www.loc.gov/METS/" xmlns:xlink="http://www.w3.org/TR/xlink"
xmlns:mods="http://www.loc.gov/mods/v3" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.loc.gov/METS/://www.loc.gov/standards/mets/mets.xsd
http://www.loc.gov/mods/v3 http://www.loc.gov/standards/mods/v3/mods-3-0.xsd" LABEL="AIHTArchive" TYPE="TAR
Archive" PROFILE="">
      <metsHdr ID="AIHT_01" CREATEDATE="2004-09-02T09:34:31" RECORDSTATUS="Draft">
            <agent ID="MD_rna" ROLE="CREATOR" OTHERROLE="" TYPE="ORGANIZATION" OTHERTYPE="">
                  <name>"Richard Anderson"</name>
                  <note>"Master Digital Archive record to document AIHT project files and preservation
                  analysis</note>
            </agent>
      </metsHdr>
      <dmdSec ID="DMD_1" STATUS="mods,raw">
      </dmdSec>
      <amdSec ID="AMD_1">
            <techMD ID="JHOVE_1" STATUS="jhove"/>
            <digiprovMD ID="REDFLAG_1" STATUS="redflag"/>
      </amdSec>
```

```
		<structMap>
			<div TYPE="root" LABEL="AIHT Archive" ORDERLABEL="">
				<div TYPE="tarfile" LABEL="AIHT Archive container file" ORDERLABEL="./aiht-
				content.tar.gz"/>
				<div TYPE="contents" LABEL="Files contained in the AIHT Archive" ORDERLABEL="./AIHT-
				CONTENT" DMDID="DMD_1" ADMID="AMD_1"/>
			</div>
		</structMap>
	</mets>
```

Output Files

MetsOut.xml

The MetsOut.xml file is the master METS instance document that records or points to descriptive, technical, and structural metadata including results of file analysis. Some of this metadata is stored in external files in the METADATA directory discussed below. Links to these files are expressed as xlink pointers within mdRef elements.

- *Filenames*

The files and/or directories analyzed are assumed to be located in or below a single base directory. File references within the METS output document are therefore expressed as paths that are relative to that base directory (rather than being expressed as absolute paths).

- *ID Numbers*

ID numbers used to reference files or file level metadata begin with a prefix designating the element type (e.g. DIV_, FILE_, ADM_, & DMD_) and end with a segmented decimal number indicating the location of the file in the directory hierarchy. For example: ID= "DIV_4.2.29" indicates that we are referencing an object that is three levels deep in the hierarchy. The top level folder's DIV has ID="DIV_4", the second folder on the level below it has ID= "DIV_4.2", and the 29th file on the next level down has ID="DIV_4.2.29".

- *Directory Hierarchy expressed in MODS*

The METS output document can optionally include a single dmdSec using MODS elements to specify a collection directory hierarchy. This descriptive information provides some human readable, intellectual understanding of about the collection level structure. At present this automatically generated descriptive metadata is very sparse, containing only directory names.

Example dmdSec:

```
<dmdSec ID="DMD_2" CREATED="2004-12-27T07:10:12">
   <mdWrap MDTYPE="MODS" LABEL="MODS record for collection(s)">
      <xmlData>
         <mods:mods ID="MODS_2">
            <mods:titleInfo>
               <mods:title>AIHT-CONTENT</mods:title>
            </mods:titleInfo>
            <mods:relatedItem  ID="MODS_2.01" type="constituent">
               <mods:titleInfo>
                  <mods:title>CONTRIBUTORS</mods:title>
               </mods:titleInfo>
               <mods:relatedItem  ID="MODS_2.01.002" type="constituent">
                  <mods:titleInfo>
                     <mods:title>1199_photos</mods:title>
                  </mods:titleInfo>
                  <mods:relatedItem  ID="MODS_2.01.002.2" type="constituent">
                     <mods:titleInfo>
                        <mods:title>wtc_web</mods:title>
                     </mods:titleInfo>
                     …
```

- *Raw Descriptive Metadata dmdSec Elements*

The METS output document can optionally include a sequence of "raw metadata" dmdSec elements for all files in a collection, with mdRef elements pointing to external raw metadata files, which should already exist in the METADATA directory described below.

These external files are assumed to have been created by a process independent of the operation of Empirical Walker. For example, the collection of data used for the AIHT project included metadata provide in the form of database tables. This metadata was transformed using MS SQL Server, Altova XMLSpy and MapForce, and a Java program into a series of "raw" metadata files which were placed in the METADATA directory.

Example dmdSec:

```
      <dmdSec ID="DMD_2.01.002.2.01" CREATED="2004-12-27T07:10:12">
            <mdRef ID="RAW_2.01.002.2.01" LOCTYPE="URL" xlink:type="simple" xlink:href="./METADATA/AIHT-
            CONTENT/CONTRIBUTORS/1199_photos/wtc_web/raw_2.01.002.2.01.xml" MDTYPE="OTHER" OTHERMDTYPE="From
            relational database" LABEL="./AIHT-CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg"/>
       </dmdSec>
```

Note that a xlink:href attribute is used to reference the external raw metadata file and that the LABEL attribute is used to specify the relative path to the data object file within the collection

- *Technical Metadata amdSec Elements*

Empirical Walker can optionally be used to generate JHOVE and Digital Providence XML files for each file in a collection. In this case the METS output document will include a sequence of amdSec elements, each one containing a techMD/mdRef element that points to the JHOVE output and a digiprovMD/mdRef element that points to the preservation risk assessment output.

Example admSec:

```
<amdSec ID="AMD_2.01.002.2.01">
      <techMD ID="TECH_2.01.002.2.01" CREATED="2004-12-27T07:10:12">
            <mdRef ID="JHOVE_2.01.002.2.01" LOCTYPE="URL" xlink:type="simple" xlink:href="./METADATA/AIHT-
            CONTENT/CONTRIBUTORS/1199_photos/wtc_web/jhove_2.01.002.2.01.xml" MDTYPE="OTHER"
            OTHERMDTYPE="jhove" LABEL="./AIHT-CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg"/>
      </techMD>
      <digiprovMD ID="DIGIPROV_2.01.002.2.01" CREATED="2004-12-27T07:10:12">
            <mdRef LOCTYPE="URL" xlink:type="simple" xlink:href="./METADATA/AIHT-
            CONTENT/CONTRIBUTORS/1199_photos/wtc_web/digiprov_2.01.002.2.01.xml" MDTYPE="OTHER"
            OTHERMDTYPE="sdrDigiprov" LABEL="./AIHT-CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg"/>
      </digiprovMD>
</amdSec>
```

Note that xlink:href attributes are used to reference external metadata files and that the LABEL attribute is used to specify the relative path to the data object file within the collection.

One possible enhancement to EW would allow the user to use mdWrap elements instead of mdRef elements, so that the technical metadata could be written inside the METS document instead of in an external file.

- *fileSec Section*

The fileSec element contains file sizes, MIME types, and MD5 checksums for all files in a collection. The data is presented in a flat layout and can be treated as the transfer manifest for the collection. This transfer manifest can be compared against the manifest supplied by the originator to verify that no file corruption has occurred during the transfer.

Example fileGrp and file elements:

```
<fileGrp ID="FG_2" … USE="Files contained in the AIHT Archive">
```

```
          <file ID="FILE_2.01.002.2.01" MIMETYPE="image/jpeg" SEQ="3" SIZE="209046" CREATED="2004-01-
          27T09:59:38" CHECKSUM="486089708ab81ecd3b2401f4fb37ed39" CHECKSUMTYPE="MD5">
                <FLocat LOCTYPE="URL" xlink:type="simple" xlink:href="./AIHT-
                CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg"/>
          </file>
          …
     </fileGrp>
```

Note that a xlink:href attribute is used to reference the object file.

There is a top level fileGrp for each file or directory in the base-level data object directory.

- *Physical structMap section*

The structMap element contains a hierarchical representation of the container files and all files contained within them.

Example:

```
     <div ID="DIV_2" ORDER="2" ORDERLABEL="./AIHT-CONTENT" LABEL="AIHT-CONTENT" DMDID="DMD_2" TYPE="folder
     containing 2106 folders, 57450 files, 12375311120 bytes">
          …
          <div ID="DIV_2.01.002.2" ORDER="2" ORDERLABEL="./AIHT-CONTENT/CONTRIBUTORS/1199_photos/wtc_web"
          LABEL="wtc_web" TYPE="folder containing 0 folders, 27 files, 5211521 bytes">
                <div ID="DIV_2.01.002.2.01" ORDER="1" ORDERLABEL="./AIHT-
                CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg" LABEL="WTC1.jpg" DMDID="DMD_2.01.002.2.01"
                ADMID="AMD_2.01.002.2.01" TYPE="file">
                      <fptr FILEID="FILE_2.01.002.2.01"/>
          </div>
          …
     </div>
```

Note that the ORDERLABEL attribute is used to specify the relative path of the directory or file. Note also that DMDID, ADMID, and FILEID IDREF attributes are used to reference dmdSec, amdSec, and file elements elsewhere in the METS document.

There is a top level fileGrp for each file or directory in the base-level data object directory.

METADATA directory

Some file-level descriptive and administrative metadata, such as JHOVE output or the details of digital providence (including format identification, format status assignment, and preservation risk analysis) are recorded in separate XML files, that are referenced from within the master METS document using xlink notation within mdRef elements.

These files are stored in a METADATA directory structure that mirrors the directory hierarchy of the data object base directory and its children. All files in the METADATA directory have filenames that begin with the prefix raw, jhove, or digiprov followed by a unique numerical sequence that relates back to the ID numbers used to represent data object files in the master METS document. Example filenames are raw_2.3.45, jhove_2.3.45, and digiprov_2.3.45 .

This design helps to keep down the size of the METS document and provides more granularity in cases where edits or replacement of the metadata for a single file may be needed. This structure will also allow us to use the more efficient access mechanisms provided by the filesystem instead of the slower access methods of current generation XML tools.

DigiProv Output files

The sdrDigiprov.xsd schema provides documentation for the output recorded in digiprov_n.n.n files.

The first part of the XML instance will document the sequence of steps involved in analysis. Each digiprov event has a eventIdentifier element, with these possible values:

- Manifest_Check = The submitted collection was traversed and a METS SIP was generated and compared against the provider's transfer manifest. This eventIdentifier is not currently used.
- Format_Identification = Identification of the file format was determined by a combination of filename extension lookup and scanning of the file contents.
- Status_Assignment = Based on file format, initial assignment of format status has been made
- Risk_Assessment = Output of the JHOVE (or other) file parsing procedure has been analyzed for preservation risks.

I probably should have used eventType instead of eventIdentifier for these coded values, so this may change in a future version of EW.

The Format_Identification event is the analysis step where the format's MIME type and format name are determined, as specified in the "File Format Identification or Verification" section above. The results of this analysis are record in the eventOutcome as well as in other metadata elements in the Object section of the output file. Code eventOutcome values for this event are:

- Format_Verified = JHOVE agrees with filename extension
- Format_FromExtension = No JHOVE module for this MIME type
- Format_Mismatch = JHOVE and filename extension suggest different MIME types

- Format_Unknown = File format is unknown

The Status_Assignment event is the analysis step where the initial default formatScore, preservationQuality, and policyValue are determined (based on format type) and recorded in the eventOutcome as well as in the formatStatus subsection of the object element. (see "Format Status Assignment" section above.) Coded eventOutcome values for this event are:

- StatusPolicy_Preferred
- StatusPolicy_Approved
- StatusPolicy_Acceptable
- StatusPolicy_Minimal
- StatusPolicy_Unknown

the Risk_Assessment event is where the red flag/transformation analysis outcome is recorded. (see "Preservation Risk Assessment" section above.) The coded eventOutcome values for this event are:

- RiskAssessment_None = No matching flags were raised
- RiskAssessment_Warning = flag was raised, but no specific action recommended
- RiskAssessment_DowngradeStatusPolicy = flag was raised, and policy status adjusted as specified
- RiskAssessment_TransformFormat = flag was raised, and transformation recommended

If multiple flags were raised, then the most severe of the flags is used to determine the eventOutcome. Additional details are recorded in the eventOutcomeDetail element.

Example digiprov_n.xml file:

```
<?xml version="1.0" encoding="UTF-8"?>
<dig:sdrDigiprov xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://sdr.sulair.stanford.edu/fileFormatStatus"
xmlns:dig="http://sdr.sulair.stanford.edu/digiprov">
    <dig:event>
        <dig:eventIdentifier eventIdentifierValue="Format_Identification"
eventIdentifierScheme="SDR_Event_Plan_v1"/>
        <dig:eventOutcome>Format_Verified</dig:eventOutcome>
        <dig:eventOutcomeDetail>ext=jpg|map=image/jpeg|scan=image/jpeg</dig:eventOutcomeDetail>
        <dig:eventDateTime>2005-03-17T19:47:45.705-08:00</dig:eventDateTime>
    </dig:event>
    <dig:event>
        <dig:eventIdentifier eventIdentifierValue="Status_Assignment"
eventIdentifierScheme="SDR_Event_Plan_v1"/>
        <dig:eventOutcome>StatusPolicy_Approved</dig:eventOutcome>
        <dig:eventOutcomeDetail>formatScore=1 preservationQuality=StatusQuality_High
policyValue=StatusPolicy_Approved</dig:eventOutcomeDetail>
        <dig:eventDateTime>2005-03-17T19:47:45.706-08:00</dig:eventDateTime>
    </dig:event>
    <dig:event>
        <dig:eventIdentifier eventIdentifierValue="Risk_Assessment"
eventIdentifierScheme="SDR_Event_Plan_v1"/>
        <dig:eventOutcome>RiskAssessment_TransformFormat</dig:eventOutcome>
        <dig:eventOutcomeDetail># JPEG_ProgressiveEncoding</dig:eventOutcomeDetail>
        <dig:eventDateTime>2005-03-17T19:47:45.707-08:00</dig:eventDateTime>
    </dig:event>
    <dig:agent>
        <dig:agentIdentifier agentIdentifierValue="Empirical_Walker_v1"
agentIdentifierScheme="SDR_SWAgent_v1"/>
    </dig:agent>
    <dig:object xmlns="http://sdr.sulair.stanford.edu/preservationAssessmentFlags">
        <dig:objectIdentifier objectIdentifierValue="./AIHT-CONTENT/CONTRIBUTORS/alfonso_gatto/4-
capt.1000341922attacks_space_image_sio101.jpg" objectIdentifierScheme="Collection_Relative_Path"/>
        <dig:objectCharacteristics>
            <dig:compositionLevel>0</dig:compositionLevel>
            <dig:fixity checkValue="a4c976aee168f8e837a6f13091fe7b8c" checkMethod="MD5"/>
            <dig:size>8772</dig:size>
        </dig:objectCharacteristics>
        <dig:format>
            <dig:formatIdentifier formatVersion="1.01" formatName="JPEG" mimeType="image/jpeg"/>
            <dig:formatVerification mimeTypeFromScan="image/jpeg" mimeTypeFromExtension="image/jpeg"
filenameExtension="jpg" verificationOutcome="Format_Verified"/>
        </dig:format>
        <dig:formatStatus>
            <fil:scoringFactors adoption="true" disclosure="true" transparency="true"
selfDocumentation="false" externalDependencies="true"
xmlns:fil="http://sdr.sulair.stanford.edu/fileFormatStatus"/>
            <fil:formatScore scoringValue="1" scoringSchemeIdentifier="SDR_PR_v1"
scoringScheme="SDR_Format_Scoring_Matrix_v1" xmlns:fil="http://sdr.sulair.stanford.edu/fileFormatStatus"/>
            <fil:preservationQuality qualityValue="StatusQuality_High" qualitySchemeIdentifier="SDR_PQ_v1"
qualityScheme="SDR_Pres_Quality_v1" xmlns:fil="http://sdr.sulair.stanford.edu/fileFormatStatus"/>
            <fil:policyStatus policyValue="StatusPolicy_Approved" policySchemeIdentifier="SDR_PPS_v1"
policyScheme="SDR_Pres_Policy_Statement_v1" xmlns:fil="http://sdr.sulair.stanford.edu/fileFormatStatus"/>
        </dig:formatStatus>
        <dig:preservationRisk>
            <dig:flagRaised flagType="JPEG_ProgressiveEncoding">
```

```
                <dig:test feature="property" path="JPEGMetadata:Images:Image:Scans" compare="any value"
datatype="integer"/>
                <dig:valuesFound>
                    <value>1</value>
                </dig:valuesFound>
                <dig:recommendation action="RiskAssessment_TransformFormat">
                    <transformation newFormatName="JPEG2000" transformationEffects="Preserves image data in
a &quot;standard&quot; format expected to gain in preservation quality, and policy status, as adoption
increases; benefits of progressive encoding may not be retained."/>
                    <transformation newFormatName="JPEG" transformationEffects="Image data retained in a
format closer to de facto baseline; greater software compatibiilty; benefits of progressive encoding will
not be retained in the conversion."/>
                </dig:recommendation>
            </dig:flagRaised>
        </dig:preservationRisk>
    </dig:object>
</dig:sdrDigiprov>
```

DigiprovSummaryOut-n.xml

Empirical Walker also creates a summary output file that contains a breakdown of the format identification, format status assignment, and preservation risk assessment outcomes for the entire collection. This report also lists any problem files found and any files flagged for transformation. The structure of this summary is specified in the sdrDigiprovSummary root element of SdrDigiprov.xsd

Condensed Example of DigiprovOut-n.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<dig:sdrDigiprovSummary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://sdr.sulair.stanford.edu/preservationAssessmentFlags"
xmlns:dig="http://sdr.sulair.stanford.edu/digiprov">
    <dig:collection>
        <dig:collectionName>Files contained in the archive</dig:collectionName>
        <dig:baseDirectory>/se7/aiht/gmu/./AIHT-CONTENT</dig:baseDirectory>
        <dig:fileCount>57450</dig:fileCount>
        <dig:analysisDate>2005-03-17T19:45:14.176-08:00</dig:analysisDate>
    </dig:collection>
    <dig:formats>
        <dig:format formatCount="1076" formatName="BYTESTREAM" mimeType="application/octet-stream"/>
        <dig:format formatCount="1664" formatName="PDF" mimeType="application/pdf"/>
        <dig:format formatCount="5" formatName="XML" mimeType="application/xml"/>
        <dig:format formatCount="1076" formatName="BYTESTREAM" mimeType="application/octet-stream"/>
        <dig:format formatCount="1664" formatName="PDF" mimeType="application/pdf"/>
        <dig:format formatCount="5" formatName="XML" mimeType="application/xml"/>
        <dig:format formatCount="162" formatName="AIFF" mimeType="audio/x-aiff"/>
        <dig:format formatCount="2016" formatName="WAVE" mimeType="audio/x-wav"/>
        <dig:format formatCount="1342" formatName="GIF" mimeType="image/gif"/>
        <dig:format formatCount="12764" formatName="JPEG" mimeType="image/jpeg"/>
        <dig:format formatCount="1537" formatName="TIFF" mimeType="image/tiff"/>
        <dig:format formatCount="1" formatName="BYTESTREAM" mimeType="text/html"/>
        <dig:format formatCount="17003" formatName="HTML" mimeType="text/html"/>
        <dig:format formatCount="1" formatName="XML" mimeType="text/html"/>
        <dig:format formatCount="19879" formatName="ASCII" mimeType="text/plain; charset=US-ASCII"/>
    </dig:formats>
    <dig:formatVerificationOutcomes>
        <dig:verificationOutcome outcomeCount="38235" verificationOutcome="Format_Verified"/>
        <dig:verificationOutcome outcomeCount="17164" verificationOutcome="Format_FromExtension"/>
        <dig:verificationOutcome outcomeCount="975" verificationOutcome="Format_Mismatch"/>
        <dig:verificationOutcome outcomeCount="1076" verificationOutcome="Format_Unknown"/>
    </dig:formatVerificationOutcomes>
    <dig:formatScores>
        <dig:formatScore scoreCount="23594" scoreValue="0"/>
        <dig:formatScore scoreCount="29773" scoreValue="1"/>
        <dig:formatScore scoreCount="3006" scoreValue="2"/>
        <dig:formatScore scoreCount="1077" scoreValue="5"/>
    </dig:formatScores>
    <dig:preservationQualities>
        <dig:preservationQuality qualityCount="53367" qualityValue="StatusQuality_High"/>
        <dig:preservationQuality qualityCount="3006" qualityValue="StatusQuality_Medium"/>
        <dig:preservationQuality qualityCount="1077" qualityValue="StatusQuality_Low"/>
    </dig:preservationQualities>
    <dig:preservationPolicyOutcomes>
        <dig:policyOutcome policyCount="25102" policyValue="StatusPolicy_Preferred"/>
        <dig:policyOutcome policyCount="29929" policyValue="StatusPolicy_Approved"/>
        <dig:policyOutcome policyCount="1342" policyValue="StatusPolicy_Acceptable"/>
        <dig:policyOutcome policyCount="0" policyValue="StatusPolicy_Minimal"/>
        <dig:policyOutcome policyCount="1077" policyValue="StatusPolicy_Unknown"/>
        </dig:policyOutcome>
    </dig:preservationPolicyOutcomes>
    <dig:preservationRisk>
        <dig:format formatName="AIFF">
            <dig:flagRaised>
                <dig:flag flagID="AIFF_CompressionType">
                    <test feature="property" path="AIFFMetadata:CompressionType" compare="any value"/>
                    <reason>In the interest of tracking any compression type which may hinder future
accessibility.</reason>
                    <recommendation action="RiskAssessment_Warning"/>
                </dig:flag>
                <dig:fileList fileCount="n">
                  ...
                </dig:fileList>
            </dig:flagRaised>
        <dig:format formatName="HTML">
            <dig:flagRaised>
                <dig:flag flagID="HTML_SyntaxError">
                    <test feature="errorMessage" compare="equals"/>
                    <values>
                        <value>Unrecognized or missing DOCTYPE declaration; validation continuing as HTML
3.2</value>
                        <value>Close tag without matching open tag</value>
                        <value>Tag illegal in context</value>
                    </values>
                    <reason>Improper syntax</reason>
                    <recommendation action="RiskAssessment_TransformFormat">
                        <transformation newFormatName="XHTML" transformationEffects="Corrected syntax and
flexibility afforded by XHTML will facilitate future access and interoperability."/>
                    </recommendation>
                </dig:flag>
                <dig:fileList  fileCount="n">
                  ...
                </dig:fileList>
            </dig:flagRaised>
        <dig:format formatName="PDF">
            <dig:flagRaised>
                <dig:flag flagID="PDF_EmbeddedImages">
                    <test feature="property" path="PDFMetadata:Images" compare="any value"
datatype="string"/>
                    <reason>Embedded images might be of interest (because they can be extracted and
preserved as other MIMEtypes which are more open). Note that probably all DL1 PDFs have image metadata
recorded here that is not of concern at this point.</reason>
                    <recommendation action="RiskAssessment_Warning"/>
                </dig:flag>
                <dig:fileList  fileCount="n">
                    ...
                </dig:fileList>
            </dig:flagRaised>
            </dig:preservationRisk>
        </dig:sdrDigiprovSummary>
```

XML Schema

METS and MODS

METS and MODS schemas are used in the master METS metadata file described above.
The schemas are documented at:
http://www.loc.gov/standards/mets/
http://www.loc.gov/standards/mods/

One problem we have encountered with the METS schema is that the current version of the schema cannot be referenced by a URL that includes the version number. This is being discussed by the METS Editorial Board and will hopefully be remedied in a future version.

JHOVE

The JHOVE file analysis tool can output its results into an XML file if the XmlHandler is chosen as the OutputHandler.

The schema for this output is located at:
http://hul.harvard.edu/ois/xml/xsd/jhove/1.2/jhove.xsd

FileFormat.xsd

FileFormat.xsd contains an enumeration of format names, a complex type for specifying format (name, MIME type, version), and a structure to be used in future for filename extension to mimetype mapping.

FileFormatStatus.xsd

FileFormatStatus.xsd contains the structures used for recording format status assignments (scoring factors, format score value, preservation quality, and policy status. There is also a structure for recording the default values used as input to the analysis process. (see sdrFormatStatusDefaults.xml)

PreservationAssessmentFlags.xsd

PreservationAssessmentFlags.xsd contains the structures used to record the preservation risk "red flags" that are to be searched for in the JHOVE output. The red flag data used for input can be found in PreservationAssessmentFlags.xml

SdrDigiprov.xsd

SdrDigiprov.xsd contains the structures used to document digital provenance (digiprov) information. Instances of this schema record events, agent, and object data. Events include format identification, preservation status assignment, and preservation risk assessment. The agent is the Empirical Walker program. Object information includes file characteristics, format identifiers, and format status

There is currently no comprehensive XML schema for recording preservation metadata. The work of the Preservation Metadata Workgroup (PREMIS) was therefore used to inform our design of a tentative XML schema to be used for documenting the results of our preservation risk assessment. Eventually we would expect to use the official PREMIS schema once it has been created. For more on PREMIS, see: http://www.oclc.org/research/projects/pmwg/

SdrDigiprov.xsd defines two output document root elements.

- Root element sdrDigiprov contains the detailed digiprov information from the analysis of a single file.
- Root element sdrDigiprovSummary contains a summary of digiprov information from the analysis of all files in the collection that was analyzed.

The current versions of the above 4 schema have been modified to make them more cohesive as a unit. The schema is more modular, and the "import" keyword is used to specify cross-references between schema.

- The format identification types defined in FileFormat.xsd are used in all the other three schema.
- The PolicyValueType defined in FileFormatStatus.xsd is used in PreservationAssessmentFlags.xsd for the newPolicyValue attribute of the recommendation.
- The FormatStatusType defined in FileFormatStatus.xsd is used in SdrDigiprov.xsd to specify the format status of a file
- The FlagType (and its components) defined in PreservationAssessmentFlags.xsd is used in SdrDigiprov.xsd to specify the results of the preservation risk assessment for a file.

Also note that the following union type has been created to hold the coded values of eventOutcome:

```
<xs:simpleType name="EventOutcomeValueType">
      <xs:union memberTypes="ManifestVerificationType FormatVerificationOutcomeType status:PolicyValueType
      risk:RiskAssessmentActionType"/>
</xs:simpleType>
```

This union allow one to use the referenced enumerations separately or as a combined set, as appropriate.

Dependencies

Harvard METS Toolkit

http://hul.harvard.edu/mets/

Harvard METS Java Toolkit, Version 1.3, was used for reading and creating METS files.

This software uses the Java "List" class for maintaining an ordered collection of elements. Care must be exercised when using the "add" method to append or insert new elements in the list, so as to put the elements in a correct sequence that will pass validation against the METS schema.

There is a possible bug in the Mets class's setSchema method. I could not get it to add the MODS namespace declaration to a Mets element that had been read in from a pre-existing METS file, but it seemed to work OK when creating a METS document from scratch.

There is also an incompatibility between the output of the METS toolkit and version 2 of the METS XLINK schema (dated Nov. 15, 2004). http://www.loc.gov/standards/mets/xlink.xsd This problem is apparently due to an error of omission in that version of METS.

Some METS output I produced with the toolkit does not validate in XMLSPY because the mdRef elements contain xlink:type="simple" attributes. XMLSPY complains: "Unexpected attribute 'xlink:type' in element 'mdRef'. Apparently xlink:type is not part of the xlink subset that is currently utilized by the METS schema.

Looking at the source code for MdRef.java, method write, I find the line:
```
      w.attribute ("xlink:type", "simple");
```
which tells me that this attribute is written by default with no option to omit it.

Stephen Abrams stephen_abrams@harvard.edu wrote in an email:

"Actually we've just run into the same problem (in a completely different context) here. I'm not sure why, but we've found that if we use the "official" XLink namespace URI <http://www.w3.org/1999/xlink>, rather than the URI defined in the METS schema <http://www.w3.org/TR/xlink> then everything validates properly.

I'm not sure why the METS version doesn't use the type attribute. It's pretty clear from the XLink specification http://www.w3.org/TR/2001/REC-xlink-20010627/#dt-must that "The value of the type attribute must be supplied" (Section 5.3). Since it is a requirement, that's why the toolkit automatically produces the attribute.

I've placed a revised version of the toolkit that uses the "1999" version of the URI on the project website."

JHOVE 1.0b2

http://hul.harvard.edu/jhove/

JSTOR/Harvard Object Validation Environment (JHOVE) was used for file identification, validation, and deep analysis.

Version 1.0b2 was used with the addition of a privately communicated HTML module. The currently available version of JHOVE (1.0b3) uses a greatly modified API interface. Modifications to Empirical Walker will be required before the new version of JHOVE can be used.

Some creativity was required for analyzing the "Properties" data from a RepInfo class instance. Inspired by examination of the hierarchical display used by JHOVE gui application, I used a javax.swing.tree.DefaultMutableTreeNode to hold the this data instance.   In retrospect I wish I had chosen to parse XML output from JHOVE instead.

Note that arrays of strings are used for the MIMETYPE and FORMAT variables of the various JHOVE module classes. It appears that the first member of each array is usually the value returned by default by the RepInfo getMimeType or getFormat methods, but there are exceptions. (search for "info.setFormat" and "info. SetMimeType" in the JHOVE code.)

Some minor points of concern:

The AiffModule of version 1.0b2 reports a MIME type value of "application/aiff" instead of the more commonly used "audio/x-aiff".   A work-around for this was incorporated into the Empirical Walker code. This behavior has been changed in version 1.0b3

The BytestreamModule reports a format name of "bytestream" (lower case), which is inconsistent with the use of upper case in other modules.

XML Beans

http://xmlbeans.apache.org/

XML Beans is an XML-Java data binding tool that was used to compile XSD schemas into JAVA jar library files that can be used to read or create XML files based on a schema.

Documentation for XmlBeans usage could be better. Note that there is extensive use of nested interfaces, which requires variable type declarations that can be quite lengthy. For example:

```
SdrDigiprovDocument.SdrDigiprov.Event.EventIdentifier.EventIdentifierValue.Enum eEventId;
```

Known Problems and Future Enhancements

Empirical Walker has been run on collections of 57,000 items, but required as much as 1 GB of memory to be allocated to the Java Virtual Machine (JVM). Larger collections can cause out of memory errors and abrupt application termination.

EW's memory consumption has been investigating using the JVM's -verbosegc option to generate a trace of memory consumption (by object creation) and recovery (by garbage collection). Our trace showed that we were getting an out of memory situation well before the allocated memory was fully consumed.

It appears possible that heap memory usage had gotten fragmented and that the garbage collection process was unable to compact the live objects in order to free up a large chunk of contiguous space. So when a new Java object was created that needed a fair amount of heap memory, the JVM was unable to locate a large enough chunk of contiguous memory and the OutOfMemoryError occurred.

Another possibility is that a memory leak of some sort may be taking place. Memory leaks occur when an unused object has references that are never freed. A memory profiler was used to monitor instantiation counts of the classes being used by a Java application. Running this program showed that an extremely large number of Strings, StringBuffers, and Char arrays are being created and that most of these objects are created by JHOVE.

Note that we are using JHOVE 1.0b2 with this version of EW. This testing should be repeated using the recently released JHOVE 1.0b3 (which has hopefully fixed these memory problems). Unfortunately, the program has been changed significantly and time available in February did not allow revision of EmpiricalWalker to use the new version.

Another major change to Empirical Walker would be to make it more modular in the sense that the output for one workflow phase would be saved out to disk and become input for the next phase.

Submission → Manifest → JHOVE output → DigiProv output

Our programmer realized somewhat too late in this project that the Empirical Walker should have been written as a set of separate modules instead of a single monolithic program. These modules would address the sequence of events that are involved with ingest, such as:

- Verification of transfer manifest

- JHOVE analysis and identification of file formats
- Assignment of format status (e.g. preferred, acceptable, minimal)
- Preservation risk analysis
- Transformations

Empirical Walker can produce METS output as well as output in other schema, but it cannot utilize those outputs for subsequent processing. The output from each of these processes should ideally act as the input for the next process. (We need a METS reader as well as a METS writer). This would hopefully reduce cpu and memory requirements for ingest of collections. It would also facilitate re-runs of later processes in cases were configuration parameters have evolved.

APPENDIX E

Stanford University AIHT Export Files

Richard Anderson
(rnanders@stanford.edu)

2004-12-29

----
aiht-content.tar.gz

Description:
The container file for AIHT collection

Size:
Compressed: 9272320 KB
Expanded: 12635471 KB

Created by:

```
ln -s  {rootdir}/archive/Files/websites/chnm/september11/REPOSITORY AIHT-CONTENT
gtar -czhf aiht-content.tar.gz  AIHT-CONTENT
```

(required 80 minutes for above command to complete)

Transfer:

```
Scp  aiht-content.tar.gz  stanford@belvedere.cs.odu.edu:Sites
```

209 minutes required for transfer (3.5 hours)

Notes:
This file contains the original AIHT files from George Mason University, as supplied to us by the Library of Congress. The only difference between this file and the original tar archive is that a different base directory has been used to generate the tar file. The directory …/archive/Files/websites/chnm/september11/REPOSITORY was symbolically linked to the name AIHT-CONTENT. The tar archive we created therefore contains files whose relative paths begin with AIHT-CONTENT (e.g. AIHT-CONTENT/CONTRIBUTORS/1199_photos/winmail.dat). The implications of this change have not been fully discussed by our group, but the motivation was to eliminate 57,450 repetitions of the "archive/Files/websites/chnm/september11" string in file path names.

----
aiht-dbases.tar.gz

Description:

The container file for the MS Access and MYSQL databases originally supplied with the AIHT collection

Size:

Compressed: 120024 KB
Expanded: 612611 KB

Created by:

```
ln -s {rootdir}/dbases AIHT-DBASES
gtar -czhf aiht-dbases.tar.gz  AIHT-DBASES
```

Transfer:

```
scp  aiht-dbases.tar.gz  stanford@belvedere.cs.odu.edu:Sites
```

3 minutes required for transfer

Notes:

This file contains the original AIHT database files from George Mason University, as supplied to us by the Library of Congress. The directory …/dbases has been symbolically linked to the name AIHT- DBASES. The tar archive we created therefore contains files whose relative paths begin with AIHT- DBASES (e.g. AIHT- DBASES /access/lc911digitalarchive.mdb).

----
aiht-mets.xml.gz

Description:

METS document describing the archive and database files

Schema:

http://www.loc.gov/standards/mets/mets.xsd
http://www.loc.gov/standards/mods/v3/mods-3-0.xsd

Size:

Compressed: 5664 KB
Expanded: 95240 KB

Created by:

Output from Empirical Walker program
(required 117 minutes to generate METS and METADATA files)

```
gzip -c aiht-mets.xml > aiht-mets.xml.gz
```

Transfer:

```
scp  aiht-mets.xml .gz stanford@belvedere.cs.odu.edu:Sites
```

<1 minute required for transfer

Notes:

The files and/or directories analyzed are assumed to be located in or below a single base directory. File references within the METS document are therefore expressed as paths that are relative to that base directory (rather than being expressed as absolute paths).

A decision was made to keep file-level metadata in a separate METADATA file hierarchy that mirrors the file hierarchy of the AIHT collection. This helps to keep down the size of the METS document and provides more granularity in cases where edits or replacement of the metadata for a single file was needed. This structure will also allow us to use the more efficient access mechanisms provided by the filesystem instead of the slower access methods of current generation XML tools.

ID Numbers:

ID numbers used to reference files or file level metadata begin with a prefix designating the element type (e.g. DIV_, FILE_, ADM_, & DMD_) and end with a segmented decimal number indicating the location of the file in the directory hierarchy. For example: ID= "DIV_4.2.29" indicates that we are referencing an object that is three levels deep in the hierarchy. The top level folder's DIV has ID="DIV_4", the second folder on the level below it has ID= "DIV_4.2", and the 29th file on the next level down has ID="DIV_4.2.29".

Files within METADATA file hierarchy have names such as raw_2.3.45, jhove_2.3.45, and digiprov_2.3.45, where the numbers correspond to the File IDs used inside the METS document.

<dmdSec ID="DMD_2" … >

The METS output document includes a single dmdSec using MODS elements to specify a collection directory hierarchy. This descriptive information provide some human readable, intellectual understanding of about what we consider to be the collection level structure of this whole archive. In this instance the descriptive metadata is very sparse, containing only directory names.

<dmdSec ID="DMD_2.01.001" … > to <dmdSec ID="DMD_2.13.70" … >

A sequence of dmdSec elements was generated for all files in the collection, with a mdRef element pointing to an external raw metadata file.

Example:

```
<dmdSec ID="DMD_2.01.002.2.01" CREATED="2004-12-27T07:10:12">
      <mdRef ID="RAW_2.01.002.2.01" LOCTYPE="URL" xlink:type="simple"
      xlink:href="./METADATA/AIHT-
      CONTENT/CONTRIBUTORS/1199_photos/wtc_web/raw_2.01.002.2.01.xml" MDTYPE="OTHER"
      OTHERMDTYPE="From relational database" LABEL="./AIHT-
      CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg"/>
 </dmdSec>
```

Note that a xlink:href attribute is used to reference the external raw metadata file and that the LABEL attribute is used to specify the relative path to the file within the collection

<amdSec ID="AMD_2.01.001"> to <amdSec ID="AMD_2.13.70">

A sequence of amdSec elements were be generated for all files in the collection, each one containing both a techMD element (for JHOVE output mdRef) and a digiprovMD element (for a preservation risk output mdRef).

Example:

```
<amdSec ID="AMD_2.01.002.2.01">
      <techMD ID="TECH_2.01.002.2.01" CREATED="2004-12-27T07:10:12">
            <mdRef ID="JHOVE_2.01.002.2.01" LOCTYPE="URL" xlink:type="simple"
            xlink:href="./METADATA/AIHT-
            CONTENT/CONTRIBUTORS/1199_photos/wtc_web/jhove_2.01.002.2.01.xml"
            MDTYPE="OTHER" OTHERMDTYPE="jhove" LABEL="./AIHT-
            CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg"/>
      </techMD>
      <digiprovMD ID="DIGIPROV_2.01.002.2.01" CREATED="2004-12-27T07:10:12">
            <mdRef LOCTYPE="URL" xlink:type="simple" xlink:href="./METADATA/AIHT-
            CONTENT/CONTRIBUTORS/1199_photos/wtc_web/digiprov_2.01.002.2.01.xml"
            MDTYPE="OTHER" OTHERMDTYPE="sdrDigiprov" LABEL="./AIHT-
            CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg"/>
      </digiprovMD>
</amdSec>
```

Note that xlink:href attributes are used to reference external metadata files and that the LABEL attribute is used to specify the relative path to the file within the collection.

Our original intention was to also include a mdWrap element within the digiprovMD element that would contain a subset of the information stored in the external digiprov file. This was not done however, dut to lack of implementation time.

<fileSec>

The fileSec element contains file sizes, mime types, and MD5 checksums for aiht-content.tar.gz, for aiht-dbases.tar.gz, and for each file contained therein. The data is presented in a flat layout and can be treated as the transfer manifest for the collection.

Example:

```
<file ID="FILE_2.01.002.2.01" MIMETYPE="image/jpeg" SEQ="3" SIZE="209046" CREATED="2004-
01-27T09:59:38" CHECKSUM="486089708ab81ecd3b2401f4fb37ed39" CHECKSUMTYPE="MD5">
      <FLocat LOCTYPE="URL" xlink:type="simple" xlink:href="./AIHT-
      CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg"/>
</file>
```

Note that a xlink:href attribute is used to reference the object file.
Note that due to a bug in EmpiricalWalker, the SEQ number sequence resets to 1 each time a new directory is encountered. This number should have been a count of files in the fileGrp element.

The top level fileGrp represents the base directory and contains these 2nd level fileGrp's:

```
<fileGrp ID="FG_1" … USE="AIHT Archive container file">
<fileGrp ID="FG_2" … USE="Files contained in the AIHT Archive">
<fileGrp ID="FG_3" … USE="AIHT database container file">
<fileGrp ID="FG_4" … USE="Files contained in the database container">
```

<structMap TYPE="PHYSICAL">

The structMap element contains a hierarchical representation of the container files and all files contained within them.

Example:

```
<div ID="DIV_2.01.002.2" ORDER="2" ORDERLABEL="./AIHT-
CONTENT/CONTRIBUTORS/1199_photos/wtc_web" LABEL="wtc_web" TYPE="folder containing 0
folders, 27 files, 5211521 bytes">
      <div ID="DIV_2.01.002.2.01" ORDER="1" ORDERLABEL="./AIHT-
      CONTENT/CONTRIBUTORS/1199_photos/wtc_web/WTC1.jpg" LABEL="WTC1.jpg"
      DMDID="DMD_2.01.002.2.01" ADMID="AMD_2.01.002.2.01" TYPE="file">
            <fptr FILEID="FILE_2.01.002.2.01"/>
</div>
```

Note that the ORDERLABEL attribute is used to specify the relative path of the directory or file. Note also that DMDID, ADMID, and FILEID idref attributes are used to reference dmdSec, amdSec, and file elements elsewhere in the METS document.

The top level div represents the base directory and contains these 2nd level div's

```
<div ID="DIV_1" ORDER="1" ORDERLABEL="./aiht-content.tar.gz" LABEL="aiht-content.tar.gz"
TYPE="file">
<div ID="DIV_2" ORDER="2" ORDERLABEL="./AIHT-CONTENT" LABEL="AIHT-CONTENT" DMDID="DMD_2"
TYPE="folder containing 2106 folders, 57450 files, 12375311120 bytes">
<div ID="DIV_3" ORDER="3" ORDERLABEL="./aiht-dbases.tar.gz" LABEL="aiht-dbases.tar.gz"
TYPE="file">
<div ID="DIV_4" ORDER="4" ORDERLABEL="./AIHT-DBASES" LABEL="AIHT-DBASES" TYPE="folder
containing 5 folders, 33 files, 626853252 bytes">
```

----

metadata.tar.gz

Description:

The container file for descriptive and administrative metadata files (referenced from within the METS document)

Size:

Compressed: 21304 KB

Expanded: 594090 KB

Created by:

Output from Empirical Walker program

gtar -czf metadata.tar.gz METADATA

Transfer:

scp metadata.tar.gz stanford@belvedere.cs.odu.edu:Sites

<1 minute required for transfer

Notes:

The metadata.tar.gz file, when expanded, will create a METADATA directory structure that mirrors the directory hierarchy of the AIHT-CONTENT directory and its children. All files in the METADATA directory are xml files. The filenames begin with the prefix raw, jhove, or digiprov followed by a unique numerical sequence that relates back to the ID numbers used in the master METS file.

Raw metadata:

Filenames: raw_n.nn.xml

Schema: rawMD.xsd

Purpose:

Provide access to a subset of the original database that contains the only descriptive metadata for the individual files within the collection.

Created by:

Using MS Access, created a query that pulled together all essential data from multiple tables into one flat view.

Using MS Sql Server, imported the data from the above query into a single database table.

Using Altova XMLSpy and MapForce, created an XML schema from the database table, and transferred the data into a single large XML file.

Using Java program and SAX library, split the single large file into multiple XML files.

EmpiricalWalker renamed files to standardized filename pattern.

JHOVE output:

Filenames: jhove_n.nn.xml

Schema: http://hul.harvard.edu/ois/xml/xsd/jhove/1.2/jhove.xsd

Purpose:

Output from JHOVE file analysis on the individual files

Created by:

Filename extension mapped to mime type (see extension-mimetype.txt)

Empirical Walker makes API calls to JHOVE in order to confirm or determine file format.

JHOVE parse output is written to file and also stored temporarily in memory.

DigiProv output:

Filenames: digiprov_n.nn.xml

Schema: sdrDigiprov.xsd

There is currently no comprehensive XML schema for recording preservation metadata. The work of the Preservation Metadata Workgroup (PREMIS) was therefore used to inform our design of a tentative XML schema to be used for documenting the results of our preservation risk assessment. Eventually we would expect to use the official PREMIS schema once it has been created.

Purpose:

Output from preservation analysis of the individual files

Events, agent, and object data written to xml file.
Events include format identification, preservation status assignment, and preservation risk assessment
Agent is the Empirical Walker program
Object information includes file characteristics, format identifiers, and format status

Created by:
File format determined by extension or JHOVE analysis
Default format scores, preservation quality, and policy status assigned
(sdrFormatStatusDefaults.xml is read for default values)
Preservation risk assessment is performed by comparing JHOVE output against a list of red flags
(preservationAssessmentFlags.xml is read. See also preservationAssessmentFlags.xsd)

See also:
SDRPresAssessDescrip.pdf and supporting documentation distributed to list by Hannah Frost

----
aiht-md5sum.md5

Description:
Output from running md5sum program against above files

Size: 208 bytes

Created by:
```
md5sum *.gz > aiht-md5sum.md5
```

Transfer:
```
scp aiht-md5sum.md5 stanford@belvedere.cs.odu.edu:Sites
```
<1 minute required for transfer

Notes:
If your system has the md5sum utility, then you can compare checksums in this file against the target files by placing all files in the same directory, then issuing the command:
```
Md5sum –c aiht-md5sum.md5
```

The belvedere host does not have the md5sum program, but does have a md5 command, which was used to generate md5 checksums for each *.gz file, which were compared against the contents of the aiht-md5sum.md5 file so as to verify that the upload was successful.
Our solaris system has a md5 command which is an alias for the openssl dgst program. But on our system this program chokes on large files.

XML Schema

See the config directory and the readme file contained therein.

Hardware Notes

Hardware = Sun Fire 4800 with four 750 MHz processors , 4 GB memory + 4 GB virtual memory

Software Notes

GNU Tar (gtar) version 1.13 was used to create container files. The 'z' option was used to filter the output through gzip, such that the files were simultaneously tar'ed and compressed. The 'h' option was used where appropriate to force the program to follow symbolic links.

GNU zip (gzip) version 1.2.4 was used for compression. This version work ok for compression, but there is a bug that can cause problems with decompression of files larger than 4 GB. A patch is available for version 1.2.4, or one can upgrade to 1.3.x to fix this bug. Or, one can use the following syntax: (see http://www.gzip.org/#faq10)

```
gunzip < file.gz > file
```

Empirical Walker is a program written in Java (using Borland Jbuilder-X Developer edition). This program recursively traverses the file tree, calculates checksums, determines file format type, analyzes the file contents, and performs preservation risk assessment. Primary output is to a METS xml file. Supplemental output is to individual metadata files, also in XML format. This program utilizes API mechanisms for invoking the functionality of Harvard METS Toolkit, JHOVE, and XmlBeans Java classes.

Harvard METS Java Toolkit, Version 1.3, was used for generation of the METS file.
http://hul.harvard.edu/mets/
Notes:

No major problems were encountered when using this software to create a very large METS file.

There is a possible bug in the Mets class's setSchema method. I could not get it to add the MODS namespace declaration to a Mets element that had been read in from a pre-existing METS file, but it seemed to work OK when creating a METS document from scratch.

This software uses the Java "List" class for maintaining an ordered collection of elements. Care must be exercised when using the "add" method to append or insert new elements in the list, so as to put the elements in a correct sequence that will pass validation against the METS schema.

JSTOR/Harvard Object Validation Environment (JHOVE) was used for file identification, validation, and deep analysis.
http://hul.harvard.edu/jhove/
Notes:

No major problems were encountered with the use of this software via an API interface.

We used JHOVE jar files supplied to us by Stephen Abrams, that include the new HTML module. Thus we were able to analyze a broader set of files that would have been possible with version 1.0b2

Some creativity was required for analyzing some of the data from a RepInfo class instance. We used a javax.swing.tree.DefaultMutableTreeNode to hold the "Properties" data from the RepInfo instance. This was inspired by examination of the JHOVE gui application, which uses this class for display of the information in a treeview. The alternatives were to parse the text or xml output, or define an alternative method of traversing the property hierarchy.

Note that arrays of strings are used for the MIMETYPE and FORMAT variables of the XxxModule classes. It appears that the first member of each array is usually the value returned by default by the RepInfo getMimeType or getFormat methods, but there are exceptions. (search for "info.setFormat" and "info. SetMimeType" in the JHOVE code.)

Some minor points of concern:

The AiffModule contains this statement:

```
    private static final String [] MIMETYPE = {"application/aiff", "audio/x-aiff"};
```

This has the affect that the RepInfo getMimeType method for created for files parsed with the AIFF module will return the value of "application/aiff" instead of the more commonly used "audio/x-aiff" (which we use in our default filename extension to mime type mapping) . This resulted in Format_Mismatch outcomes in our subsequent preservation risk assessment procedures. (Format_Mismatch = JHOVE found a different mime type than would be suggested by the extension.)

The BytestreamModule contains this statement:

```
private static final String [] FORMAT = {"bytestream"};
```

The use of lower case for the "bytestream" `FORMAT` string is inconsistent with the use of upper case in other modules. No big deal, however, and this was easily

XML Beans is an XML-Java data binding tool that was used to compile XSD schemas into JAVA jar library files that could be used to read or create XML files based on a schema.
http://xmlbeans.apache.org/
Notes:

Documentation for XmlBeans usage could be better. Note that there is extensive use of nested interfaces, which requires variable type declarations that can be quite lengthy. For example:

```
SdrDigiprovDocument.SdrDigiprov.Event.EventIdentifier.EventIdentifierValue.Enum eEventId;
```

Previous versions of EmpiricalWalker used the Castor Open Source data binding framework (http://www.castor.org/) and Altova XMLSpy's code generation function.

Altova XMLSpy and Mapforce was used for extraction of AIHT database data into XML format.
http://www.altova.com/

APPENDIX F

Report on Stanford's Ingest of the Johns Hopkins Export

Richard Anderson
Stanford Digital Repository
March 5, 2005

Introduction

Phase II of the Archive Ingest and Handling Test (AIHT) examines "the issues and protocol requirements for the passing of whole archives – with accompanying metadata – between institutions operating different preservation regimes." [AIHT Statement of Work]

The JHU Export

Stanford chose the export from Johns Hopkins University (JHU) for use in this phase of the project. The JHU export consists of a single TAR file with the following contents:

- "archive" directory containing 57,446 item data files, 57,446 item metadata files, and 2,106 group metadata files
- "code" directory containing java source code used for metadata generation, ingest, and export.
- "logs" directory containing logs created during metadata generation, ingest, and export
- lc911digitalarchive.sql = MySQL dump of the original archive metadata
- lc911toMETS.xls = Excel spreadsheet that maps original metadata to the JHU METS format
- description.txt = a fuller description of the export file's contents
- MANIFEST = checksums for all files in the export

The JHU export has a structure much different than Stanford's preconception of what a Submission Information Package (SIP) might ideally look like. Our ideal (of course) would be something that more closely resembles what <u>we</u> produced as an export product. In the JHU export, there is no single document that corresponds to our notion of a master metadata file. Instead the metadata is located in 59,552 separate METS documents (one per original data file or directory).

All the data files are stored in the archive directory, with names like "archive/item:12120.data.0", so the original directory structure can NOT be inferred from the path name of the data files alone.

The metadata file for each item (e.g. "archive/item:12120.xml") contains a field that records the path of the data file in the original GMU archive. Recreating the original GMU data hierarchy from this set of files would require a custom program that would read each of the 57,446 item metadata files. This assumes that the supplied metadata files are trustworthy. The metadata itself comes from the original GMU database, which was not totally in sync with the actual archive. Presumably JHU did a good job of matching the data files with the metadata (and fixing any discrepancies found).

In addition to the metadata files that are associated with each data file, JHU provides a "group" metadata file for each of the original directories. These files have names like archive/group:1001.xml". This information is also derived from the original GMU metadata database. Each of these files, contains the name of a directory, which would provide an alternative way to recreate a skeleton directory hierarchy.

We should note that all of the AIHT participants' archives (including our own) have unique structures that provide challenges for ingest by partner institutions. As a future follow-up project, we plan to pursue an examination of the metadata from all AIHT exports with an eye toward identifying best of breed practices that could help us all move closer to designing export package structures that facilitate cross-institutional transfers of collections. (Or that improve our internal AIP design).

The Philosophical Question

In January, Keith Johnson posed a philosophical question to the AIHT mailing list:

> The question is, simply, who is the "client" or "depositor" for the JHU version of the GMU archive? JHU, GMU, or both?
>
> In other words, what are we trying to preserve here? The original GMU archive as presented, or JHU's version of that archive?
>
> Originally I thought that the main point of transmitting an archive from one Repository to another was to get the original archive into a different preservation environment, thereby providing geographical redundancy, the robustness afforded by technological heterogeneity, and potentially an easier path into different delivery environments . Given this assumption, we logically should not worry too much about the form in which a Repository presents an archive to us (their particular METS/tar/etc . encoding, i.e., their DIP), but rather look through the DIP to the original form of the archive and try to ingest that--but benefit from the efficiency of a more regular encoding and any other value added by the other Repository (such as JHU's conversion of the descriptive metadata from its original database format into MODS elements.)
>
> But now my colleagues have pointed out that there could be real value to another repository in preserving their expression of an archive as-is . Presumably that would be a better backup for them than "our" version, which, if re-exported to them, wouldn't resemble what they gave us . So to bring up one of NDIIPP's burning questions, what does it mean to be a partner in digital preservation? If a Repository ingests and processes an archive, is their version then the ur-copy of that archive and any federated preservation efforts should place priority on preserving that Repository's encoding? Or is the ur-copy the original unprocessed version, and federated preservation efforts should prioritize preserving that?
> …

Clay Shirky responded:

> You are Dante . Tim DiLauro is Virgil . Marty over at GMU is Aristotle.
> Dante wanted access to the wisdom of the Greeks, but spoke only Latin, so Virgil became his guide, of necessity.
>
> That, I think, is your situation . Setting aside the oddity of your previous direct experience with the GMU archive, I think the answer is: You're *trying* to preserve the original, but the mechanism you

> have for doing so is the JHU archive; you can only do as good a job preserving an original not deposited with you as the depositor has done in preserving the original.
>
> (At the same time, of course, you're trying to preserve the JHU archive, as they may have created additional value in marking it up or, for accretive objects, in adding to it, in the way the people preserving the Temple of Dendur also worked to preserve the 19th c graffiti.)
>
> So for the receiver, I think the imperative is "See the original as clearly as you can, with the understanding that your Virgil is the donor." And for the sender, I think the imperative is "However you have marked it up, do whatever you can to help them see the original." (including, ideally, including a bit-identical copy of the archive as delivered.)
>
> This creates the risk that the size of the archive will grow as a multiple of the N times its been transferred -- when Stanford gives a copy of the archive to J.B.'s Preservation Hut down at the mall, JPH will have to preserve the GMU, JHU, and Stanford versions, and so on . (If you trust that previous versions will also remain available elsewhere, you can always treat this as a pass-by-reference problem, with pointers to intermediate versions not included in the current DIP.)
>
> This creates the meta version of the "preserve all versions, or preserve first, last, and edit decision list?" question, but, as with the intra-archive version of those choices, that question will probably require a context-specific answer .

So… to paraphrase Clay's response: we should be trying to preserve the original GMU archive to the extent that we can see it through the JHU archive. When doing so we should pretend that we have no previous knowledge of what the GMU archive looked like. We also should preserve any of the added value that JHU has contributed to the archive (such as the MODS descriptive metadata derived from the original database).

Stanford's Goals for Ingest

Keeping the above discussion in mind, Stanford explored the following optional strategies:

(1) Import the JHU export as is, treating the entire content as if it were data. That would allow us to be in a position to exactly recreate the JHU export at any future point in time. This option includes creation of a master METS document whose fileSec contains fixity information (checksums) that can be compared against the JHU transfer manifest.

(2) Treat the JHU METS metadata files in a manner similar to what we did with the XML files resulting from our own transformation of the GMU database metadata. We would run Empirical Walker against the JHU data files of the archive directory. The master METS document thus created would include amdSec and/or dmdSec elements with mdRef pointers to the JHU metadata (item.xml) files. We would retain the JHU file naming conventions. We would treat other files in the JHU export as data without accompanying metadata.

(3) Write a program that reads in each of the JHU item.xml METS documents. Use the original GMU file path information in these files to create a master METS structMap representing the original directory hierarchy and filenames. This structure was a pretty key aspect of the original GMU archive in terms of our understanding the grouping or collections inherent within it.

Time permitting, integrate the resulting structMap with other output from Empirical Walker.

(4) We also discussed, but eventually discarded, the option of attempting to physically re-create a copy of the original GMU archive file structure from the JHU export. We had also considered doing Empirical Walker analysis of this re-creation in order to compare it to the EW output from the original. This approach was criticized at a recent AIHT conference call as having the potential to introduce errors, but we still feel it is a potentially useful exercise for verifying the integrity of the archive.

Stanford has completed our test ingestion of the JHU Export, but we can only claim partial accomplishment of the goals we had set for ourselves.

- We have accomplished the minimal goals of option 1, producing a master METS file that also acts as a manifest for the collection.

- We did not have time to attempt option 2, electing instead to spend our time on the more interesting challenge afforded by option 3.

- We also completed the minimal goal of option 3, producing the desired structMap.

We learned a great deal along the way. We learned that our Empirical Walker design does not scale well when confronted with a collection having over twice as many files as the original GMU archive. We learned that our use of JHOVE and METS Toolkit need to be re-thought. We learned that there are data problems in the JHU export that were unanticipated.

The remainder of this report details the processes we followed and our findings.

Transfer Manifest

We successfully ran EmpiricalWalker in a minimal output mode in order to generate the fixity information needed to verify that we had indeed received all content files in an uncorrupted state.

We extracted the jhu-aiht-distrib.tar file into a base jhu-aiht-distrib directory. When this directory was traversed by EmpiricalWalker, it produced a METS file whose fileSec section that contains a list of the filenames and checksums.

A copy of this METS file has been uploaded to belevedere, and can be found at:
`https://belvedere.cs.odu.edu/~stanford/ExportsFromOthers/JhuMets.xml`

Since this format was not directly comparable with the JHU MANIFEST file, the following steps were performed to verify that the transfer did not introduce any errors:

- Wrote a SAX-based parser for the METS file that output a flat text file containing filename and checksum in tab-delimited form
- Wrote a simple java program that converted the JHU MANIFEST file into the same format.
- Did a global search and replace in the Stanford manifest file so that filenames had same prefix.

- Sorted both files into the same alphabetical sequence
- Used unix diff to confirm that there were no differences
- The above overhead could be minimized if a standard could be agreed to for the structure and file order of transfer manifest files.

Files Missing

We discovered that the archive directory of the JHU export contains 57,446 data files, 57,446 item xml files, and 2,106 group xml files. The original GMU archive contained 57,450 files. Perhaps JHU has already explained the missing files, but if not, we believe this discrepancy may possibly be connected to errors in the original GMU metadata as stored in database format. During preliminary analysis of the GMU archive, we discovered a number of filename mismatches between the names actually used and the text strings recorded in the database. Details provided upon request.

Bit-level Preservation

The Mets document generated by the above step in combination with the JHU export tar file would fill the need for bit-level preservation ingestion. We should be able to re-create and export the JHU archive as presented to us.

Preservation Analysis

We also attempted a more ambitious run of EmpiricalWalker in order to generate JHOVE output and preservation analysis information similar to what we had previously gathered on the original archive. In this case, the EmpiricalWalker was unable to run to completion due to out of memory errors. The larger memory consumption during analysis of the JHU export is due to the additional ~57,000 METS XML files in the archive which doubled the number of files to be analyzed.

When we first attempted this analysis on a production system, the process ultimately consumed all of the 1 GB of memory allocated to the Java Virtual Machine (JVM) and interfered with oracle database applications that were competing for the memory. For this reason we were unable to test larger memory allocations for the process on this host computer.

Memory limitations

To investigate the cause of the memory consumption in more depth, we transferred the JHU export to a Windows XP computer with a 3 GHz Pentium 4 processor, 1 GB of RAM and 4 GB of paging space. We experimented with various java memory allocation settings and performed analysis of object creation and garbage collection.

The maximum heap size is specified by using the -Xmx option to the JVM when running a Java application. As it turns out, the maximum heap size that the Windows implementation of JVM can utilize is 1500MB. This is due to the JVM's need for a contiguous allocation of heap memory together with Window's limitations on memory:

> See http://www.unixville.com/~moazam/2004/06/03.html
>
> "[Windows 32bit OS] processes can only use a max of 4GB memory address space. Windows further splits that into half by allocating 2GB to the kernel and 2GB to the application. The reason you can not hit the 2GB limit within the VM is because there is memory overhead that the VM and OS use for the process ..."

Garbage Collection

We also used the JVM's -verbosegc option to generate a trace of memory consumption (by object creation) and recovery (by garbage collection). Our trace showed that we were getting an out of memory situation well before the allocated memory was fully consumed.

```
....
[GC 893191K->851911K(1184496K), 0.0025800 secs]
[GC 893191K->851911K(1184496K), 0.0025648 secs]
[GC 860603K->851912K(1184496K), 0.0026493 secs]
[Full GC 851912K->591815K(1184496K), 58.5608954 secs]
[Full GC 591815K->591685K(1503232K), 3.1584876 secs]
java.lang.OutOfMemoryError
Exception in thread "main"
```

For any given line of the above trace, the value in parentheses is the total allocated heap size. The numbers to the left show the actual memory usage within the heap before and after a garbage collection cycle. Note that just before the crash, only 591685K (0.5G) out of an available 1503232K (1.5G) of heap was in use. The crash might be explained by assuming that the heap memory usage had gotten fragmented and that the garbage collection process was unable to compact the live objects in order to free up a large chunk of contiguous space. So when a new Java object was created that needed a fair amount of heap memory, the JVM was unable to locate a large enough chunk of contiguous memory and the OutOfMemoryError occurred.

Another possibility is that a memory leak of some sort may be taking place. Memory leaks occur when an unused object has references that are never freed.

Further research and consultation may provide clues as to whether an how this fragmentation can be remedied. For example, there is a lot of information to absorb in articles such as:

> http://www.petefreitag.com/articles/gctuning/
> http://www.skywayradio.com/tech/WAS51/Java_memory_tuning_tips.html

But such research is beyond the scope of this project.

Our working assumption is that we should instead devote our efforts to reducing the memory footprint occupied by the Empirical Walker

Memory Usage Analysis

In an attempt to figure out which java objects were consuming the most memory, We installed the Borland Optimizeit Profiler for Java. This program includes a memory profiler that allows one to monitor instantiation counts of the classes being used by a Java application. Running this program showed that an extremely large number of Strings, StringBuffers, and Char arrays are being created.

The profiler also allows one to determine which parts of the program are responsible for creating these objects. We found that as much as 50% or more of the Char arrays could be traced back to our use of JHOVE. We confirmed this by running EmpiricalWalker without invoking JHOVE and preservation analysis functionality. In this case the program completed successfully and the final garbage collection trace looked like this:

```
      [GC 351389K->328158K(376272K), 0.0325458 secs]
```

Note that we were using JHOVE 1.0b2 during this analysis. We need to repeat this testing using the recently released JHOVE 1.0b3. Unfortunately, the program has been changed significantly and time available in February did not allow revision of EmpiricalWalker to use the new version.

We will probably be forced to significantly rewrite EmpiricalWalker so that the JHOVE and preservation analysis steps are done in phases, with intermediate analysis results saved into disk files and consolidated later into the master METS document. This has been a valuable lesson for us to learn.

Filename character set

The files contained in the JHU export's archive directory all begin with "item:" or "group:". When transferring the export from Unix to PC we discovered that the colons in the filenames were incompatible with the Windows platform. We therefore had to rename all the files to substitute "-" for ":" in all filenames. This of course required creativity when utilizing the filenames in the metadata and the MANIFEST. We'd recommend that repositories consider using a more platform neutral naming convention in the future.

Reconstructing the original hierarchy

In addition to the above specified attempts at preservation analysis, we had planned to create a METS document that utilized the item xml files as metadata, rather than simply treating them as additional data files. One possibility we considered was to create a series of admSecs that contained mdRef pointers to the sequence of item xml files. Because of the above difficulties we have not have time to try that experiment.

We have instead pursued the more interesting experiment of attempting to utilize the 57,446 item xml documents to create a single METS document having a structMap whose div elements mirror the original directory and file hierarchy of the GMU archive. We have been successful in achieving this goal, albeit with a few bumps in the road.

We wrote a fairly simple Java program that sequentially reads each xml file, extracts the original path name, places the segments of that path into a hierarchy of TreeMap objects, then uses that TreeMap hierarchy to generate the structMap.

A copy of the structMap can be found at:
https://belvedere.cs.odu.edu/~stanford/ExportsFromOthers/JhuStructMap.xml

The bumps in the road were as follows

- We used the Harvard METS Toolkit to read each item xml file into a Mets object. We then traversed that object to find the <MODS:identifier displayLabel="Object Absolute Path" type="uri"> element that contained the original full path of the item. This element is located in an instance of the "Any" class. We discovered that the "Any" class does not have a method which can be used to retrieve and examine the attributes of the element. We therefore had to hard code a path to this element instead of testing the values of the attributes. In future we might instead use XPATH, a Sax processor, or a tool such as JHU's XMLUtil instead of the METS Toolkit for this purpose.
- The METS Toolkit was unable to read any of the 947 item xml files that contained a "&amp;" entity in the text. The error was:
  edu.harvard.hul.ois.mets.helper.MetsException: Invalid parser token: ENTITY_REF
  Further research is necessary to figure out why this exception is occurring. I believe that "&amp;" is a standard built-in entity that should not need declaration. For the purposes of this experiment, we simple ignored these errors.
- One of the item xml files (item-56066.xml) did not provide a full filename path. The path given was /websites/chnm/september11/REPOSITORY/IMAGES/
  This is probably a direct copy of an item record from the GMU database table (this bogus item record does exist in the database). We trapped and ignored this file.
- The StructMap class of the METS Toolkit has a "write" method that appears to generate no output. I had to put the structMap into a Mets object in order to use that class' write method.

If we had more time, we'd extend this experiment and proceed to generate an entire Mets document containing the dmdSec and amdSec elements that are stored in the item xml files. This would not be terribly difficult undertaking. The hard part was figuring out how to generate the structMap. And we believe we have demonstrated proof of concept. Hopefully our approach could be generalized and parameterized if a future need arises to use such a tool.

AIHT Collection Statistical Breakdown

	Hannah's	GMU	JHU				
	Estimate						
	fileCount						
	57450	57450	57446				
formats							
	formatCount			**formatName**	**mimeType**		
	162	162	162	AIFF	audio/x-aiff		
	19767	19879	30876	ASCII	text/plain; charset=US-ASCII		
	47	1076	3600	BYTESTREAM	application/octet-stream		
		1	0	BYTESTREAM	text/html		
	1302	1342	1336	GIF	image/gif		
	16680	17003	3650	HTML	text/html		
	12758	12764	12592	JPEG	image/jpeg		
	1664	1664	1659	PDF	application/pdf		
	1531	1537	1537	TIFF	image/tiff		
		0	11	UTF-8	text/plain; charset=UTF-8		
	2016	2016	2015	WAVE	audio/x-wav		
		0	1	XHTML	application/xml		
	1	5	7	XML	application/xml		
		1	0	XML	text/html		
formatVerificationOutcomes							
	outcomeCount			**verificationOutcome**			
		38235	0	Format_Verified			
		17164	0	Format_FromExtension			
		975	53846	Format_Mismatch			
		1076	3600	Format_Unknown			
formatScores							
	scoreCount			**scoreValue**			

	23504	23594	34601	0			
	29526	29773	16249	1			
	2966	3006	2995	2			
	0		0	3			
	499		0	4			
	955	1077	3601	5			
preservationQualities							
	qualityCount			**qualityValue**			
		53367	50850	StatusQuality_High			
		3006	2995	StatusQuality_Medium			
		1077	3601	StatusQuality_Low			
preservationPolicyOutcomes							
	policyCount			**policyValue**			
	24979	25102	36105	StatusPolicy_Preferred >			
	29715	29929	16404	StatusPolicy_Approved >			
	1302	1342	1336	StatusPolicy_Acceptable>			
	1454	0	0	StatusPolicy_Minimal>			
		1077	3601	StatusPolicy_Unknown >			
preservationRisk							
		fileCount		**flagID**	**recommendation action**	**newFormatName**	**newPolicyValue**
		11	11	AIFF_CompressionType	RiskAssessment_Warning		
		11	11	AIFF_CompressionName	RiskAssessment_Warning		
		16990	3637	HTML_ErrorMessage	RiskAssessment_Warning		
		3569	3637	HTML_NotValid	RiskAssessment_Warning		
		3569	3356	HTML_SyntaxError	RiskAssessment_TransformFormat	XHTML	

		13130	0	HTML_TokenMgrErro	RiskAssessment_TransformFormat	XHTML	
		13	13	HTML_IsValid	RiskAssessment_TransformFormat	XHTML	
		1998	160	JPEG_ErrorMessage	RiskAssessment_Warning		
		1838	0	JPEG_NotWellFormed	RiskAssessment_Warning		
		10926	12592	JPEG_ProgressiveEncoding	RiskAssessment_TransformFormat	JPEG2000/JPEG	
		0	5	JPEG_InfoMessage	RiskAssessment_Warning		
		1250	1249	PDF_EmbeddedImages	RiskAssessment_Warning		
		1663	1659	PDF_InfoMessage	RiskAssessment_Warning		
		12	7	PDF_ErrorMessage	RiskAssessment_Warning		
		5		PDF_NotWellFormed	RiskAssessment_Warning		
		26	26	PDF_Encryption	RiskAssessment_DowngradeStatusPolicy		StatusPolicy_Minimal
		1516	1516	TIFF_Not_DLF_Benchmark	RiskAssessment_Warning		
		1537	1537	TIFF_Compression	RiskAssessment_DowngradeStatusPolicy		StatusPolicy_Acceptable
		561	561	TIFF_MultipleImages	RiskAssessment_DowngradeStatusPolicy		StatusPolicy_Approved
		1486	1486	TIFF_InfoMessage	RiskAssessment_Warning		
		19	19	TIFF_ColorProfile	RiskAssessment_Warning		
		2	2	TIFF_ErrorMessage	RiskAssessment_Warning		
		0	11	UTF8_NotLatin	RiskAssessment_Warning		
		1	0	WAVE_ErrorMessage	RiskAssessment_Warning		
		1	0	WAVE_NotWellFormed	RiskAssessment_Warning		
		2013	2013	WAVE_FactChunk	RiskAssessment_DowngradeStatusPolicy		StatusPolicy_Acceptable
Transformations							
		16712	3369	Attempted			
		16591	3333	Success			
		121	36	Failure			

Report on Stanford Mass Transformations of AIHT HTML Docs to XHTML Format

Richard Anderson,
Stanford Digital Repository
April 1, 2005

Introduction

Phase III of the Archive Ingest and Handling Test (AIHT) provided "an opportunity for participants to examine and document a small test of file transformations that may be required when formats go out of scope." [AIHT Statement of Work]

Goals

Our goals for this phase were to develop designs and/or automated procedures for identifying files to be transformed, for performing transformations, for storage of the transformation outputs, and for recording the relationships between base versions and derivatives in metadata.

Identification of Files to be Transformed

The greatest part of our effort in Phase III was devoted to devising and implementing extensions of our preservation risk assessment procedures that would better support the automation of file transformations. We revised our PreservationAssessmentFlags.xsd schema and preservationAssessmentFlags.xml instance to better record the criteria for specifying transformations, and added data to the xml instance that is specific for files in HTML format.

For this trial we elected to perform transformations based on the analysis of JHOVE output. A HTML file was flagged for transformation if JHOVE identified it as being well formed and valid were . We also flagged files where a JHOVE error message was equal to any of the following values:

Unrecognized or missing DOCTYPE declaration; validation continuing as HTML 3.2
Close tag without matching open tag
Tag illegal in context
Parse error
Undefined attribute for element
Unknown tag
TokenMgrError

Empirical Walker was enhanced to be compatible with this new schema and was also modified to create a summary output file that contained a breakdown of the format identification, format status assignment, and preservation risk assessment outcomes for the entire collection. This report also lists any problem files found and any files flagged for transformation. The structure of this summary is specified in the sdrDigiprovSummary root element of SdrDigiprov.xsd.

Transformation Procedure

A separate Java program was written which read in the preservation assessment summary, and utilized the JTidy implementation (http://sourceforge.net/projects/jtidy) of HTML Tidy (http://tidy.sourceforge.net/) . The XHTML output and HTML Tidy log files were named using unique identifiers (e.g. _1.01.137.1.5.433.xhtml and _1.01.137.1.5.433.log. Output files were stored in a single directory separate from the original collection. A summary file listing the original filename, transformation filename, log filename, and transformation success or failure outcome was also produced by the transformation program.

Transformation Results

Of the 57450 files in the original archive, 17003 were identified by Empirical Walker to be in HTML format. Of those, 16712 were flagged for transformation to XHTML. 16591 successful transformations were performed with 121 failed attempts, due to unrecoverable errors in the original HTML. Time required for transformations: 11 minutes.

Here is a summary of relevant JHOVE output for the files that were selected for transformation:

Number of files	JHOVE results
13	Well Formed and Valid
3569	Syntax Error of some sort
13130	TokenMgrError

As mentioned, 121 files were unable to be transformed. JTidy output in this case includes the message: "This document has errors that must be fixed before using HTML Tidy to generate a tidied up version." The log files for these (and only these) transformations include error lines. There were a total of 236 error lines in all log files, with errors breaking down into the following error types:

Number of errors	Error type
148	Unrecognized or illegal element tag
70	Missing quotemark for attribute value
18	Erroneous use of '<' or missing '>' for end of tag

The transformation logs for all files include a total of 504,956 warning messages. (189677 from successful transforms and 315279 from failed transforms) Here is a summary of the types of warning messages found in all the logs:

Number of warnings	Warning type
291246	Non ASCII characters
106072	Unexpected quote mark
34913	Expected attribute is missing
14381	Missing close tag
10678	Missing a tag expected from context
8443	Empty element
7748	Unknown entity "&…"
6011	Discarding unexpected tag
5495	Repeated attribute name
5391	Entity missing closing semicolon
2870	Unknown attribute value
2769	Illegal data in container element
2341	Should use instead of </p>
2119	Closing tag in wrong position
1963	Misc illegal syntax
1562	Wrong closing tag for context
766	Unknown attribute
188	Proprietary attribute value

Filenames, Containers, Metadata

We unfortunately ran out of time before being able to decide upon a design for documentation of relationships between base files and derivative versions that resulted from transformations. We spent much time in discussions about file naming conventions, unique identifier options, container design, file locator services, and metadata for packages. Our design work in this area is still ongoing, and we would be happy to share our thoughts in this area with our partners at some future point. Perhaps the NDIIPP Technical Architecture Affinity Group would be a good forum for continuing this discussion.

Testing of Output Files

We have only done very minimal testing of the differences in browser renderability between base files and derivatives. No visible differences were observed in the few cases examined. Rigorous testing would require re-creating the original file structure with the derivative replacing the original base version. This would be necessary for relative href links to work.

An alternative mechanism for dealing with links was discussed at the AIHT meeting at Stanford. It was suggested by JHU that the internal structure of files be analyzed for links at the time of original ingest. The ingested file and its dependencies would then be re-written to use abstracted links that reference other objects in the repository using the unique IDs of those objects in place of file paths. When extracted from the repository, those abstract links could be re-constituted into relative filepath links. We did not attempt this procedure, but appreciate its merits.

Another relevant suggestion was that it would be much preferable for the submitting party to provide a URL to a web page instead of attempting to capture and transmit the web page and it's dependencies. This would help mitigate problems with filenames have the form of a URL instead of having a meaningful extension like ".htm". It would also enable the archiving facility to crawl the web page in a consistent manner and capture the dependent images or links in a reliable fashion.

Clay Shirkey suggested that an interesting test case would be to identify files that have a high degree of complexity involving layers of DIV elements. He postulated that a HTML file of this complexity may render successfully in a browser, but that the XHTML output from a transformation might fail to render correctly. In this case the transformation might actually degrade the preservation quality of the object. Unfortunately, in the time we had available, we did not discover any off-the-shelf tool that might identify such files in the AIHT collection.

Conclusions

Our programmer realized somewhat too late in this project that the Emperical Walker should have been written as a set of separate modules instead of a single monolithic program. These modules would address the sequence of events that are involved with ingest, such as:

- Verification of transfer manifest
- JHOVE analysis and identification of file formats
- Assignment of format status (e.g. preferred, acceptable, minimal)
- Preservation risk analysis
- Transformations

Empirical Walker can produce METS output as well as output in other schema, but it cannot utilize those outputs for subsequent processing. The output from each of these processes should ideally act as the input for the next process. (We need a METS reader as well as a METS writer). This would hopefully reduce cpu and memory requirements for ingest of collections. It would also facilitate re-runs of later processes in cases were configuration parameters have evolved.

APPENDIX H

Object Type:	ALL										
Document Type:	ALL										
Automated	**Subject**	**Question**		**sub-question**		**sub-question**				**Rationale**	**Comments**
	Acquisition	**Was the digital object derived from a physical (analog) source material, such as a book, manuscript, photographic print, or audio cassette?**								To determine if object is born-digital or product of digitization process.	
			YES	**Were SUL staff, facilities, or other resources (including funding) involved in the supporting the creation of the digital version of the original physical (analog) source?**	Yes					Level of insitutional involvement, commitment, and investment in creation of digital object may have bearing on level of service.	
					No	**Was the digital object purchased outright?**	Yes				
					Don't Know		No				
							Don't Know				
				Did grant-funding or other forms of donations support the creation of digital object from source material?	Yes					Grant-funding or gifts may have stipulations to consider when determining level of service.	
					No						
					Don't Know						
				Does SUL/AIR own the physical source object?	Yes						
					No					May be harder to re-create if necessary, therefore possibly weighs in favor of higher level of service.	
					Don't Know						
				Is the physical source object rare or unique (e.g., Special Collections material)?	Yes						
					No						
					Don't Know						
				Is the physical source object able to withstand the digitization process again, if necessary?	Yes						
					No						
					Don't Know						
			NO	**Is the digital object a derivative of another digital object?**	Yes	Please explain				To determine if the object in question is a copy derived from another digital object of potentially greater quality.	
					No	[Born digital master-copy]					
					Don't Know						
				Was the digital object purchased outright?	Yes					Does this have a bearing on level of service?	
					No	**Was the digital object a gift?**	Yes	**Are there any stipulations from the donor?**	Yes	May have a bearing on level of service.	
									No		
									Don't Know		
					Don't Know		No	**Was the digital object downloaded or copied?**	Yes	There may be legal complications.	
							Don't Know		No		
									Don't Know		

Automated	Subject	Question		sub-question		sub-question				Rationale	Comments
	Uniqueness/ Rarity	**Is the digital object considered rare or unique?**	Yes	Please explain.						Carefully define "unique". Could mean the equivalent of Special Collections material, such as born-digital manuscripts, or Web publications no longer online ("out-of-print").	
			No	**Is the digital object held by another research library(s)?**	Yes	**Is SUL's copy of the object different in any respect?**	Yes	Please explain / List possible reasons.		It is possible that SUL's version of a digital resource is of a higher quality, such as an increased order of mark up for enhanced discovery or utility, higher resolution, indexing, broader scope, complementary scope, unique functionality, etc.	
			Don't Know		No		No				
					Don't Know		Don't Know				
		Is the on-going maintenance of the object's authenticity important?	Yes	**What feature(s) of the object embodies its authenticity?**						Some objects, particularly born-digital "archival manuscripts" objects, have specific features or formatting which must be preserved in order to maintain the object's authenticity. Migration of such objects may prove challenging; documentation is critical.	
			No								
			Don't Know								
		Is the digital object replaceable, in whole or in part?	Yes	Please explain.							
			No								
			Don't Know								
X	Adherence to Quality Standards	**Does the quality of the digitizal object adhere to SUL/AIR's specifications for preservation-quality capture?**	Yes								Does this apply to reformatted objects only?
			No	Please explain.							
			Don't Know								
	Retention	**What is the object's retention period?**	Long-term (50+ yrs)								
			Middle-term (5-50 yrs)								
			Short-term (1-5 yrs)								
	Stability	**Is the digital object serial in nature or likely to be updated incrementally in the future?**	Yes	**Is the future issue or edition of the object likely to effectively replace/supplant the current object?**	Yes						
					No						
					Don't Know						
				Shall preservation services for the digital object in question change once future issues or editions of this object are ingested?	Yes						It is conceivable that preservation services to objects, especially complex ones, can change over time (a shift in level of service) if they are replaced by updated versions.
			No		No						
			Don't Know		Don't Know						
	Current Level of Use	**What is the object's current level of use?**	High								It is likely that the combination of current and future levels of use shall be considered in determining level of service.
			Medium								
			Low								
			Don't Know								
		How long is the level of use expected to continue?	[free text]								
	Future Level of Use	**What is the object's anticipated level of use in the future (such as in 10 years)?**	High								
			Medium								
			Low								
			Don't Know								

Automated	**Subject**	**Question**		**sub-question**		**sub-question**				**Rationale**	**Comments**
	User Population	**What is the object's current user population?**	SU								
			Broader research community								
			Unlimited								
		What is the object's anticipated future user population?	SU								
			Broader research community								
			Unlimited								
	Intellectual Property Rights and Other Legal Issues	**Is the digital object in the pubic domain (free of copyright restrictions)?**	Yes								
			No	**Does SUL retain the copyright to the digital resource?**	Yes						
			Don't Know		No	**Who is the copyright holder(s)?**					
					Don't Know						

Object Type:	ALL						
Document Type:	ALL						
					DO NOT		
Automated	**Question**	**sub-question**	**YES**	**NO**	**KNOW***	**Rationale**	**Comments**
X	**Is the object compressed?**		1	0		Compression may present problems in long-term care of digital materials. Proprietary algorithms may no longer be supported in the future. As a result, information may become either partially lost or totally inaccessible.	
		If yes, By what software program is it compressed?				To determine the nature of the algorithm employed.	
		If yes, is internal compression employed?	1	0		Potential inherent loss of quality	There are two kinds of compression: internal and external. An example of internal compression is LZW; Winzip, gzip, and StuffIt are examples of external compression algorithms.
		If yes, is both external and internal compression employed?	1	0			
	Is the object encrypted?		1	0		Security measures applied to data may inhibit long-term preservation.	Encryption can take the form of password-protection, public-key encryption, symmetric encryption, or watermarking. Each of these processes makes the object inherently more complex. PDF encryption is addressed in the PDF subsection of the questionnaire.
		If yes, How is it encrypted?	[free text]				
		If password-protected, is the password available?					
		Is the software used to encrypt the object known?	0	1			
		Is the software used to encrypt the object available?	0	1			
		Is it permissible to decrypt the object for ingestion/storage in the repository?	0	1			

Object Type:	ALL						
Document Type:	ALL						
					DO NOT		
Automated	**Question**	**sub-question**	**YES**	**NO**	**KNOW***	**Rationale**	**Comments**
	Are there other documents (in paper or digital form) associated with the object that are integral to the object's meaning and/or use?		1	0		Maintaining relationship between associated materials adds to the complexity.	Examples may include code books, finding aids, etc.
	Is another digital object (distinct from the subject of this questionnaire) in any way dependent on the object in question's current file name?		1	0		File names will change as objects are managed within the repository. Detailed logs will be maintained to document the changes. However, it is critical to know if other objects (within or outside the SDR) refer to the object in question as it is currently named.	
	Are there particular viewing requirements for the object?		1	0		Not necessarily a barrier to preservation, but potentially a complicating factor.	If yes, then higher level of metadata may be required. See questions pertaining to specific object types for related questions.
	Are there particular printing requirements for the object?		1	0		As above.	As above.
		MAX SCORE:	11				

Object Type:	**IMAGE**								
Document Type:	TIFF								
							DO NOT		
Automated	**Subject**	**Question**	**sub-question**	**sub-question**	**YES**	**NO**	**KNOW**	**Rationale**	**Comments**
X	TIFF Types	**Is the image multi-page?**			1	0		Some TIFF readers do not support multipaged TIFFs. They may display the first image but may not indicate that in fact there are additional images within the file.	
	Source Materials	**Is the TIFF derived from digital video (I.e, a video still)?**						May require some specialized metadata, if available.	
		Are the dimensions of the source material pictured in the digital image available?						If yes, then qualifies for higher level of metadata. Proper reading or understanding the digital surrogate may depend on additional information on the size of the original item; in such cases, metadata can be used to document these kinds of contextual data.	
		Is the physical condition of the original source material of research importance or otherwise of note with respect to the understanding of its digital surrogate?						If yes, then qualifies for higher level of metadata. Proper reading or understanding the digital surrogate may depend on additional information on the condition of the original item; in such cases, metadata can be used to document these conditions, as well as the disposition of original material.	
X	Imaging System	**Is information pertaining to the digital imaging system (scanner or camera maker, model, serial no., pixel size; scanning software version) known or available?**						If yes, qualifies for higher level of metadata.	
		If the performance of the imaging system has been evaluated, is the performance test data available?						If yes, qualifies for higher level of metadata encoding. It is possible to objectively measure the ability of an imaging system to reproduce color, line, etc. and the amount of "noise" in a system. Such testing is by no means mandatory; however, data from performance tests may be documented and preserved using metadata if desired.	
X	Color Info	**Is the image color?**							

Object Type:	**IMAGE**								
Document Type:	TIFF								
							DO NOT		
Automated	**Subject**	**Question**	**sub-question**	**sub-question**	**YES**	**NO**	**KNOW**	**Rationale**	**Comments**
			If yes, was color management employed during the creation of the image?						Color management is best practice.
X				**If yes, is the color profile embedded in the image?**	0	1		An external profile requires maintaining the relationship between object and profile throughout the object's lifecycle.	External profiles are preferred, despite the fact that they complicate data management.
			If yes, is the color information integral to the image's meaning and/or use, such that if the image were converted to greyscale, its usefulness as a resource would be lessened or nullified?					Some images may not need to be rendered in color to maintain their utility. Converting images to greyscale may save on storage room, and it is conceivable that such cost-saving measures, however drastic, may be necessary at some point in the lifetime of the repository.	Unscored: data management issue.
	Target Data	**Was a target used in the imaging process?**							Targets are included in digital objects to provide fixed reference points by which to measure objectively aspects of an imaging system. They may include test patterns, standardized color bars, scales/rulers, etc. Because the significant characteristics of a target are well-documented, the reproduction of a target by an imaging system indicates in a consistent way the idiosyncrasies of a specific imaging system (e.g. scanner, camera) which enables technicians or systems to apply appropriate corrections to the images.
			If yes, is it an external target?		1	0		An external target requires maintaining the relationship between object and target throughout the object's lifecycle.	Some targets may be included within the scanned image; for example, color bars are often imaged along with the subject (an internal target). Other targets, such as standard test patterns, may be imaged separately (per batch or project) and stored in a file separate from the object itself (an external target).

Object Type:	**IMAGE**								
Document Type:	TIFF								
							DO NOT		
Automated	**Subject**	**Question**	**sub-question**	**sub-question**	**YES**	**NO**	**KNOW**	**Rationale**	**Comments**
			If yes, is information about the target (such as maker, name, number, media) known or available?					If yes, then qualifies for higher level of metadata.	
				MAX SCORE:	3				

Object Type:	**PDF**		(This subset of questions will be developed further as PDF metadata is further assessed, and as a standard for the PDF-Archive format is established.)						
Document Type:	PDF								
							DO NOT		
Automated	**Subject**	**Question**	**sub-question**	**sub-question**	**YES**	**NO**	**KNOW**	**Rationale**	**Comments**
X	Versioning	**Is the software and version used to create the PDF object known?**			0	1		Many applications may be used to create PDFs, including several Adobe products in addition to Acrobat, as well as other third-party programs, such as the custom B-Scan application employed in the DL1 lab.	There is a distinction between the PDF creators (creating application) and PDF producers (distiller version). Typically auto-recorded in file metadata.
	Plug-ins	**Does the PDF object rely on plug-ins?**			1	0		Reliance on plug-ins may be a preservation challenge for some PDFs. According to Appendix H of PDF Reference v1.4 (3rd edition), "[An Acrobat viewer] will *read without errors any file that does not require a plug-in extension*, even if the file's version is older than the viewer's" (emphasis added). Because some plug-ins are required to enable features like annotations or hyperlinks and other interactive behaviors, it is conceivable that such functionality may become inoperable in the future if the appropriate plug-in is unavailable.H14	However, it is noteworthy that "a plug-in is never required in order to display the contents of a page."
X	Security and other Restrictions	**Is a password required to access or use the PDF object?**			1	0		Encryption of PDFs can be complex, and because PDF is proprietary and ubiquitous, PDF encryption may deserve special attention in this part of the questionnaire, in addition to the encryption questions for all digital objects.	
		Does the PDF object have a digital signature?			1	0			
X		**Are any functionality restrictions associated with the PDF object?**			1	0		Certain features and services can be restricted, such as printing or copying. Documentation of such restrictions is crucial to object management, particularly with respect to preserving the restrictions as necessary over time as well as eliminating such restrictions when and if it is appropriate (or possible) to enable migration.	

Object Type:	**PDF**		(This subset of questions will be developed further as PDF metadata is further assessed, and as a standard for the PDF-Archive format is established.)						
Document Type:	PDF								
							DO NOT		
Automated	**Subject**	**Question**	**sub-question**	**sub-question**	**YES**	**NO**	**KNOW**	**Rationale**	**Comments**
X			**If yes, is it necessary to maintain the restrictions as the PDF is used?**		1	0			
X	Interactivity and other Embedded Functionalities	**Does the PDF object have interactive features or other embedded functionality?**			1	0		Examples of such features include notes, active URLs, color highlighting, drawing, embedded sounds, as well as cause-and-effect "actions" and the ability to gather data using forms.	Related to plug-ins.
X			**If yes, is the interactivity integral to the PDF object's meaning and/or use?**		1	0		Such features may prove to be difficult to maintain over time.	
				MAX SCORE:	8				

Object Type:	**TEXT**							
Document Type:	Plain text / Word-processed File / Marked-Up Text / Dataset							
Automated	**Subject**	**Question**	**sub-question**	**YES**	**NO**	**DO NOT KNOW**	**Rationale**	**Comments**
X	Character Encoding	**Is the character encoding 7, 8, 16 or 32 bit?**		0	1			These are ASCII and "ASCII-like" char. encodings: ANSI; Unicode (ISO 10646); big-endian Unicode; UTF-8; UTF-16 (two-byte encoding of Unicode); ISO-8859-1 (Latin-1, ASCII plus characters for most Western European languages).
			Record character encoding.	[free text]				
X		**Are any special / encoded characters employed in the text object?**		1	0			Need to define further.
			If yes, define how the special characters are encoded.	[free text]				
		What languages are represented in the text?		[free text]				
X	Dependencies	**Is specific software required to render or use the text object?**		1	0			
			If yes, what is the software?	[free text]				
			If yes, is a specific version of the software required?	1	0			
			If yes, what is the version of the software?	[free text]				
			If yes, is the specified software version available?	0	1			
		Is specific hardware required to render or use the text object?		1	0		Hardware may include special I/O devices, display devices, etc.	What about processors? Specific processing speed may be a need for certain objects, such as animated artistic works. Justified as its own question?
			If yes, what is the hardware?	[free text]				
			If yes, is the hardware available?	0	1			
X		**Does the text object have embedded program code?**		1	0		Embedded macros, functions and the like usually rely on programs or scripts external to the document in order to function properly. As the object is migrated over time, maintaining this relationship in order to enable the original functionality becomes difficult or impossible in some cases.	Examples of embedded program code include a Microsoft Word macro, a Microsoft Excel function, or Javascript in a Web page.

Object Type:	TEXT							
Document Type:	Plain text / Word-processed File / Marked-Up Text / Dataset							
Automated	**Subject**	**Question**	**sub-question**	**YES**	**NO**	**DO NOT KNOW**	**Rationale**	**Comments**
X		**Does the text object reference external digital objects (external links)?**		1	0		Links to external objects, whether the external objects are controlled by SUL/AIR or not, are difficult to maintain.	
X		**Does the text object contain links which point to places within the object (internal links)?**		1	0		Internal links, such as Microsoft Word bookmarks or anchors in an HTML file, may be lost as the text object is migrated over time.	
		Is the text object marked up using a proprietary program such as MS Word or Wordstar?		1	0			
			If yes, What mark-up language and version is employed?	[free text]				As mark up languages (not to mention DTDs and schemas) evolve over time, support for features (tags, hierarchical structures, etc.) will inevitably change. For this reason, it is imperative that versions of mark-up languages, DTDs, and schemas are documented.
			If yes, How were the entity or character references encoded?	[free text]				XHTML (eg., &nbsp;); Hexadecimal Character Reference (eg., $#xA0;); Decimal Character Reference (eg., &#160;)
			If XML, is there reference to other XML specifications beyond the standard, such as xslt, xlink, or xquery?	1	0		Used like embedded program code to create certain behaviors. Requires attention and could be problematic over time.	
X		**Does the text object require a DTD, Stylesheet, or schema?**		1	0		Documents reliant on an external DTD (Document Type Definition), Stylesheet, or schema are inherently complex.	DTDs, stylesheets and schemas will be collected, ingested, and preserved in the SDR as objects worthy of preservation in their own right. The SDR shall maintain a listing of those DTDs and schemas it has ingested and thus supports. Depending on the nature of the DTD or schema, SDR clients may be encouraged or required to submit a copy or URI of the DTD/schema which supports the text object in question. While the availability of the DTD or schema contributes greatly to the preservability of an encoded text, it does not negate its complex, dependent nature.
			If yes, What is the required DTD or schema?	[free text]			Is the DTD a canonical DTD or is it a local variant that has been altered for local uses?	
			If yes, is the DTD or schema file been altered to make it work for a local resource?	[free text]			DTDs are often modified by users to make them work for local resources. Documenting that the DTD is non-standard is critical for preservation purposes. All DTDs both standard and non-standard should be submitted.	

Object Type:	**TEXT**							
Document Type:	Plain text / Word-processed File / Marked-Up Text / Dataset							
Automated	**Subject**	**Question**	**sub-question**	**YES**	**NO**	**DO NOT KNOW**	**Rationale**	**Comments**
			If a system file, Is the required DTD or schema available?	0	1			
	Formatting and Style	**Is the text object a dataset?**						To do: review NDAD metadata for datasets and evaluate for determining possible levels of service.
			If yes, how are the data delimited?					
			Does the dataset require special software?					Is special software required to manipulate the dataset?
			Does the dataset codebook being submitted with the dataset?					Codebooks are integral to the meaning of datasets. If at all possible the codebook needs to be preserved and linked with the submitted dataset.
X		**Is the way in which lines of text break within the text object integral to its meaning and/or use?**		1	0		Linebreaks may be represented in text objects in a number of ways. In marked-up text, linebreaks can be created using , <p>, or <lb>; depending on the version of the encoding language or how well formed the encoding is, the <p> and <lb> tags may be present with or without their respective closing tags. In free text, a linebreak may result from a line feed or from a carriage return. In any of these cases, as text objects are migrated over time, the way in which lines break in the document upon submission to the repository may or may not be supported in future rendering environments. For some types of text documents, such as poetry, the place in which a line breaks is intentional and thus crucial to its interpretation by human readers.	
X		**Is the horizontal white space within the text object integral to its meaning and/or use?**		1	0		For example, the use of tabs to arrange and format information for display must be assessed. It is common for such structuring to fall apart upon file migration.	
X		**Does the text object contain font treatment that is integral to the object's meaning and/or use?**		1	0			Font treatment includes any of the following: multiple fonts; styles such as italics, underlining, bold; colors; varying sizes.

Object Type:	**TEXT**							
Document Type:	Plain text / Word-processed File / Marked-Up Text / Dataset							
Automated	**Subject**	**Question**	**sub-question**	**YES**	**NO**	**DO NOT KNOW**	**Rationale**	**Comments**
X		**Does the text object contain symbols that are integral to the object's meaning and/or use?**		1	0			Symbols include bullet points, arrows, etc.
X		**Does the text object contain tables or other specially formatted or structured text that are integral to the object's meaning and/or use in their current form?**		1	0			This question does not refer to HTML tables.
			MAX SCORE:	19				