The International Journal of Digital Curation
Issue 2, Volume 3 | 2008

Migration Performance for Legacy Data Access

Kam Woods, Geoffrey Brown,
Department of Computer Science,
Indiana University

September 2008

Abstract

We present performance data relating to the use of migration in a system we are creating to provide web access to heterogeneous document collections in legacy formats. Our goal is to enable sustained access to collections such as these when faced with increasing obsolescence of the necessary supporting applications and operating systems. Our system allows searching and browsing of the original files within their original contexts utilizing binary images of the original media. The system uses static and dynamic file migration to enhance collection browsing, and emulation to support both the use of legacy programs to access data and long-term preservation of the migration software. While we provide an overview of the architectural issues in building such a system, the focus of this paper is an in-depth analysis of file migration using data gathered from testing our software on 1,885 CD-ROMs and DVDs. These media are among the thousands of collections of social and scientific data distributed by the United States Government Printing Office (GPO) on legacy media (CD-ROM, DVD, floppy disk) under the Federal Depository Library Program (FDLP) over the past 20 years.

The *International Journal of Digital Curation* is an international journal committed to scholarly excellence and dedicated to the advancement of digital curation across a wide range of sectors. ISSN: 1746-8256 The IJDC is published by UKOLN at the University of Bath and is a publication of the Digital Curation Centre.

SOME RIGHTS RESERVED

Introduction

The electronic publication of important scientific and social data and documents over the past 20 years has created a preservation crisis because the software necessary to access them is facing rapid obsolescence (Hedstrom, 2003; Rothenberg, 1999). Ensuring that our grandchildren will have continued access to these data and documents requires the creation of new software systems that enable their conversion (migration) to modern renditions where feasible, and the execution of obsolete software (emulation) where necessary (Department of Defense, 2007; National Archives and Records Administration [NARA], 2003). We describe our efforts to create a software system to enable web access to a large heterogeneous document collection. Our focus is on the issues we faced in supporting migration on request, which provides conversion to modern renditions in response to user request, and the corresponding performance data.

Our system combines two widely discussed preservation strategies – migration and emulation. We employ both static and dynamic migration as a primary strategy to enable browsing and searching the document collections[1] We use access to documents where migration is infeasible or leads to information loss. We are also exploring the use of virtualization as a strategy to maintain viable migration tools. Although we discuss our system architecture and implementation, the primary focus of this paper is migration. Our objective for migration is to enable browsing of heterogeneous document collections.

Our test workload consists of many of the thousands of document collections distributed by the United States Government Printing Office (GPO) through the Federal Depository Library Program (FDLP) over a 20 year period[2]. These collections, created by various government agencies, were originally distributed on approximately 5,000 CD-ROM and DVD disks[3]. They consist of millions of individual files including fundamental data on economics, environment, population, and life and physical sciences. Accessing many of these items requires installation of proprietary and increasingly obsolete software. For example, a sample of 1,885 discs contains 19,089 Lotus 1-2-3 files for which support has been discontinued in Office 2007[4]. The FDLP collections are an attractive research model for preservation because of the diversity of the formats used and the time span over which they were created. Furthermore, the number and size of the collections mandates the use of automated preservation techniques which, as we shall describe, stress the available tools.

The FDLP collections were originally organized as file systems with independent directory structures. A collection was used by mounting its file system (using removable media) on a suitably configured workstation (Federal Depository Library Program [FDLP], 2005). A user could then explore the file system, open documents and data with appropriate applications (often included on the media), and execute programs provided on the collection media. This file system structure is important in preservation because many of the files (e.g. HTML) depend upon the original path

[1] The strategy of migration on request uses original documents as the input for migration tools that may change over time (Mellor, Wheatley & Sergeant, 2002).
[2] About the FDLP http://www.access.gpo.gov/su_docs/fdlp/about.html
[3] Regional depository CD/DVD database http://www.uky.edu/Libraries/cdrom04inv.xls
[4] Deprecated features for Excel 2007 http://blogs.msdn.com/excel/archive/2006/08/24/718786.aspx

structure to be correctly rendered. Consequently, our current system is organized around a *virtual file system*. As described in *System Overview*, a user browsing a collection is presented with a view which may include both the original collection files as well as migrated renditions of these files and contemporary usage information and metadata. These dynamic views are generated from bit-faithful images of the original collection media. A significant feature of our system is the capability for both privileged and non-privileged users to contribute to the existing metadata and usage information presented by our system.

The remainder of this paper is organized as follows. We provide an overview of our system architecture and current implementation. We present data about the test collection including the set of file types and number of objects of each file type, and data about migration including associated space and time requirements and information about the failure modes of the migration tools we use under *Document Migration*. We conclude with a summary of results and a discussion of related work.

System Overview

Our system consists of two major logical components: a *virtual file system* which maintains all of the data and metadata for the various collections, and a *web application* which provides access to this file system. The virtual file system is organized at the collection level with subdirectories for media images, links to mounted images, static files such as usage notes and metadata, and dynamic files such as those produced by migration tools. The web application provides user control over the file system view, and provides privileged access to modify supplementary files such as metadata. The basis for preservation in our work is bit-accurate ISO (ISO-9660) images of CD-ROMs and DVDs – all renditions are derived dynamically from these images. Support for browsing ISO images is a standard part of Linux and BSD; we use auto-mounting to manage mounting images on demand. Our system also supports browsing within archive formats such as TAR and ZIP.

A session view is created by stacking the various file system directories to form a single union which merges files within matching directory paths with permissions derived from session information. These stacked directories include synthetic files that are generated on access including migrated renditions of legacy files. The idea of utilizing synthetic file systems to present dynamic data has a long history in operating systems, most notably in Plan 9 (Killian, 1984; Pike et al., 1995). Union file systems have been widely implemented and are used for applications such as “live CD” Linux distributions that support execution of an operating system without installation (Pendry & McKusick, 1995; Write & Zadok, 2004).

As an example of this file system model, consider Figure 1. The “root” directory of a collection contains four subdirectories: “media” containing raw ISO-9660 images, “files” containing mounted versions of each of the raw images, “static” containing catalog metadata (marc.xml), usage notes and additional files associated with the mounted images, and a “dynamic” directory with files derived from the original collection data[5].

[5] For performance reasons, it may be desirable to pre-compute and cache some dynamic files.

In the GPO collections there are many obsolete files and directories. Examples include applications such as Internet Explorer for which there are more modern versions available, or installation directions that are outdated with respect to more recent operating systems. In keeping with our principle of deriving views from original media images, we elide these directories through the use of *whiteout* files. These whiteout files may be created both statically by system administrators and dynamically based upon the current session. For example, it may be necessary to block access to certain files for copyright reasons; this requirement can be implemented for all users with a static whiteout or dynamically based upon current session attributes including preferences, location, and user identification.

Returning to our example, a session with no access restrictions and with migration enabled might be logically organized as illustrated in Figure 2. The actual presentation

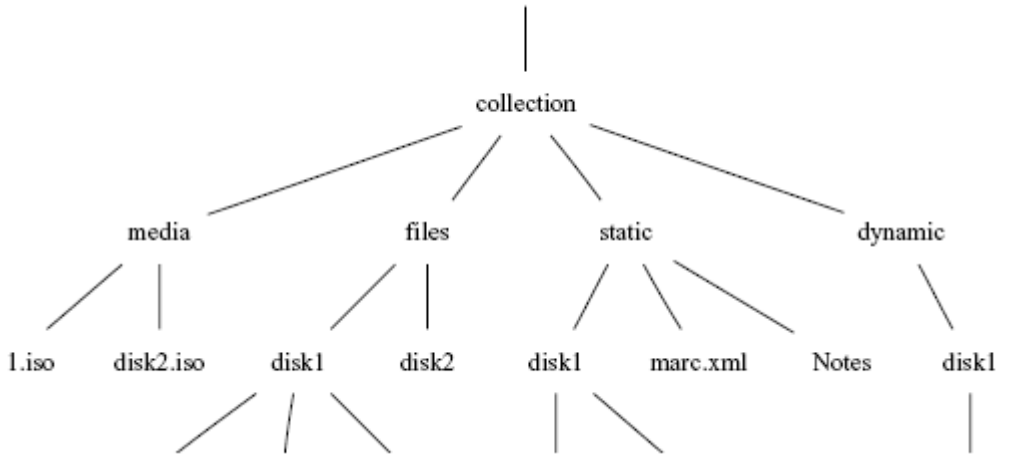

Figure 1. Example "File System"

of this view is provided by the web application which generates root pages for each collection from the cataloging information and contributed metadata.

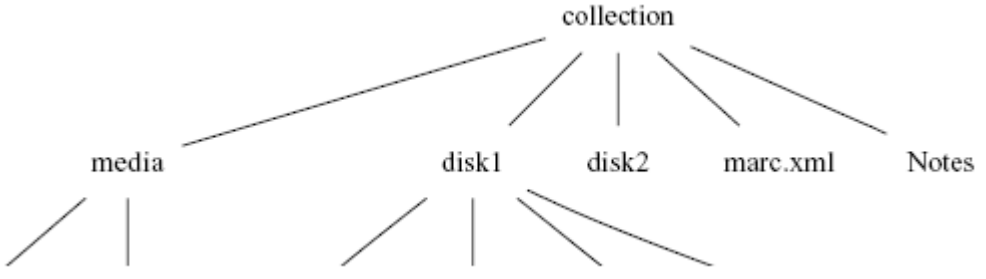

Figure 2. Example "File System View"

Our implementation is built using the open source Django framework written in Python[6]. Django naturally supports a file system organization; a Django application is organized around a set of "path rules" which are used to match against the path in a URL request. A selected rule is directly mapped to executed code. This code may access static files or generate dynamic content. Django provides an attractive model that maintains a strong separation of content and functional elements.

Our system provides a search interface that allows location of collections using

[6] Django: the web framework for perfectionists with deadlines http://www.djangoproject.com/

the CQL query syntax[7]. On completing a query, the user is presented with a view of the relevant items from the "virtual" file system constructed from the individual ISO images. Additionally, the user is presented with statistics on the search, including the number of items restricted, based on the user's privilege level and location. On selecting an item, the user is provided the opportunity (again permission-dependent) to update the archival metadata associated with that item.

Server-side modules and utilities for file migration operate both independently of the web-based interface as daemons in full-batch mode and triggered by user navigation. A model within the Django framework specifies those file types that are candidates for migration and whether the migration for a type has been precomputed through batch processing, should occur synchronously for each file requested from the server, or should be processed asynchronously. Asynchronous migration requests are triggered the first time a user navigates to the top-level directory for a particular ISO image. All files with types corresponding to asynchronous migration requests within that image are migrated in the background while the user continues to navigate the archive.

Many of the GPO collections include files which can only be accessed using proprietary programs included with the collection. We have implemented web-based virtualization which allows any of the media images to be mounted in a Vmware-based virtual machine[8]. Users can execute programs on the media image within the virtual machine. Under user control, the server allocates a preconfigured virtual machine, mounts the requested media image, and provides a browser applet for access. The current guest images consist of Windows XP with Microsoft Office 2003 and various other applications. File transfer support between the user and guest machine is provided securely within the web browser. Printing is supported through a PDF printer driver. Console access is provided using a VNC Java applet[9].

Document Migration

We describe our approach to migrating legacy files to modern formats and extensive experimental results analyzing the success rate of our tools as well as the space and time cost of migration. Our goal for migration is to enable users to explore the available collections in order to identify documents and data for further study. We convert formats that are not readily accessed within a web browser - including office documents, databases, and older multimedia – into browser-accessible formats including HTML, PDF, and Flash Video. We also describe our approach to an important technical problem relating to document preservation – the reliable identification of file formats.

Our migration model is *migration on request*, which translates from the original file format to modern format in response to user requests. Migration performance can be enhanced by "caching" pre-converted files; however, our conversion path is always from the original file to modern rendition. The advantages of this approach are documented in existing literature, and associated risks are mitigated here through the exclusive use of open source tools which can themselves be readily preserved (Mellor et. al., 2002).

[7] Cql: Contextual query language http://www.loc.gov/standards/sru/specs/cql.html
[8] Vmware home page http://www.vmware.com/
[9] About RealVNC http://www.realvnc.com/

The migration tools we analyze are all open source. For example, we use OpenOffice to convert Microsoft Word, PowerPoint, and Corel WordPerfect files to PDF[10]. Similarly, we use open source libraries to generate HTML renditions of DBase and Microsoft Access databases. An alternative technology which we intend to explore in subsequent work is the use of Microsoft Office on a virtual machine as a translation server; there are well known risks to this approach[11].

The remainder of this section is organized as follows. We begin with a discussion of file format identification. We then describe our migration tools and their average time cost and success rate. We conclude with a discussion of migration and a deeper analysis of performance issues including the space requirements for caching precomputed migrated renditions.

Format Identification

Reliable automated identification of file formats is a key technical issue with document preservation. There are projects focused upon cataloging and identifying file formats, but there are no solutions which are immediately applicable for systems such as ours[12] [13] [14]. Initially we performed file format identification using the UNIX file utility and suffix matching[15]. We subsequently modified the approach by substituting `libsharedmime`, an implementation of the open source Freedesktop Shared MIME-Info Database that provides for both identification and basic validation of a vast range of formats.[16] (A similar approach is used in the DROID system (Brown, 2006).) `Libsharedmime` has significant advantages over DROID and JHOVE in terms of performance, breadth of format support, integration into the UNIX environment, ease of database customization, and cost. Efforts to provide extensive format characterization in preservation-specific software tools are unquestionably important. High-performance automated identification and validation of large, heterogeneous collections of files (and associated formats) are effectively addressed by open source tools used to support major desktop systems such as Gnome and KDE.

`Libsharedmime` in particular provides relatively sophisticated measures for examining - in order of preference - an explicitly provided MIME type, high-priority "magic data" rules (for explicit binary sequence or text-based identifiers within the file itself), "glob" rules for matching based on file name, and finally low-priority magic rules. Along with a database customized to a particular collection, these methods result in a very low false-positive rate for file identification. The major file types identified using these techniques and their frequencies are shown in Figure 3.

[10] Openoffice: Free office suite http://www.openoffice.org/
[11] Considerations for server-side automation of office http://support.microsoft.com/kb/257757
[12] The technical registry Pronom http://www.nationalarchives.gov.uk/pronom/
[13] Global digital format registry http://hul.harvard.edu/gdfr/
[14] Fred: A format registry demonstration http://tom.library.upenn.edu/fred/
[15] Fine Free File Command http://www.darwinsys.com/file/
[16] Shared MIME-Info Database http://www.freedesktop.org/wiki/Specifications/shared-mime-info-spec/

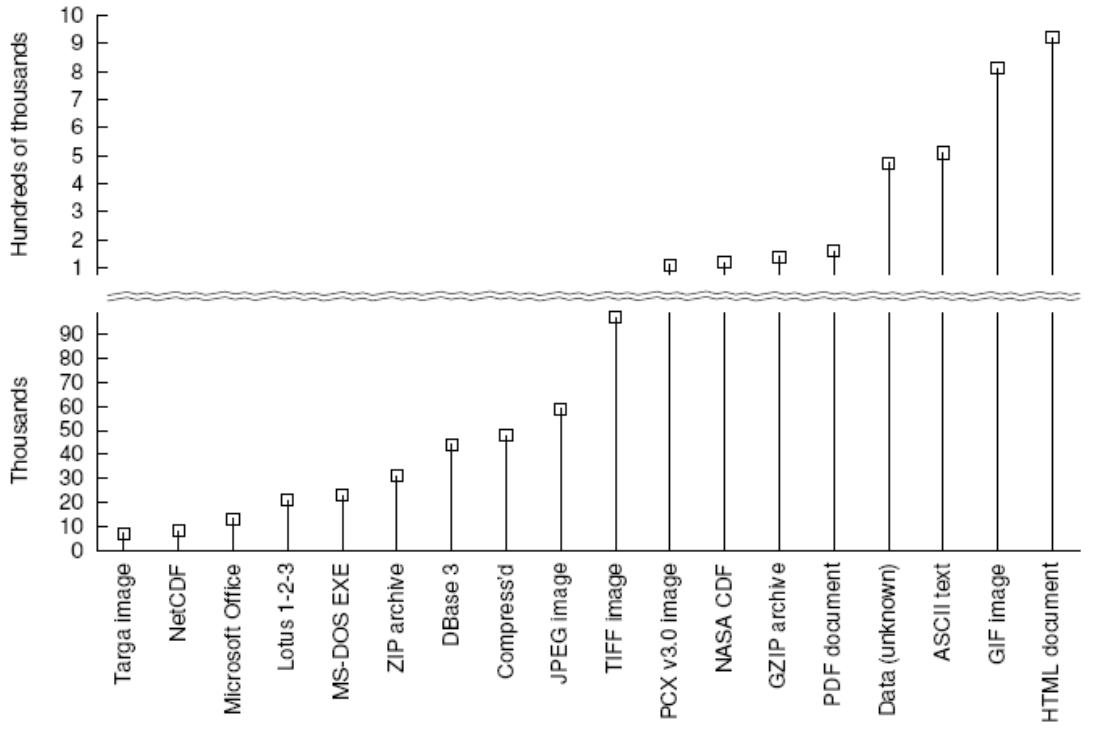

Figure 3. Top 18 file distributions by type.

It is important to note that there is no reliable, fully automated way to identify file formats. For example, dBase index files cannot be identified by content, though their suffix (`.idx`) in the context of related database files is a strong indicator. We found it necessary to "tune" the database used by `file` in order to eliminate obvious false matches and to develop suffix matching in the context of the collections we are studying. While tuning our identification heuristics was a necessary step, it was not particularly time-consuming. Ultimately this hybrid format identification module proved sufficient for the current set of experiments, in spite of known limitations. Using the data extracted from file identification, we selected the migration paths in Table 1 for further study.

	Suffix	Number in Archive	Destination
Image formats	.gif .bmp .pcx .tif .png	697624	JPG
NASA CDF	.cdf	122079	ASCII text, XML
DBase III, IV	.dbf	35070	XML / HTML
Lotus 1-2-3	.123 .wk1 .wks	19089	XML / HTML
MS Excel	.xls	12279	XML / HTML
MS Access	.mdb	2805	XML / HTML
MS Word 6, 95, 97	.doc .wri	1804	Adobe PDF
Video formats	.avi .swf .mov .mpg .vob	1224	Flash Video (.flv)
Corel WordPerfect	.wp4(5,6) .wp	1104	Adobe PDF
MS PowerPoint	.ppt	380	Adobe PDF

Table 1. Conversion candidates.

Those file formats selected for migration comprise nearly 25% of the 3.9 million files within the complete archive. Of the remaining documents, approximately 33% (915,138 HTML and 448,716 additional ASCII) required no migration. 18,461 files could not be read due to permissions errors or corrupt images.

653,287 files in the archive were identified as binary data without a further application reference. We developed a simple automated test to determine whether an individual file's contents consisted of greater than 70% ASCII text, and found 57,454 of these files met this criterion. Of the remaining 596,265 "true" binary files, 122,079 were NASA CDF files already accounted for in the migration scheme. 88,117 of the remaining files were easily identified by common extension and automated scripted tests as geospatial (GIS) data. An additional 86,671 GIS-derived binary files with no extension were located in related directories. Approximately 30 other well identified binary formats were observed – including various MS-DOS, Windows and UNIX executables, font files, and database support documents – each comprised of between 1,000 and 10,000 files. Several hundred older or more esoteric formats with fewer than 100 files were also identified.

Migration Tools

All migration tasks are handled with readily available open-source tools. A Python wrapper implementing a standard UNIX-style daemonization routine along with a high-precision timing library allows batch-mode processing of all relevant file types. Conversion of Microsoft Word and PowerPoint documents is performed via an instance of OpenOffice 2.2 running in "headless" mode, bridged through Py-UNO to a Python script which handles associated path construction and cleanup tasks. All image conversions are generated by the Python Imaging Library. Microsoft Access and DBase III/III+/IV files are converted with Python modules obtained from SourceForge that extract relevant file data directly from the known binary format. Spreadsheet data from Lotus 1-2-3 are extracted to a structured XML file via the Gnumeric ssconvert utility. Spreadsheet data from Microsoft Excel (through Office 2003) are extracted to a structured XML file via the xlrd Python module. Media objects, including MPEG-encoded video clips and .VOB files extracted from DVDs, are re-encoded in a web-suitable resolution (320x240) Flash Video using the ffmpeg utility.

We considered three basic profiles for translation of items within the archive:

- Batch conversion of all applicable file types within the entire archive, performed at appropriate intervals and kept in store. For very large archives, storage cost associated with this may be undesirable or prohibitive, although the continued rapid decrease in cost of storage mitigates this.
- Batch conversion of archival items based on access profiles. Requests for certain areas of the archive can trigger these conversions. Advantages include storage cost, suitable for rarely accessed archives.
- Conversion of data items just-in-time, on request. Good candidates for this approach include image files which may be rapidly translated and smaller non-binary or open binary formats. This method may also be used to partially convert substantially larger items, providing preview snapshots.

More complex translation paths are possible. Those documents in proprietary binary formats for which no freely available tools exist may still be converted efficiently without requiring the use of a fully interactive emulated environment (for example, scripting of DOS-based tools for file processing as batch jobs under Wine). As our system focuses on on-demand migration for document browsing, a 'simple' strategy suffices, where objects are retrieved directly and migrated via a direct mapping to the appropriate tool. In systems where migration is performed as a long-

term accessibility strategy, migrations may be triggered by an automated request from a migration registry, and applied to objects for which the contained files meet specific criteria. A basic outline of such a service is given in Figure 4. The complex migration path described is an approximation of prototype functionality described in the koLibRI documentation[17].

At the heart of this problem lies the question of whether dynamic (e.g. on-request) migration is an appropriate strategy. As noted above, storage costs continue to drop rapidly, while the processing time required to translate even moderately sized documents can be considerable. For many archives, the cost of added storage may be negligible compared to other costs associated with archive maintenance. Performing batch migrations to allow instantaneous access to the full document collection can therefore have significant advantages.

We discuss the techniques used to translate each of the identified file types and evaluate two distinct criteria for those types: the percentage of documents that can be successfully translated, and the average translation time required. A full evaluation of successful migration would necessarily involve multiple independent assessments of the migrated document contents as compared to our original "gold standard" archival objects; for this preliminary work, we consider a migration "successful" if a non-empty migrated object is produced without an error generated by the translation tool. This criterion for success is obviously not suitable for storage of migrated objects in an archival context. Given that the intended usage profile here is dynamic (on-request) generation during browsing of the archive via a website, it suffices for these preliminary trials. Our results are summarized in Table 2 below.

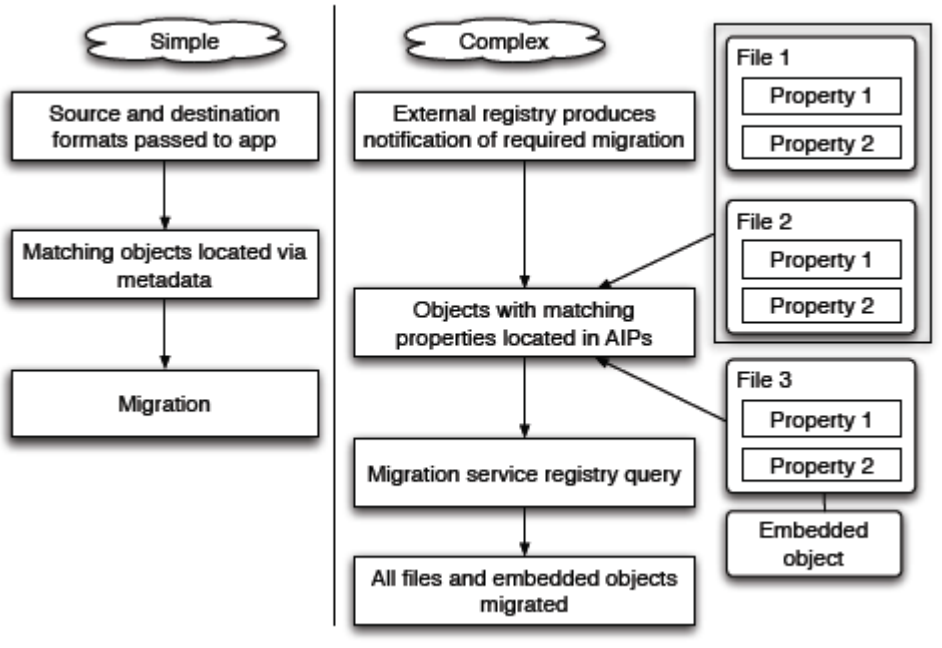

Figure 4. Migration strategies.

[17] kopal library for retrieval and ingest (koLibRI)
http://kopal.langzeitarchivierung.de/index_koLibRI.php.en

These statistics are included to illustrate the practical issues that arose in this migration strategy rather than provide a complete view of corrupt, damaged, or otherwise inaccessible items within the archive. As discussed in the previous section, some content from damaged images - notably areas where directory structures could be read but from which no files could be extracted – was discarded prior to any conversion attempt. These numbers therefore provide an approximation of real-world operation on these tools when using undamaged ISO-9660 images. Validation of both media and extracted ISOs can be difficult. The standard UNIX *isovfy* tool, part of the cdtools package, generates basic statistics on image integrity. These include the information on the tool used to produce the ISO, extension (Rock Ridge, Joliet) characteristics, and errors such as zero-length files for which extents have been written. In this set of images, 7 were identified with one or more null extents, while over 200 exhibited errors related to the use of Rock Ridge extensions. In many of these cases, the errors resulted from versioning issues with the extensions and had no apparent effect on file access. Future work will include additional analysis of the original media, along with testing of alternate ISO extraction tools.

	Successful conversions	Failures	Time/File
Image formats	692118	2908	< 1s
NASA CDF	122076	3	3s
DBase III, IV	34352	718	< 1s
Lotus 1-2-3	19086	3	< 1s
MS Excel	12268	11	< 1s
MS Access	2084	1	< 1s
MS Word 6, 95, 97	1267	99	7s
Video formats	1176	48	6s
Corel WordPerfect	1099	5	6s
MS PowerPoint	374	6	5s

Table 2. Conversion Success.

Microsoft Word, PowerPoint, and Legacy Word Processing Formats.

Migration of Microsoft Office documents in an automated environment presents a series of challenges. Microsoft specifically recommends against the use of server-side automation with the Office suite of applications, primarily for stability and security concerns. Additionally, alterations to the proprietary Microsoft Compound Document File Format, along with changes in the Windows operating system, have led to modern versions of the Office suite which will no longer reliably read all documents created with previous Office products. While Microsoft provides additional tools for the migration of such documents (at the time of writing, a conversion utility for migration of all documents from Office 97 through Office 2007 is available), these are similarly subject to server-side scripting concerns.

We explored the use of OpenOffice in the interpretation and migration of existing Microsoft Word and legacy WordPerfect documents. OpenOffice is capable of reading the proprietary Microsoft format, can be run in a “headless” environment suitable to execution in daemonized mode on a server, and supports a number of language bindings through the Universal Network Objects package that make scripting of conversion processes possible. While OpenOffice continues to suffer from a number of stability problems in the context of server-side automation, we were able to mitigate their effect through the use of a watchdog timer tuned to terminate processes after a threshold period. Our archive contains 494 files identified as productions of early (MS-

DOS) versions of Microsoft Word; they cannot be handled directly by OpenOffice, and no conversion attempt was made on them. Of the remaining 1,816 Word documents, approximately 450 were discarded prior to conversion due to errors in disk permissions, lack of data content, or other I/O issues related to access to the original file. Of those 99 conversion failures observed, the majority were on attempts to read and convert ASCII files containing some small percentage of non-ASCII data, and identified by the UNIX `file` command as *data*. Statistics for the various office formats are provided in Table 2.

Spreadsheet and Database Formats.

Our implementation uses ssconvert (part of the Gnumeric spreadsheet package) to interpret and migrate Lotus 1-2-3 documents, along with a variety of additional open-source libraries to handle Excel, Access, and DBase documents[18]. Data extraction from multiple worksheets or record format descriptions into individual HTML tables (or appropriately defined XML documents) is relatively straightforward. However, conversion may involve some loss of formula data or OLE inclusions.

For the Access and DBase formats, each migrated document represents a preview of the table structure, rather than a structured dump of all data within the original file, yielding significantly smaller sizes in the migrated collection. The purpose of this truncated migration is to enable web browsing without excessive download and rendering times.

The relatively large failure percentage on .dbf files is due primarily to a selection of files with correct header information but which were otherwise empty or contained malformed tables. These files were predominantly clustered on a small (less than half a dozen) number of media images which may be corrupt.

Video and Image Formats.

Candidate multimedia formats for conversion were selected based on frequency of occurrence. The era from which this collection dates precludes the appearance of a number of currently popular formats, including Windows Media video and MP3 audio files. FFmpeg with an appropriate version of the libavcodec was used to transcode all video sources into a Flash video format suitable for embedding[19]. All videos were re-encoded at a fixed bitrate to a resolution of 320x240, typically yielding a factor of 10 compression. All image data files were migrated to the JPEG format using the Python Imaging Library. Of the failures, a small number (73) were PCX files with non-v3.0 encoding. 2,835 TIFF images with non-standard encoding or otherwise damaged data structures failed conversion. Finally, 5,506 GIF images with correct heading information but containing no image data were discarded prior to any attempt at conversion.

Common Data Format.

The FDLP collection contains a large number of documents from NASA encoded in the Common Data Format (CDF), a self-describing binary format used to store scalar and multidimensional data[20]. In addition to a web client for format translation, NASA provides source code for a number of tools capable of interpreting this format.

[18] GNOME Office/Gnumeric http://www.gnome.org/projects/gnumeric/
[19] Ffmpeg http://ffmpeg.mplayerhq.hu/
[20] The common data format (CDF) http://cdf.gsfc.nasa.gov/

In our system, a text "skeleton" is created for each CDF document in the archive, consisting of the format-internal metadata along with a formatted list of variable identifiers and document data. The *SkeletonTable* script in the CDF utilities distribution creates ASCII representations which may contain data values for only a subset of the document-internal variables, resulting in significantly smaller migrated files suitable for preview via the related web application.

Performance Issues

While the migration times for most files as indicated in Table 2 are relatively short, they may be too long in certain cases (e.g. Microsoft Word documents) to provide an adequate browsing experience. As we discuss in this section, these averages can be misleading due to startup costs associated with translating a single file that are obscured by this metric. In the following we provide further data about the *latencies* associated with migrating files on request. Table 3 illustrates the space required to store precomputed migrated renditions for the various formats in our test collections. The entire archive occupies roughly 1.1TB on disk.

OpenOffice, which we use to convert various Microsoft Word, Powerpoint, and Corel WordPerfect formats, has significant performance overheads when used to convert single files. For example, in our trials Microsoft Word documents were converted in batches of 10, with a new child process running an instance of OpenOffice used for each batch. This operational profile was selected to overcome runtime and memory usage issues associated with OpenOffice 2.2. On our reference hardware running Red Hat Enterprise Linux, the startup time associated with each instance of OpenOffice was approximately 3 seconds. The storage requirements for preconverted versions of all documents migrated with OpenOffice are small, totaling just over half a gigabyte.

In contrast, the various image formats have high storage requirements (both in original and migrated rendition) and low latency conversion cost. For these formats, synchronous migration on request appears to be both viable and desirable.

In handling formats such as NASA CDF, DBase, Lotus, and Excel, our migrated rendition contains only a subset of the original data (recall our objective to enable browsing) and hence the storage costs for migrated renditions are small. Caching may be reasonable even with modest migration latency under such circumstances.

Finally, video formats are excellent candidates for caching, as we require only low-resolution renditions to support browsing and the quality of these renditions can be improved with more computationally intensive codecs. As with all files in this archive, the user may elect to retrieve the original if needed.

The system we have developed provides flexible migration profiles to handle formats with migration paths that vary significantly in terms of latency and throughput. The advantages to this approach include the ability to dynamically provide multiple migration paths for individual files and to maintain real-time access irrespective of file size.

Discussion

We have presented data summarizing experiments with a system providing *migration on request* for a set of 1,885 CD-ROMs and DVDs containing heterogeneous document and data collections originally distributed by the United States Government Printing Office.

	Number Converted	Original Size Total	Original Size Mean	Converted Size Total	Converted Size Mean
Image formats	692118	50676MB	75KB	11807MB	175KB
NASA CDF	122076	148543MB	1247KB	9121MB	77KB
DBase III, IV	34352	65410MB	1991KB	1360MB	41KB
Lotus 1-2-3	19086	617MB	33KB	2142MB	115KB
MS Excel	12268	1038MB	84KB	3698MB	308KB
MS Access	2084	12241MB	6018KB	331KB	< 1KB
MS Word 6, 95, 97	1267	405MB	330KB	196MB	160KB
Video formats	1176	7856MB	18KB	978MB	2308KB
Corel WordPerfect	1099	179MB	167KB	227MB	212KB
MS PowerPoint	374	199MB	800KB	132MB	53KB

Table 3. Conversion Resources.

We conducted our experiments utilizing open source tools to perform migration from proprietary and increasingly obsolete formats to modern web accessible renditions. The results indicated a high success rate with these tools and modest space/time requirements.

The collections studied are a subset of those distributed by the GPO. For example they do not include publications of the US Department Health and Human Services which include significant data in SETS (Statistical Export and Tabulation System) and additional formats. We also do not provide migration for GIS data because we are uncertain how to present these data in a browsable format.

The presented results reflect the operation of tools modified over many iterations to provide acceptable results in an automated environment suitable for server-side operation. Binary file types, in particular, present a challenge in scripted settings where care must be taken to differentiate between programmatic exceptions related to the tool or I/O subsystem, and those which represent legitimate failures to convert non-corrupt files. The relatively low failure rates presented in the previous section, along with per-file conversion times suitable for on-demand migration, provide strong support for the development of such systems.

Future work may include tunable profiles with timeouts optimized for file type and distribution within the archive, custom codec sets for multimedia translation, additional pre-migration tests for file integrity, and more rigorous conversion success criteria. In the context of the web application for which these migration profiles have been developed, a small percentage of failures is not unreasonable, as the user has access to the original files and may pursue additional strategies to translate these into an appropriate modern format.

The migration techniques we discussed are part of a larger system development effort. This system will provide collection browsing through migration and enable

execution of original documents in their contemporary execution environments utilizing emulation (virtualization). The tradeoffs between emulation and migration have been discussed elsewhere (Lorie, 2001; Rothenberg, 1999, 2000). Both migration and emulation are considered acceptable document preservation strategies by the Department of Defense and National Archives (Department of Defense, 2007; NARA, 2003). Our position is that emulation is required under circumstances where migration paths are non-existent or result in unacceptable loss (Mellor, 2003). Furthermore, migration has the potential for information loss. For example, there have been notable unintended information leaks through original documents that might be lost through migration (Hamburger, 2005; Jesdanum, 2005). Emulation is challenging for users because it requires detailed knowledge of obsolete software to access documents. Thus, our strategy is to utilize migration wherever possible and fall back on emulation as a safety strategy. We do believe emulation is an essential technology for preserving migration paths. As formats fall from use, the supporting migration tools may be discontinued (this is evident in Microsoft Office 2007). In subsequent work, we intend to explore the use of migration tools within emulated servers.

This work has used open source migration tools exclusively. We intend to study the use of commercial tools such as Microsoft Office to support migration and compare the commercial and open source results. Finally, many of the files cannot readily be supported through migration. We are continuing to develop our capability to provide file access through original software executing in an emulation environment.

References

Brown, A. (2006). *Digital preservation technical paper 1: Automatic format identification using PRONOM and DROID*. Technical report, National Archives, UK, March 2006.

Department of Defense. (2007). Design criteria standard for electronic record management software applications (DoD 5015.2-std). Retrieved November 20, 2008, from http://www.dtic.mil/whs/directives/corres/html/50152std.htm

Federal Depository Library Program. (2005). Minimum Technical Requirements (MTR), FDLP Desktop. Minimum technical requirements for public access workstations in federal depository libaries. Retrieved November 20, 2008 from http://www.fdlp.gov/computers/mtr.html

Hamburger, T. (2005, May 19). Nonpartisan testimony gets white house edit. *Los Angeles Times*.

Hedstrom, M. (Ed.). (2003). *It's about time: Research challenges in digital archiving*. Final Report. Workshop on Research Challenges in Digital Archiving and Long-Term Preservation. National Science Foundation, August 2003.

Jesdanum, A. (2005, May 2). Military mistake caused data leak. Associated Press.

Killian, T. J. (1984). Processes as files. In *USENIX Association. Proceedings of the Summer 1984 USENIX Conference*, pp. 203–207. Berkeley, CA: USENIX.

Lorie, R.A. (2001). Long term preservation of digital information. In *JCDL '01: Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital libraries*, pp. 346–352. New York: ACM Press.

Mellor, P. (2003). CaMiLEON: Emulation and BBC doomsday. *RLG DigiNews, 7*(2), 2003.

Mellor, P., Wheatley, P., & Sergeant, D.M. (2002). Migration on request, a practical technique for preservation. In *ECDL '02: Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries*, pp. 516–526, London, UK, 2002. Springer-Verlag.

NARA. (2003). *Endorsement of DoD electronics records management application (RMA) design criteria standard*, version 2. NARA Bulletin 2003-03. Retrieved September 30, 2006, from http://www.archives.gov/records-mgmt/bulletins/2003/2003-03.html

Pendry, J., & McKusick, M. (1995). Union mounts in 4.4BSD-Lite. In *Proceedings of the New Orleans Usenix Conference*, January 1995.

Pike, R., Presotto, D., Dorward, S., Flandrena, B., Thompson, K., Trickey, H., & Winterbottom, P. (1995). Plan 9 from Bell Labs. *Computing Systems*, *8*(3), pp. 221–254.

Rothenberg, J. (1999). *Avoiding technological quicksand: Finding a viable technical foundation for digital preservation*. Council on Library & Information Resources.

Rothenberg, J. (2000). *An experiment in using emulation to preserve digital publications*. Technical report, Koninklijke Bibliotheek, July 2000.

Write, C. P., & Zadok, E. (2004). Unionfs: Bringing file systems together. *Linux Journal, (128)*, pp. 24–29. December 2004.