PROJECT REPORT

Testing Software Tools of Potential Interest for Digital Preservation Activities at the National Library of Australia

Matthew Hutchins
Digital Preservation Tools
Researcher

30 July 2012

Published by
Information Technology Division
National Library of Australia
Parkes Place, Canberra ACT 2600 Australia

Contents

Summary

This is a report on a project conducted at the National Library of Australia in 2012 to investigate and test software tools of potential interest for digital preservation activities. The project focussed on testing file characterisation tools, and in particular file format identification and metadata extraction tools. The results of the project are intended to be useful in the specification and development of systems and processes for supporting digital preservation activities with software tools, and in particular in the context of the DLIR program at the National Library.

A significant test data set was collated, consisting of files from public sources and files from within the Library's collection. The test data set contained files with a large range of characteristics suitable for testing. A testing framework was developed to run the selected software tools over the test data set and record the results for further analysis.

Four file format identification tools were tested: File Investigator Engine, Outside-In File ID, FIDO and file/libmagic. This represents a mix of commercial and open source tools. The results were analysed from the point of view of comparing the tools to determine the extent of coverage and the level of agreement between them.

Five metadata extraction tools were tested: File Investigator Engine, Exiftool, MediaInfo, pdfinfo and Apache Tika. The results were analysed in terms of the number and range of meatdata items extracted for specific file subsets.

Based on the analysis of the test results, the following list of recommendations was derived.

List of Recommendations

1 Introduction

This is a report on a project conducted at the National Library of Australia in 2012 to investigate and test software tools of potential interest for digital preservation activities. The National Library of Australia defines the primary objective of digital preservation activities as maintaining the ability to meaningfully access digital collection content over time. Carrying out any activities on digital collection content requires software tools. There are many software tools in the world, and this project was part of the Library's ongoing objective to maintain awareness of what tools exist, what can they be used for, and how effective they are, for the various activities that comprise digital preservation. The project aimed to conduct practical comparative testing on a selected set of software tools. Other institutions have completed similar projects (see, for example, [14], [3], [4]).

One of the core activities of digital preservation is file characterisation: identifying and describing precisely what a file is and what its technical characteristics are. Accurate file characterisation is an essential prerequisite for maintaining meaningful access to digital content. For this reason, the project focussed specifically on testing file characterisation tools, and in particular on file format identification and metadata extraction tools. Other types of software tools that are of potential interest for digital preservation activities, but were not able to be covered within the time available for this project (but could be covered by future projects) include:

- Tools for bit-level characterisation of files, such as making cryptographic hashes
- Tools for detecting viruses within files
- Tools for checking the structure of files or validating them against specifications
- Tools for extracting content from files
- Tools for converting content from one file format to another
- Tools for describing structures and/or making manifests
- Tools for emulating other platforms

The National Library of Australia is currently in the process of specifying and sourcing a system to replace and enhance its infrastructure for managing digital content, a program known as the Digital Library Infrastructure Replacement (DLIR). The DLIR solution will be required to support the Library in its digital preservation activities. The results of the tool testing project, as contained in this report, are intended to influence the specification, evaluation and development of that system, both in a general sense of increasing awareness of how software tools can be used in the context of digital preservation, and in the specific sense of recommending particular tools or version of tools that could be considered for incorporation into automated workflows within the new system. Thus the recommendations of this report are written from the point of view of developing a system, or processes to be implemented within a system, for supporting digital preservation activities with software tools.

The methods and data sets used for conducting the tests are described first, followed by the results for different types of tools. Conclusions and recommendations derived from the results for the different types of tools are presented in a combined section at the end.

2 Methods

2.1 Test Data Sets

Testing software tools for digital preservation requires a collection of test files to use as input to the tools. A collection of test data sets was assembled for this project from both publically available and internal sources. The publically available sources serve as a point of comparison with other projects in the field. The internal ones ensure relevance to the aims of the library. The data is described below.

A significant feature of the collected data is that there are no well established and verified reference results against which the tools can be tested. That is, for the most part there is no accompanying metadata that would establish the file formats or characteristics of the files. That is left for the tools under test to determine. This in some ways constrains the type of tests that can be conducted, but also provides a realistic problem to be solved. It has been a point of discussion in the digital preservation community that there is a recognised need for a test data set with reference results for these types of projects (e.g. [19]). However, producing a data set that has many files, is accurately characterised, is representative of real data, and is freely distributable is quite a challenge, and there does not currently seem to be one available.

2.1.1 Govdocs1

Govdocs1 is a corpus of nearly one million files that has been collated for the purpose of supporting research in digital forensics and related disciplines [46]. The corpus is distributed as a set of 1000 zip files containing (approximately) 1000 files in each. The documents were collected at random from web servers in the ".gov" domain, so the corpus is dominated by modern web and office formats. The files are provided free of context and without further metadata. The entire Govdocs1 corpus was collected and included as test data for this project, although only a small part of the corpus had actually been examined at the time of writing this report.

2.1.2 Selections from Prometheus Ingest

Prometheus is the National Library's system for ingesting digital content from various kinds of physical media: CD, DVD, floppies, hard drives, etc. A typical example of an item in the system would be the contents of a CD that was included as supporting material for a book in the library's collection. Material from the Prometheus ingest is by definition material that is highly relevant to the Library's aims for digital preservation. A selection of material was chosen to be included as test data for this project, as follows:

- Objects that had been identified as containing files with formats that had not been successfully identified at the time of ingest.

- Objects manually selected to represent particular media types and file formats.
- A random selection of additional objects.
- Metadata files in an XML format for the selected objects, where available.

Note that each "object" could contain one or many files, depending on the contents of the media. Objects typically come from the repository in a compressed "master" format, which is one of a few types of disc or media image formats, depending on the type of media. The uncompressed media images were included as test objects along with the unpacked files they contained. The media images and objects combined totalled 169794 files. All the test files are included in the results of this report.

2.1.3 Special Selections

An Internet search for collections of samples of files of particular formats was conducted. The following collections were harvested and included as a test data set. They included a total of 6897 files, all of which are included in the test results.

- **Ahoy Disks**: A collection of disk images of disks that were included with Ahoy! magazine. These are disk images in .d64 format that contain material for Commodore 64 and possibly Amiga. From http://www.archive.org/details/ahoy-magazine-disks
- **archive.org**: Sample ARC and WARC files generated by the Internet Archive. From http://archive.org/details/ExampleArcAndWarcFiles
- **AutoCAD**: Samples of files in AutoCAD formats. From http://usa.autodesk.com, Data & Downloads, Sample Files.
- **croczilla.com**: A collection of SVG sample files. From http://croczilla.com/bits_and_pieces/svg/samples/
- **earlymacintosh.org**: A collection of Macintosh system disk images from Apple Macintosh Operating Systems before System 7. From http://earlymacintosh.org/disk_images.html
- **ebooks**: A selection of free samples of books in epub and other ebook formats. I ran KindleGen on some of the epubs to produce files in mobi format. From various web sites.
- **exif.org**: A collection of images from various digital cameras, with embedded metadata using the EXIF metadata format. From http://exif.org/samples.html
- **fileformat.info**: Collections of files illustrating various file formats. From http://www.fileformat.info/
- **ftp.apple.asimov.net**: A selection of disk images from older Apple computers, mostly for Apple II. From ftp://ftp.apple.asimov.net/pub/apple_II/images
- **graphicsmagick.org**: A selection of sample tiff images. From ftp://ftp.graphicsmagick.org/pub/tiff-samples/
- **Isartor testsuite**: A collection of sample files to be using for testing compliance with the PDF/A standard. All the files are valid PDFs that fail to be compliant with the standard. From http://www.pdfa.org/2011/08/download-isartor-test-suite/

- **J2KConformance**: A JPEG 2000 conformance document ITU T.803 including test data sets. From http://www.itu.int/rec/T-REC-T.803/en
- **jburkardt**: A well documented collection of sample files of various file formats, mostly scientific and graphics formats, collated by John Burkardt. From http://people.sc.fsu.edu/~jburkardt/data/data.html
- **openxmldeveloper.org**: Sample files and documentation from Open XML workshops. From http://openxmldeveloper.org/
- **remotesensing.org**: A collection of sample files in the GeoTIFF format. From ftp://ftp.remotesensing.org/pub/geotiff/samples
- **worldclim.org**: Climate data in ESRI and generic formats. From http://worldclim.org/current

2.1.4 Selections from Pandora Web Archive

Pandora is the National Library of Australia's curated web archive. A selection of material from a local backup of the archive was available for testing. From the roughly 25000 folders available in the copy, a random selection of 2500 folders was copied for the test, supplemented by a further selection of folders known to contain material that had caused problems in previous testing. This sample contained over 10,000,000 files. As it turns out, none of these files were examined for the results in this report, due to time constraints exacerbated by problems copying the data between file systems and a faulty hard drive interface card. This data set is left as a challenge for future work.

2.2 Focus of the Testing

The focus of this tool testing project was on the practical application of the selected tools within digital preservation workflows. The issue of raw speed and performance of tools has been adequately covered by previous projects in this field (e.g. [14]). Techniques to improve the performance of tools, such as managing the startup of the Java Virtual Machine for Java-based tools, or loading the file format database, have been presented, and should be used where practical. In a complex workflow, the overheads of managing the inputs and results of the tools, deciding which tool needs to be run on which data, ensuring the data is available to the tool, and the ability to run multiple tools in parallel, may well outweigh the significance of the performance of a single run of the tool, provided the tool is reasonably well behaved.

A more challenging issue is to investigate the value of tools in terms of the correctness, reliability, and usefulness of the results they produce. In the absence of a test data set for which all the "correct" answers are known in advance, it is not possible to directly measure the "correctness" of a set of tools. Instead, the primary aim of the testing in this project is to investigate what can be learned from comparison and integration of the results of different tools (both file format identification tools and metadata extraction tools), in terms of overall coverage, relative strengths and weaknesses, and outstanding problems.

A secondary aim of the project is to investigate and demonstrate approaches to managing complex workflows involving processing a large number of files and managing the results.

An example of the type of workflow of interest is as follows. A file is examined and determined to be a particular type of disk image. An appropriate disk image mounting utility is used to mount the image, and the files in the mounted volume are examined in turn. One is determined to be a zip archive. The archive is unpacked to a temporary area and the files in the archive are examined. One is determined to be a PDF, so a PDF metadata extractor is run. Another is a JPEG, so an image metadata extractor is run. When all the files in the archive have been examined, the temporary unpacked archive is deleted. When all the files in the mounted volume have been examined, the volume is unmounted.

As of the writing of this report, the primary focus of testing and comparing the individual tools has been given priority over the secondary aim, and the complex workflow issues remain as challenges for future work.

2.3 Software Framework for Testing

In line with the primary and secondary aims of the tool testing project, a software framework was developed to automate the testing workflow. The aims of the framework were:

- Process large numbers of files efficiently, using parallelism where possible.
- Behave predictably and robustly in the face of processing errors. Provide useful information about failures.
- Allow processing to be stopped and started. Do not expect the test framework to be always running for the duration of the testing.
- Produce and store results in a form that provides for analysis and comparison between tools.
- Demonstrate complex workflow issues.

The aims for robustness and interruptability lead to a need to have an explicit and saveable representation of process state. The desire for parallel processing means the state and results storage must be able to handle parallel access. The need to process large numbers of files means the state and results store must be able to handle large volumes of data. To resolve these issues, a MySQL database server is used as the central state and data store for the framework. This is an existing software solution that can be expected to robustly handle large volumes of parallel transactions.

The testing framework was developed as an application in the Python programming language. The core of the application is a producer/consumer queue implemented through a MySQL database job table. Multiple processes operate by taking jobs off the queue for processing, and pushing new jobs onto the queue when required. Jobs were developed to scan the filesystem for files to process, and to run each of the tools under test and record the results in database tables. Jobs in the job table have an associated state that tracks their progress through the system: ready to run, held, running, completed, failed.

The workflow issues around volume mounting and archive unpacking add a significant layer of complexity to job processing. Volume mount points and temporary disk space are scarce and shared resources that must be managed carefully. A job requiring a file on a mounted

volume can not run until the volume has been mounted. When (and only when) there are no jobs requiring a mounted volume it must be unmounted so the mount point can be used by other jobs. As of the writing of this report, the volume mounting issues had not been satisfactorily resolved, and dynamic volume mounting and archive unpacking were not enabled for the testing.

2.4 Test Platform

All the tests described in this report were conducted on a single computer: an HP workstation with an Intel Core i5 CPU running at 3.2GHz, 4GB of RAM, running the Windows 7 Enterprise 32 bit operating system. The workstation had additional 1TB and 500GB hard drives installed and an existing 230GB data partition. The data sets for the tests were spread across these three disks.

The MySQL database server used to manage the test framework and record the test results was run on the same computer as the tests.

3 File Format Identification Tools

3.1 File Format Identification

A computer file is a stream of bytes stored on a long-term storage medium. Typically, software applications write a file as a representation of some data structure existing in volatile memory, and in a manner that allows either the same application or other applications to read the file and reconstruct the same structure in memory at a later time. The "file format" is a description of how the stream of bytes in a file is organised so that it can be used to reconstruct the original structure. Most computer operating systems and file systems have no method of recording the file format of a file (although some do). Although there are conventions for naming files that help users remember the file format (such as using a three letter "file extension" like ".doc" or ".pdf"), these conventions are usually not enforced by the operating system, and in any case do not provide a unique identification.

Knowing the format of a file is a crucial step in digital preservation work at any level above preserving the byte stream. The file format determines, for example, what type of content or data is in the file, what applications can open or work with the file (and therefore whether we can open or work with the file at all given a fixed set of applications), how data and metadata can be extracted from the file, and what effect changes to the byte stream will have on our future ability to read data from the file. File format identification is the process of attempting to identify the file format based on clues that are present in the byte stream. There are multiple approaches to format identification, and different tools use different approaches.

In many cases, the relationship between a file and a file format is quite simple: the file is unambiguously an instance of a single file format. However there are also edge cases where the relationship is more complicated. A file can simultaneously be an instance of multiple file formats, or be a container for other files in multiple formats, or not be a valid instance of any file format. Because file format identification is a process of inference based on clues, it will

not always lead to a unique, correct decision. Ultimately, the only true arbiter of the "correct" format of a file is the author's intent in creating it.

3.2 Overview of Tools Included in the Test

3.2.1 Selection Criteria

Software tools for file format identification were chosen for the test based on several criteria.

- The tools needed to be mature and/or professional, with a reasonable expectation that they could be used in a serious production workflow in the Library in the near future.
- The tools could be integrated into a batch/automated workflow without excessive difficulty, and could be expected to operate in a multi-process or multi-threaded framework (this generally favours APIs and command line tools over GUI tools).
- There should be a balance between testing tools that hadn't been included in other similar reported tests, and overlapping enough with other tests that some comparison would be possible.
- The number of tools to be tested needed to be manageable.

The tools were chosen from a list of potential tools previously identified as being of interest, plus some additional internet searching. The search was not necessarily comprehensive. Four tools were chosen for the tests, described below. In the descriptions, the "potential" strengths and weaknesses refer to what was known or expected about the tool before testing, from documentation, marketing material, etc.

3.2.2 File Investigator Engine

Background

File Investigator Engine is a commercial API for file format identification, developed by Forensic Innovations, Inc. [12] It is a C library API, with versions available for Windows, Macintosh, Solaris and Linux. The Windows version was tested, with a Python wrapper generated by SWIG [24]. The product comes from a background in digital forensics. The API was used under the conditions of a trial/evaluation license.

Potential Strengths

- Commercially developed and supported API
- Distributed as C API with shared library, therefore easy to integrate into software development projects
- Good control over which features to use
- Fast and robust
- Large and varied range of file formats supported
- Combines several methods of file identification

- Reports file formats by unique identifier (easy to handle in software)
- Reports accuracy of identification (high, medium, low)
- File format database includes summary and background information on file formats, common extensions, mime types, associated software
- Can collect metadata, file statistics, and hashes

Potential Weaknesses

- Fixed set of file formats identified for each release, updates are released quarterly
- This is an API for software developers, not an end-user tool (there is an end-user tool available that uses the engine)
- Single file at a time identification (no support for compound objects)

Settings Used for the Test

- Software Version: 2.3.3.
- Analysis Stages: header pattern match, inter-file pattern match, byte value distribution pattern match, file extension match, interpret file & verify identification.
- Directory Add: no.
- Checksum Add: none. The calculation of an SHA-1 hash was tested initially with no problems, but as this was not the main focus of the testing it was omitted from the main testing for performance reasons.
- Get Details: yes. This means that the File Investigator Engine is also being tested as a metadata extraction tool.
- Text File Search Depth: default (32 bytes).
- Summary Length: 255 characters.
- Filter CR/LF: no.

In correspondence with Forensic Innovations, Inc. I asked whether customers could influence the choice of file formats included in the quarterly updates. They replied:

> We strive to constantly improve File Investigator, by implementing the recommendations that we receive from our clients. Yes, we like to receive requests for file types to be added. In that way, we continue to add the most pertinent types. When you request a new file type, or improvements to be made to an existing file type, any specifications and example files that you can provide will help the process.

3.2.3 Outside-In File ID

Background

Outside-In File ID is a commercial API for file format identification that is part of a suite of software tools called Outside In Technology distributed by Oracle [22]. It is a C library API, with versions available for Windows and Unix. The Windows version was tested, with a

Python wrapper generated by SWIG [24]. The product has a focus on office/business file formats. The API was used under the conditions of a trial/evaluation license.

Potential Strengths

- Commercially developed and supported API
- Distributed as C API with shared library, therefore easy to integrate into software development projects
- Fast and robust
- Reports file formats by unique identifier (easy to handle in software)
- Good coverage of office formats (compared to File Investigator Engine)
- May integrate well with other tools in the Outside-In range (not investigated)

Potential Weaknesses

- Fixed set of file formats identified for each release
- Smaller overall set of recognised file types (compared to File Investigator Engine)
- No report of accuracy of identification
- This is an API for software developers, not an end-user tool
- Single file at a time identification (no support for compound objects)

Settings Used for the Test

- Software Version: 8.3.7.
- Normal/Extended: extended results for text files.

3.2.4 FIDO

Background

FIDO is an open-source software tool for file format identification developed/distributed by the Open Planets Foundation [20]. FIDO uses the PRONOM database [17] of file formats that is also used by DROID [10], but has a simpler command-line interface than DROID, making it easier to incorporate into batch workflows. FIDO is distributed as Python source, and was incorporated into the test framework through its Python API.

Potential Strengths

- Free, open source software
- Compatible with DROID/PRONOM file format identifiers, link to the digital preservation community
- Reports file formats by unique identifier (easy to handle in software)
- Can be locally extended with new file formats
- Command line operation, suitable for incorporation into batch process or Python program

- Some support for Container file types (as defined by DROID)

Potential Weaknesses

- Probably slow (compared to compiled language tools)
- Smaller overall set of recognised file types (compared to File Investigator Engine)
- Limited reporting of accuracy of identification, and reports inaccurate results

Settings Used for the Test

- Software Version: 1.0.0
- Signature files: Default as released with version 1.0.0. Matches DROID signatures V55.
- Recurse into zip files: no.
- Deep scan of containers: yes (that is, "nocontainer" setting is false).

3.2.5 Unix file Command/libmagic

Background

The "file" command is included as standard in most Unix distributions. It is based on a C library called libmagic. There is an open source version of libmagic and the file command for inclusion in linux distributions [7],[8]. There are various approaches to compiling it for Windows. The testing used a version of the file command included with Cygwin [1], with a Python wrapper.

Potential Strengths

- Free, open source software
- Fast and robust
- File: Command line operation, suitable for incorporation into batch process
- libmagic: C API with shared library, therefore easy to integrate into software development projects
- Attempts to classify a large range of source code and other text-based files
- Variety of non-Windows (and Windows) file types
- Possible to extend with new file types
- Reports metadata for some file types

Potential Weaknesses

- Reports textual description rather than unique identifier (hard to handle in software)
- No report of accuracy of identification
- Single file at a time identification (no support for compound objects)
- Difficult to compile for non-unix systems

Settings Used for the Test

- Software version: 5.09, Cygwin version.
- Brief mode: yes (helps in parsing the output).
- MIME types: run once without and once with the MIME types flag to get both outputs.
- All other settings are defaults.

3.2.6 Other Tools Included in the Test

Two other file format identification tools were included in the testing. These were not considered part of the main test, but were included in case they provided any information not provided by the other tools.

TrID is a free command-line tool for file format identification [23]. TrID was not included as part of the test because it had been previously decided that the license conditions for TrID prevent it from being used in production in the library. It operates in a similar manner to the Unix file tool. It was the only tool to correctly identify the Sibelius score file format, and correctly or incorrectly some Claris Works files.

Optima SC File ID is another freeware command-line file-format identification tool [21]. It operates in a similar manner to the Unix file tool. It proved to be slow and buggy and did not provide any information not provided by the other tools.

Some of the metadata extraction tools included in the test also include a file format identification component.

3.3 Test Results: Tool Performance

The four major file format identification tools tested all performed reliably and robustly during the testing. After integration with the testing framework had been completed, there were no unexpected crashes, hangs, or unpredictable behaviours recorded during the tests. This includes each tool being run on hundreds of thousands of files in the test dataset.

As described previously, timing and/or CPU usage tests were not included in this project.

3.4 Test Results: Data Processing

The raw file identification test results were stored in a MySQL database. Each tool had its own results table. A row in the result table matched a file entry from the test dataset with the output of a tool.

The File Investigator Engine tool result table had one result entry per file tested, which included metadata results as well as the file format identification as a "description number" and "accuracy". The description numbers are a unique identifier for each format, which can be used as an index into separate tables giving the name, description and background information for the file format. The description number 0 is reserved for unidentified files.

The accuracy is a number: 0,1,2 or 3 representing not identified, or identified with low, medium or high accuracy respectively.

The Outside-In File ID tool result table had one result entry per file tested, which had file format identification as a "type id". The type id numbers are a unique identifier for each format, which can be used as an index into a separate table of type names. The type number 1999 is reserved for unidentified files.

The FIDO tool result table had one or multiple result entries per file tested. Each entry included a "match type", which is a string indicating what type of match was used for this entry: "fail" for unidentified files, "signature" for files matched by a signature rule, "container" for files matched by a container rule, and "extension" for files matched by a file extension rule. Multiple entries were produced if a file matched multiple rules. This was usually the case for files matched by extension. The result entry also included a "puid" which is a PRONOM unique identifier, a string that identifies a file format. Other information from the signature database was also included, such as the MIME type and apple UTI if known.

The libmagic tool result table had one result entry per file tested. The result was in the form of a textual description of the file format. As metadata is included in the description, the description strings are not, in general, unique identifiers for file formats, nor do they have a static internal structure. If no other description applies, the file is identified as "data". The libmagic result table also records a MIME type.

To extract value from these millions of individual result table entries, the data needed to be processed, summarised, and compared. A "head to head" comparison report was developed that combined the results from all four result tables, and grouped and summarised the entries according to the result reported by each tool for each file ("head to head" used here in the colloquial sense of referring to an item-by-item comparison between competing products or sporting results). For the purposes of the report, the different tools were determined to have given a result for a file as follows: for File Investigator Engine, a non-zero description number with an accuracy rating of 2 or more; for Outside-In File ID any result other than 1999; for FIDO, a match produced by a "signature" or "container" rule; and for libmagic, any result other than "data". For the libmagic tool (with no unique identifiers) the results were grouped according to the first 12 characters of the description, and described by the alphabetically minimum and maximum description. This was effective at distinguishing the different results.

For each of the four tools the report lists the distinct file formats identified by that tool and the number of files identified with that format. Then for each format identified by one tool, it breaks down, for each of the other three tools, which distinct file formats were assigned to the set of files identified with that format. So, for example, the File Investigator Engine identified 10 files as having the format "Amiga Interchange File Format Image" (with unique identifier 6). Of those 10 files, Outside-In File ID did not identify all 10 (or identified them as "Unknown format" with identifier 1999); FIDO identified all 10 as "Interchange File Format Interleaved Bitmap" (identifier "fmt/338"); and libmagic identified them with a range of

results (alphabetically) from "iff data, ilbm interleaved image, 124 x 124" to "iff data, ilbm interleaved image, 300 x 150".

This presentation can tell us, for example, where one tool claims to identify files that another doesn't. It then also forms the basis for developing a mapping between identifiers in one tool and identifiers in another tool.

The report was augmented with two types of manual checking. Based on the descriptions reported by the tools for each file format, a table was added to the database that recorded where, in my opinion, the tools were describing the same format. From the example above, File Investigator Engine format 6, "Amiga Interchange File Format Image", is likely to be describing the same format as FIDO format "fmt/338", "Interchange File Format Interleaved Bitmap", which is likely to be the same format as all libmagic results that start with the text "iff data, ilbm interleaved image". There is plenty of ambiguity there, and this is a manual process based on opinion, but it is the beginning of a definition of what it means for the tools to agree or disagree on a file's format.

The second type of manual checking was to look at individual files to discover if it could be determined from the file itself or its context within the test data set whether the assigned format was correct or not. It was not possible to do this for all or even a significant portion of the files under test (because of the large number of files in question), so the manual checking concentrated on a few interesting cases, such as where only one tool assigned a format to a file. Again, the manual checking process was a matter of opinion and inference and is not necessarily correct. It also highlighted a central issue in file format identification, which is that in many cases the actual format is ambiguous, and so "correct" or "incorrect" are not adequate to describe the results of manual investigation. I used the following categories to record manual checking results:

- **Wrong**: the file format assigned by the tool and the actual file format seem to be unrelated.
- **Wrong Subclass**: the tool assigned a specific format which is a subclass of a more generic format. The file seems to be an instance of the more general format, but not of the specific subclass.
- **Correct Class**: the tool assigned a format which is a generic format that has more specific subclasses. The file appears to be a member of one of the more specific subclasses, but this was not reported.
- **Undecidable Subclass**: the tool assigned a specific format which is a subclass of a more generic format. The file seems to be an instance of the more general format, but it is not decidable whether it is actually a member of the specific subclass, because the file or the subclass is ambiguous. (This is discussed in more detail below).
- **Correct Subclass**: the tool assigned a specific format which is a subclass of a more generic format. The file appears to be a member of both the general class and the specific subclass. It is possible that an even more precise subclass could have been assigned.
- **Best**: the tool assigned a specific format and the file seems to be an instance of this format, and it is unlikely that there is an even more specific subclass that could have been assigned instead.

In some cases, manual checking of a file indicated that it had a specific format that was not reported by any of the tools. A table of manually identified file formats was added to the database so that the manual result could be recorded.

In general, there were two types of manual investigations that could provide information about a file. For text files (including plain English text, source code files, HTML and XML, etc.), the files were opened in a text editor and examined to see if they were instances of a specific subclass. For other types of files, in some instances, the context of the file within the test collection could be used. For example, in the "special selections" test data, files were often included specifically as examples of a particular file format, and so that format was assumed to be correct for the file. In the "Prometheus" test data, files were present in the context of a directory structure that occasionally contained documentation describing the associated files, or meaningful directory names, or groups of files of the same type, or specific applications that could open certain types of files. Hypotheses about the actual file format were derived from these types of clues.

A second report was developed to examine the files not identified by any of the four tools. These were broken down by file extension, and by location within the test collection. This information was used to drive further manual investigation of individual files, to determine if useful patterns could be found within the unidentified files.

The complete data analysis reports are too large to include in this document. They are available as separate HTML files. Appendix A presents a table of just the individual file formats identified by File Investigator Engine within the test data set.

3.5 Test Results: General Agreement

The "head to head" report demonstrates that there is a set of file formats that are consistently well identified. If a tool has a rule for that format, then it will consistently recognise those files, in agreement with the other tools. Where it has no rule, then it will report unidentified or a non-specific or incorrect answer. These file formats are typically graphics, media and archival formats that have been developed with robust specifications for interchange between applications; specific types of system files; or file formats that just happen to have a strong and unique signature that makes them stand out.

If we define complete agreement as cases for which all four tools identify precisely the same set of files as having precisely one file format that appears to be the same format, for more than 3 files in the test data, then there are remarkably few examples of complete agreement. At the time of writing (more files from the test set were still being examined) the only examples were:

- GEM VDI Paint Image, 9 files in the test set
- Audio/Video Interleaved (AVI), 287 files in the test set
- BinHex archive, 6 files in the test set
- Windows shortcut, 27 files in the test set
- Adobe PhotoShop Image, 62 files in the test set

If we loosen the definition of agreement to include cases where three or more of the tools assign only a small set of inter-related formats, with occasional misclassifications, then there are more examples of general agreement to choose from. Some more examples are:

- TIFF images (but there are little-endian and big-endian versions that can be distinguished)
- GIF images (but there are at least two versions, 1987a and 1989a, that can be distinguished, and some mis-classification)
- Windows bitmaps (but there are at least two versions and some mis-classification)
- QuickTime movies (but there are at least two versions that can be distinguished)
- MPEG-4 media file (but there are audio and video versions that can be distinguished)
- dBase III databases (but there are confounding versions and some mis-classifications)
- Rich text format documents (but there are several versions and some mis-classification)
- Truetype fonts (but there are at least two versions that can be distinguished)
- Microsoft Outlook PST file (there was only one example in the test data, but all four tools agreed on it)
- Microsoft Windows compiled HTML help files (three tools agreed, FIDO did not identify)

3.6 Test Results: Sources of Confusion

The "head to head" report also demonstrates several recurring themes where there was disagreement, mis-classifications, or difficulty in even deciding whether the tools agreed or not, or were correct or not.

3.6.1 Text Files

A text file is a file made up of bytes that can be interpreted as characters in a character set, and are thus often intended to be readable by both humans and machines. Even this definition of a text file is noticeably weak; there is in fact very little distinction between a binary file and a text file, other than the range of bytes used. From a software point of view, the distinction between a text file and a binary file is that the text file is read and interpreted by a process of lexical analysis and parsing, rather than directly loading bytes into memory. Text files play an important role in computing, as the basis of most programming languages, scripts, batch files, simple data files, log files, program output, etc. Some text files have a well-defined internal structure, but others are less structured.

Given that a file has been identified as (probably) being a text file, the file format identification tools will often then try to further identify it as belonging to a particular subclass of text file. The methods that are used to do this tend to be probabilistic and approximate, and so the success rate is low. That is, they are sometimes right and sometimes wrong, and so this distinction can not be relied upon, and is therefore useless.

3.6.2 HTML, XML and Internet Source Files

A particularly noticeable case of text file confusion is found in HTML, XML, and other Internet source files: CSS, Javascript, PHP, etc. These cause confusion because:

- As Internet usage has grown, these types of files have become more common. In test datasets derived from Internet sources, these will be a majority of files.
- Although modern HTML has a proper specification, historically web browsers have accepted and displayed a large variety of files and tolerated mistakes, missing pieces and loose formatting without complaint. Thus it is sometimes very difficult to determine if a file actually should or should not be classified as an HTML file, even with close inspection.
- Formats like HTML, XML and SGML are closely related and use a very similar syntax, and so it is difficult to distinguish between them.
- HTML can and frequently does have other formats embedded within it. It is possible to have an HTML file that contains mostly CSS or Javascript or PHP. How should it then be classified?
- As HTML and XML become more widely used, it becomes more common to find sample text files that have a few HTML-like formatting instructions included within the text. These are either intended to be included within or displayed as an HTML file, or they are just using a familiar syntax to convey a familiar concept. Again, it is difficult to distinguish between a text file with some HTML tags and an actual HTML file, even if the distinction makes sense.

Another example of Internet text files that are hard to distinguish (or provide a workable definition of) are ascii email files, email inboxes, news posts, news digests, mime files, etc.

3.6.3 Weak Magic

A "magic number" or file format signature is a sequence of bytes that appear at a particular place within a file (usually at the beginning) that can be used to identify the file format. Sometimes they are a deliberate part of the file format specification, and sometimes they have been determined by analysing examples of particular file formats. Magic numbers or signatures are the main method used to identify binary files. The utility of a magic number to distinguish between file formats is determined by the probability or frequency that the same sequence of bytes will appear "by accident" in other files.

As examples, the magic number for a PBM image is the two ASCII characters "P1". This resulted in the misclassification of text files starting with these characters as PBM images. Similarly, a set of text files starting with the characters "MED" (including MEDIA RELEASES and MEDICAL documents) were misclassified by libmagic as "MED_Song" files (which appears to be an Amiga music format, we can guess its magic number is the three ASCII bytes "MED"). Note that libmagic seems to have a dependency on the file format extension in this regard. Text files that did not have the three character extension "txt" were more frequently misclassified in this way.

3.6.4 Files That Look Like Other Files

It is often the case that a particular file format is an extension of another format, uses another format as a container, or is a combination of formats. An SVG file is an XML file which is a text file. A Java archive is a PKZip archive. A WAV sound is also a RIFF file. An MP3 file can be preceded by ID3 tag data. A self-extracting archive is both an executable and an archive. A printer job file can contain a Postscript document. In all these cases, there are multiple correct answers to the question of what the actual format of the file is, and so different identification tools can (and do) give different answers without any of them necessarily being wrong.

Some files can also accidentally look like they belong to a particular file format, when they don't. The "weak magic" examples from the previous section are examples. Generic, unstructured binary files could, by chance, appear to be something else.

Although there were no examples identified in the test dataset, presumably malicious software will be designed to look like something other than what it is, and so is another candidate for causing confusion in file format identification.

3.6.5 Version Proliferation

One of the most frequent causes of disagreement among the tools tested was the cases where there were multiple versions of a particular file format. This is likely to happen for any file format that has achieved both wide usage and longevity. Some tools will identify just the broad class of the file (such as Adobe Portable Document Format or Graphics Interchange Format) and leave the version number as metadata or not relevant, while others will attempt to identify a precise version number (e.g. Portable Document Format 1.4, GIF 1987a). In some cases this is not a serious problem, but it does make it impossible to establish a one-to-one mapping between the outputs of two different tools.

In other cases, it makes it difficult to resolve the question of whether two tools are in agreement about a format without knowing the precise details of the format in question. For example, there was considerable variability in the descriptions of JPEG files. Files described by one tool as "JPEG File Interchange Format" were variously described as "Progressive JPEG", "JPEG 2000", "JPEG Tiled Image Pyramid", "Raw JPEG Stream", "Exchangeable Image File Format", "JPEG 2000 codestream", "JPEG 2000 image data", "JPEG Image Data" and "JPEG Image Data, JFIF Standard". Only an expert in the JPEG standard could determine if these tools agreed or disagreed on the file format.

3.6.6 Microsoft Office and Similar Formats

The Microsoft Office Suite has achieved widespread use, longevity, implementation on different platforms, and has a wide variety of both core and auxiliary file formats. This means that Microsoft Office files caused frequent confusion among the file format identification tools tested, as examples of version proliferation, files that look like other files, and even files that look like HTML and/or XML. For example, for the set of 6000+ files identified by File Investigator Engine as "MS Excel Worksheet/Template (OLE)", Outside-In File ID identified eight different versions of Microsoft Excel ("2.x", "3.0", "4.0", "5.0/7.0", "97/98/2004", "2000",

"2002", "2003"), FIDO classified most as "OLE2 Compound Document Format" but also suggested various versions of Excel, Powerpoint and Word, and libmagic split them into "Microsoft Excel Worksheet" and "Composite Document Format". Files identified by File Investigator Engine as "MS PowerPoint Slides (XML)" were identified by Outside-In File ID as "PowerPoint 2000 HTML", but by FIDO and libmagic as just HTML files.

Although there were fewer examples in the test data sets, other office software suites such as WordPerfect and Lotus exhibited some of the same problems, but to a lesser degree.

The widespread use of office software, for business and for preparation of documents of all types, means that office documents can be expected to make up a significant portion of many document collections, and thus dealing with the uncertainty in office document identification takes on an extra level of importance.

3.6.7 False Positives

There are some tool outputs that appear to be positive file format identifications, but are in fact just another way of saying the format is unidentified: "data file", "binary file", "compressed data", "random data", "headerless data", "raw data", "wiped data", even some of the generic text file formats. These types of outputs confuse the issue of whether the file has actually been identified.

3.6.8 Simply Unidentifiable Files

Some files are simply not instances of well defined file formats, and can not be accurately identified by tools that assume they are. The test datasets contained examples of files that were corrupt, damaged or random binary data. There were unstructured and fragmented text files. There were binary files with no identifying characteristics associated with an accompanying executable or separate description.

The assumption that any file can and should be automatically identified will lead to confusing results for files that are not identifiable.

3.7 Test Results: Unique and Special Cases

Here are some examples of file formats for which File Investigator Engine gave the only definitive answer (some of these have been confirmed, others might be incorrect):

- ArcExplorer Project
- Borland Paradox Primary Index
- DB/TextWorks Database (various associated file formats)
- DVM Movie
- Erdas Image (HFA)
- InstallShield (various associated file formats)
- Khoros Visualization Image

- Macintosh Desktop Database
- Mac OS X Folder Information
- Macromedia Director File (Intel)
- MapInfo Map
- MS Access Lock File
- MS Setup Package
- MS Windows Security Catalog
- NOAA-PMEL BBIS Data
- Radiance 3D Scene Octree
- Sybase Compressed Data
- Toolbook Database
- Zebra Metafile

Here are the only examples of file formats for which Outside-In File ID gave the only definitive answer (neither of these has been confirmed):

- WordPerfect 4.2
- Mac PowerPoint 3.0

Here is the only example of file formats for which FIDO gave the only definitive answer (unconfirmed):

- ESRI Shapefile (some associated file formats, not ArcView GIS Shape)

Here are some examples of file formats for which libmagic gave the only definitive answer (some of these have been confirmed, others might be incorrect):

- d64 image
- epub ebook data (other tools identified as PK ZIP file)
- pcx ver. 2.5 image data
- rdi acoustic doppler current profiler (adcp)
- squeak image data

3.8 Test Results: Unidentified Files

Given the large number of files in the test datasets, there were remarkably few files for which no tool offered any definitive result. The unidentified files report did not include files for which the only identification offered was determined to be incorrect, or was in fact a false positive (as described in Section 3.6.7). About 1-2% of the files were included in the unidentified files report, and a small number of file formats account for those that were investigated and appeared to be identifiable:

- DCL file associated with CALS raster
- Dr. Halo image

- NAPLPS: North American Presentation Layer Protocol Syntax
- BOB ray tracing format
- GMOD
- TRIB binary triangle format
- Apple disk image
- Apple Lisa disk image
- Big TIFF little endian
- ESRI Arc/Info binary file (one of several types)
- The BIN part of a cdrdao BIN/TOC pair
- AutoPlay Media Studio menu file
- Borland Graphics Interface device driver
- Binary electronic navigation chart data
- ERDAS GIS file format
- OziExplorer binary data file
- FileMaker dictionary file
- Sibelius music score

A majority of the remaining files are probably unidentifiable, in the sense described in Section 3.6.8.

3.9 Test Results: Summary

The following figures are summary statistics of file identification, as of the writing of this section of the report (further testing is still underway). For these figures, identification is as described in Section 3.4 for the "head to head" report.

Table 1 File identification summary

Total number of files examined	314250
Files identified by File Investigator Engine	307546
Files identified by File Investigator Engine only	5196
Files identified by Outside-In File ID	278824
Files identified by Outside-In File ID only	642
Files identified by FIDO	217127
Files identified by FIDO only	20
Files identified by libmagic	283175
Files identified by libmagic only	727
Files identified by one of four tools only	6581
Files identified by two of four tools only	32586
Files identified by three of four tools only	66829
Files identified by all four tools	203607
Files not identified	4643

4 Metadata Extraction Tools

4.1 Metadata Extraction

Metadata about a digital object can be broadly categorised into extrinsic and intrinsic metadata. Extrinsic metadata is generated and stored external to the object. Intrinsic metadata is included in the byte stream of the object itself. Metadata extraction could be considered as a process of making intrinsic metadata extrinsic. That is, metadata extraction tools typically read certain items of data from the byte stream of a file and present them in a different format.

Typically, file-level metadata extraction tools only operate on intrinsic metadata (although they will usually report some extrinsic metadata from the file system). In the case of intrinsic metadata, the distinction between what is metadata and what is content is not well defined, and will depend on the objectives of the user. Thus, metadata extraction becomes a special case of content or data extraction. One of the challenges of metadata extraction, then, is to define what content is to be extracted, based on the context of use. In the context of the library in general, metadata could be useful across several areas, including classification, cataloguing, search, delivery, and presentation. In the more limited context of digital preservation, the relevant metadata is that which provides information about requirements for maintaining access to the content in the file. In many cases this will be further elaboration of the file format, such as version, byte order, encoding, codec, number of channels, etc. or dependencies such as fonts used.

4.2 Overview of Tools Included in the Test, and Some That Weren't

The selection process for metadata extraction tools was somewhat more limited than the process for file format identification tools. A small number of tools were chosen based primarily on what could be quickly found and integrated into the testing framework, and also considering a balance of extractors for different types of files. As described above, the File Investigator Engine was run with settings that included extraction of some metadata. The other four tools tested were:

- ExifTool [11]. This is a Perl library and command-line tool for reading, writing and editing metadata from a wide variety of file types. Developed initially to extract metadata stored in image files, it has been extended to cover many more file types, including audio and video files, office files, PDFs and containers. Version used for testing: Windows command line version 8.85. Run with the "-a", "-ee" and "-m" flags.
- MediaInfo [16]. This is an open source utility for extracting technical and tag data from video and audio files. There are GUI and command-line versions available. Version used for testing: Windows command line version 0.7.54. Run with no flags for default output.
- The pdfinfo tool from the Xpdf toolkit [27]. Xpdf is an open source collection of utilities for handling PDF documents. Although developed for the X Windows system running on

Unix, Windows versions of the command-line utilities are available. The "pdfinfo" utility extracts content from PDF files. Version used for testing: Windows binary version 3.03. Run with the "-box" flag to print basic information and the page bounding boxes.

- Apache Tika [2]. This is an open source toolkit from the Apache Software Foundation written in Java. It wraps and combines a number of other libraries for parsing various sorts of documents and extracting content from them. It has parsers for HTML, XML, text, various office formats, PDF, some containers, and some media. Version used for testing: runnable Java archive version 1.1. Run with the "-m" flag to extract only metadata.

All four tools were run as Windows command-line versions, wrapped into the test framework with Python scripts to launch the tools and parse the outputs. The tools were selectively run based on the outputs of the file format identification stage, and in particular the file format identified by the Outside In File ID tool.

Several additional tools were identified that would have been good candidates for testing, but were excluded simply due to time constraints in preparing wrappers to incorporate them into the test framework. These should be considered candidates for testing in future projects. The tools were:

- The National Library of New Zealand Metadata Extractor [18]. This is a tool specifically for extracting preservation metadata from digital files. It is already in use in the National Library of Australia's Prometheus system. Note that the architecture of the Metadata Extractor makes it difficult to robustly wrap it into a workflow that uses parallel processing.
- Outside-In Content Access [22]. This is a commercial API from the same suite as the Outside-In File ID API tested in the file format identification tests. It would be expected to have good coverage of office and publishing formats.
- Adobe XMP SDK [1]. This is an SDK for working with Adobe's Extensible Metadata Platform. This is a specific metadata format that can be serialised and embedded into certain types of image, media and document files.
- ImageMagick "identify" tool [13]. This is a command line tool that has wide coverage of image formats.

4.3 Test Results: Tool Performance

In contrast to file format identification tools, metadata extraction tools must read and make sense of the content stored in a file. They are therefore more vulnerable to being derailed by badly formed files. The four metadata extraction tools all reported errors and warnings of various sorts while running. The Apache Tika tool frequently terminated with a stack trace or an error code. ExifTool and MediaInfo both failed to terminate on at least one file.

4.4 Test Results: Result Subsamples and Item Counts

To process the metadata results, four specific result subsamples were defined based on the file formats identified by the Outside In File ID tool. The four subsamples were:

- Image files: GIFs, TIFFs, JPEGS, etc.
- Multimedia files: sound files, videos, animations.
- Office files: Microsoft Word, Excel, PowerPoint.
- PDFs: Adobe Portable Document format, PDF, PDF/A, PDF/X.

Three tools (ExifTool, FIE and Tika) provided metadata for files in all four subsamples. MediaInfo only reported on Image and Multimedia files. PDFInfo only reported on PDFs. The numbers of files in each subsample where each tool reported at least one metadata item are shown in Table 2.

Table 2 Metadata: Number of files reported by tool and subsample

Tool	Image	Multimedia	Office	PDF
ExifTool	110,675	7,466	85,409	137,341
FIE	110,587	3,791	85,123	137,284
Tika	110,523	7,466	83,858	137,014
MediaInfo	110,675	7,466		
PDFInfo				137,341

The results of running the metadata extraction tools were homogenised into a set of item/value pairs, where "item" names a metadata item and "value" gives its value. The items were by no means uniform across the different tools. The total numbers of items reported by each tool for each subsample are shown in Table 3.

Table 3 Metadata: Number of items reported by tool and subsample

Tool	Image	Multimedia	Office	PDF
ExifTool	4,274,189	121,486	2,750,952	2,605,467
FIE	680,453	11,534	469,251	787,432
Tika	2,949,361	41,452	1,324,505	1,385,416
MediaInfo	752,848	99,533		
PDFInfo				2,577,597

The ranges of number of items (minimum and maximum) per file reported by each tool for each subsample are shown in Table 4.

Table 4 Metadata: Minimum and maximum number of items per file reported by tool and subsample

Tool	Image	Multimedia	Office	PDF
ExifTool	4–298	6–75	4–150	4–229
FIE	1–15	2–13	1–28	1–17
Tika	2–223	2–16	2–134	3–109
MediaInfo	1–12	1–42		
PDFInfo				13–23

The numbers of *distinct* items reported by each tool for each subsample are shown in Table 5.

Table 5 Metadata: Number of distinct items reported by tool and subsample

Tool	Image	Multimedia	Office	PDF
ExifTool	1686	202	999	1598
FIE	15	17	23	13
Tika	1154	13	997	1069
MediaInfo	21	85		
PDFInfo				57

ExifTool reported more items in total and more types of items across all four subsamples.

4.5 Test Results: File Format Identification

ExifTool and Tika report the MIME type of the file as one of the metadata items. Thus they could also have been considered for inclusion in the File Format Identification Tools section. Table 6 shows the number of file MIME type identifications for each tool and each subsample that were broadly consistent and inconsistent with the file format assigned by Outside In File ID.

Table 6 File Format Identification results of metadata tools

Tool/Subsample	Consistent	Inconsistent
ExifTool/Image	110596	0
ExifTool/Multimedia	7466	0
ExifTool/Office	84670	6
ExifTool/PDF	137205	12
Tika/Image	110409	114
Tika/Multimedia	7283	183
Tika/Office	83761	97
Tika/PDF	137005	9

In this table, a "consistent" result is one in the same file format class or subclass as the reference, and an "inconsistent" result is in a different class or subclass (it excludes cases where no MIME type was suggested). For ExifTool, the inconsistent office files were individual instances of Microsoft Office compound documents classified into a different subclass (e.g. PowerPoint as Word or Excel). Manual inspection showed several of them to be badly or strangely formed documents. The inconsistent PDFs were classified as "image/jpeg". Manual inspection showed the majority of them were lacking the "%PDF" tag at the start of the file, although they opened in Adobe Acrobat Reader. For Tika, the inconsistent image files were unidentified JPEG 2000 and Windows Bitmap files (that is, identified with only "application/octet-stream"). The inconsistent multimedia files were unidentified MPEG-2 videos. The inconsistent office files were largely unidentified older versions of Office files, plus some of the same ambiguous file formats as for ExifTool. The inconsistent PDFs were classified as "text/plain", and were mostly missing the "%PDF" tag.

4.6 Test Results: What Metadata is Extracted?

From Table 5 we can see that the tools vary widely in what items of metadata they extract for each of the subsamples. From Table 4 we can see that the amount of metadata extracted per item also varies widely, typically depending on the file format and the amount of content metadata that is actually present. The following tables present the metadata items most commonly extracted by each of the tools for each of the subsamples, excluding the extrinsic metadata from the file system (file size, permissions and dates), and file format identification.

Table 7 Most frequently extracted metadata items by ExifTool for the Image subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Image Size	“200x132”, “520x330”	110596
Image Width	“200”, “520”	110596
Image Height	“132”, “330”	110596
Bits Per Sample	“1”, “8”, “16”, “8 8 8”, “8 8 8 8 8”	78720
Y Resolution	“72”, “100”, “182.88”, “0”, “inf”	78496
X Resolution	“72”, “100”, “182.88”, “0”, “inf”	78496
Resolution Unit	“inches”, “cm”, “None”	78469
Encoding Process	“Baseline DCT, Huffman coding”, ”Progressive DCT, Huffman coding”, “Extended sequential DCT, Huffman coding”	76635
Color Components	“3”, “1”, “4”	76635
Y Cb Cr Sub Sampling	“YCbCr4:4:4 (1 1)”, “YCbCr4:2:0 (2 2)”, “YCbCr4:2:2 (2 1)”, “YCbCr4:4:0 (1 2)”, “YCbCr4:1:1 (4 1)”	68366
JFIF Version	“1.00”, “1.01”, “1.02”, “2.01”	67587
Compression	“JPEG (old-style)”, “Uncompressed”, “Deflate/Inflate”, “None”, “T6/Group 4 Fax”, “T4/Group 3 Fax”, “LZW”, “Adobe Deflate”, “JPEG”, “PackBits”, “CCITT 1D”, “JPEG 2000”, “4-Bit RLE”	37310
APP14 Flags 0	“[14]”, “[14], Encoded with Blend=1 downsampling”, “Encoded with Blend=1 downsampling”, “(none)”	33735
APP14 Flags 1	“(none)”	33735
DCT Encode Version	“100”, “101”	33735
Color Transform	“YCbCr”, “Unknown (RGB or CMYK)”, “YCCK”	33735

Table 8 Most frequently extracted metadata items by FIE for the Image subsample

Metadata Item Name	Metadata Value Examples	Number of Files
X Resolution (dots)	“200”, “520”	110559
Y Resolution (dots)	“132”, “330”	110559
# of Color Bits	“24”, “16”, “8”, “4”, “3”, “1”, “65536”	110540
File Version	“JFIF”, “Exif”, “Photoshop 3.0”, “89a”, “87a”, “Adobe_Photoshop2.5”, “JFXX”, “II”, “Radius”, “FPXR (APP2)”	95963
Format Version (major)	“100”, “101”, “102”, “200”, “201”, “300”	75200
Software	“Adobe Photoshop CS Windows”, “File written by Adobe Photoshop 4.0”, “Adobe Photoshop 7.0”, “LEAD Technologies Inc. V1.01”, “Created with The GIMP”, “Digital Camera DX-10 Ver1.00”, “Lovely smoke effect here, of which I'm secretly very proud.”	33054
Image Compression	“0” [uncompressed], “2” [4bit RLE], “3” [LZW], “9” [CCIT/3 1-D], “10” [FAX CCITT Group 3], “11” [FAX CCITT Group 4], “12” [JPEG], “13” [PackBit]	28017

Date Created/Sent	“2005:05:09 16:01:42”	22895
Copyright	“J P Bowen”, “KODAK DC240 ZOOM DIGITAL CAMERA”, “(C) by RDC-5300 User”, “Copyright (C) 1996 by the Library of Congress, All rights reserved. : #198 The book of Mormon”	11501

Table 9 Most frequently extracted metadata items by Tika for the Image subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Image Width	“200 pixels”, “520 pixels”	71756
Image Height	“132 pixels”, “330 pixels”	71756
Data Precision	“8 bits”	71756
Number of Components	“1”, “3”, “4”	71756
Component 1	“Y component:Quantization table 0, Sampling factors 1 horiz/1 vert”, “Y component:Quantization table 0, Sampling factors 2 horiz/2 vert”	71438
Component 2	“Cb component:Quantization table 1, Sampling factors 1 horiz/1 vert”, “Y component:Quantization table 1, Sampling factors 1 horiz/1 vert”	63635
Component 3	“Cr component:Quantization table 1, Sampling factors 1 horiz/1 vert”, “Cr component:Quantization table 2, Sampling factors 1 horiz/1 vert”	63635
Compression CompressionTypename	“lzw”, “deflate”, “BI_RGB”, “BI_RLE4”	31846
Compression Lossless	“true”	24225
Compression NumProgressiveScans	“1”, “4”, “7”	24225
Data SampleFormat	“Index”, “UnsignedIntegral”	24225
Dimension ImageOrientation	“Normal”	24225
Chroma BlackIsZero	“true”	24225
Chroma ColorSpaceType	“RGB”, “GRAY”	24225
Chroma NumChannels	“1”, “2”, “3”, “4”	24225
Resolution Unit	“Inch”, “cm”, “(No unit)”	24021
X Resolution	“72 dots per inch”, “180 dots per inch”, “75 dots per (no unit)”, “71 dots per cm”, “475466304/16777216 dots per cm”	24010
Y Resolution	“72 dots per inch”, “180 dots per inch”, “75 dots per (no unit)”, “71 dots per cm”, “475466304/16777216 dots per cm”	24010

Table 10 Most frequently extracted metadata items by MediaInfo for the Image subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Format	“JPEG”, “GIF”, “LZ77”, “RGB”, “TIFF”, “Raw”, “CCITT T.4”, “LZW”, “PackBits”, “CCITT Group 3”, “JPEG 2000”, “BDAV”, “RLE”, “MPEG Audio”, “LATM”	110589
Width	“200 pixels”, “520 pixels”	110586
Height	“132 pixels”, “330 pixels”	110586
Compression mode	“Lossy”, “Lossless”	98106
Bit depth	“8 bits”, “24 bits”, “1 bit”, “8bits / 8bits / 8bits”	91181
Chroma subsampling	“4:4:4”, “4:2:0”, “4:2:2”, “2:1:4:2”	66623
Format/Info	“Graphics Interchange Format”, “Portable Network Graphic”, “Blu-ray Video”, “Run-length encoding”	24229
Format profile	“89a”, “87a”, “No PAT/PMT”, “MPEG-4”, “Layer 2”	19416
Stream size	“142 KiB (100%)”, “138 KiB (100%)”	4378

Display aspect ratio	“1.000”, “0.961”, “3:2”, “4:3”	4377
Color space	“Y”, “RGB”, “CMYK”, “YUV”, “Grey”	2085
Codec ID	“jp2”, “jpx”	9

Table 11 Most frequently extracted metadata items by ExifTool for the Multimedia subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Duration	“20.53 s”, “0.08 s”, “0:00:30”, “0:00:41”, “17.05 s (approx)”, “0:04:16 (approx)”	6712
Sample Rate	“11025”, “44100”	3607
Image Size	“800x600”, “250x300”, “640.05x480.2”	3301
Image Width	“800”, “250”, “640.05”	3301
Image Height	“600”, “300”, “480.2”	3301
Frame Rate	“15”, “12”, “23.976 fps”, “16.667”	3191
Frame Count	“308”, “1”, “131479”	3007
Flash Version	“4”, “5”, “6”, “7”, “8”, “9”, “10”	2722
Compressed	“False”, “True”	2722
Bits Per Sample	“8”, “16”, “0”, “32”	2147
Encoding	“Microsoft PCM”, “MP3”, “Microsoft IEEE float”	2147
Num Channels	“1”, “2”	2147
Avg Bytes Per Sec	“11025”, “22050”, “176400”	2147

Table 12 Most frequently extracted metadata items by FIE for the Multimedia subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Image Compression	“0” [uncompressed], “1” [8bit RLE], “4” [Cinepak Codec], “15” [not defined by FIE?], “842094169” [not defined by FIE?]	3009
Format Version (major)	“4”, “5”, “6”, “7”, “8”, “9”, “10”	2722
# of Sound Bits	“8”, “16”	786
Sound Compression	“1” [PCM], “15” [MPEG 1.0 layer 3], “16” [MPEG 2.0 layer 3]	786
# of Sound Channels	“1”, “2”	785
Sound sampling Rate in Hz	“44100”, “22050”, “11025”	785
Title	“gltrv-”, “(12)”, “Song of Cooloola”	569
X Resolution (dots)	“320”, “340”	310
Y Resolution (dots)	“200”, “344”	310
Time Length (1/100 of a second)	“1000”, “500”	303
Frames/second	“1270”, “1000”,	294
# of Color Bits	“8”, “12”, “16”, “24”	287
# of Frames/Images	“100”, “50”	287

Table 13 Most frequently extracted metadata items by Tika for the Multimedia subsample

Metadata Item Name	Metadata Value Examples	Number of Files
xmpDM	“audioSampleRate:600”, “audioSampleType:8Int”, “genre:Blues”, “audioCompressor:MP3”, “audioChannelType:Stereo”	3527
channels	“1”, “2”	2799
samplerate	“11025.0”, “44100.0”, “44100”	2799
bits	“8”, “16”, “32”	2041
encoding	“PCM_UNSIGNED”, “PCM_SIGNED”, “PCM_FLOAT”	2041
Author	“”, “null”	1193
title	“Chord Ex (1)”, “”, “null”, “gltrv-”, “Song of Cooloola”	1193
version	“MPEG 3 Layer III Version 1”, “MPEG 3 Layer III Version 2”	758

Table 14 Most frequently extracted metadata items by MediaInfo for the Multimedia subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Duration	“20s 533ms”, “83ms”, “6mn 14s”	6203
Overall bit rate	“31.4 Kbps”, “741 Kbps”, “1 441 Kbps”, “104 Mbps”	6203
Sampling rate	“11.025 KHz”, “8 000 Hz”	4832
Channel(s)	“1 channel”, “2 channels”	4832
Stream size	“58.2 KiB (74%)”, “612 KiB (100%)”, “5.89 MiB (100%)”	4778
Bit rate	“23.2 Kbps”, “598 Kbps”, “104 Mbps”	4700
Bit rate mode	“Variable”, “Constant”	4226
Overall bit rate mode	“Variable”, “Constant”	4091
ID	“0”, “1”, “150”, “189 (0xBD)-32 (0x20)”	3721
Bit depth	“8 bits”, “16 bits”, “24 bits”, “32 bits”	3373
Display aspect ratio	“4:3”, “0.833”, “3:2”, “1.600”	3341
Width	“800 pixels”, “250 pixels”	3341
Height	“600 pixels”, “300 pixels”	3341
Codec ID	“IV50”, “cvid”, “0x00000001”, “IV32”, “YV12”, “jpeg”, “rpza”, “20”, “MJPG”, “3”, “SVQ3”, “ima4”, “CRAM”, “sowt”	3099
Format profile	“Advanced Simple@L3”, “Quick Time”, “Layer 3”, “MP@LL”, “MP@ML”, “Main@Main”, “Layer 2”, “Pro”, “MP@HL”, “LC”, “Float”, “AP@L1”	2547
Frame rate	“15.000 fps”, “12.000 fps”, “23.972 fps”	2178
Format settings, Endianness	“Little”, “Big”, “Float”	2072
Compression mode	“Lossy”, “Lossless”	1941
Format version	“Version 1”, “Version 2”	1911

Table 15 Most frequently extracted metadata items by ExifTool for the Office subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Last Modified By	“”, “kim taylor”, “ktaylor”, “ocio”, “U.S. Census Bureau – Population Division”, “zolec300”, “Employee Name”	83809
Code Page	“Windows Latin 1 (Western European)”, “Unicode UTF-16, little endian”, “Mac Roman (Western European)”, “Windows Japanese (Shift-JIS)”,	83438

	“Unknown ()”	
Links Up To Date	“No”, “Unknown ()”	83310
Scale Crop	“No”, “Unknown ()”	83310
Heading Pairs	“Title, 1”, “Worksheets, 1”, “Fonts Used, 5, Design Template, 1, Embedded OLE Servers, 1, Slide Titles, 23”, “Introduction to Metallogenic Belt and Mineral Deposit Maps for Northeast Asia, 0”	83279
Create Date	“2005:08:25 20:50:00”, “2004:01:27 23:25:00”	82379
Modify Date	“2005:08:25 20:54:00”, “2004:01:27 23:25:00”	82154
Title Of Parts	“BLACK HUCKLEBERRY”	82086
Hyperlinks Changed	“No”, “Unknown ()”, “Unknown (-1)”	80239
App Version	“10.3501”, “9.6926”, “11.8107”	80239
Shared Doc	“No”, “Yes”, “Unknown ()”, “Unknown (-1)”	80239
Author	“”, “USDA”, “J. Scott Peterson”, “Default”, “zolec300”, “tempuser”, “FEMA Employee”, “XP Installer”, “Employee Name”	80177
Software	“Microsoft Word 10.0”, “Microsoft Word 9.0”, “Microsoft Office Word”, “Microsoft Power Point”, “gnumeric”, “SoftArtisans ExcelWriter”	79111
Company	“USDA”, “NCI”, “institut”, “Private”, “Self”	77545

Table 16 Most frequently extracted metadata items by FIE for the Office subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Character Set	“9” [Double Byte], “8” [Single Byte]	84423
Author/From	“USDA”, “J. Scott Peterson”, “Default”, “zolec300”, “tempuser”, “FEMA Employee”, “XP Installer”, “Employee Name”	72330
# of Words	“297”, “2106”	64157
Title	“BLACK HUCKLEBERRY”	63138
Template	“Normal.dot”, “Normal”, “C:\Program Files\Microsoft Office\Templates\Blank Presentation.pot”	51393
# of Pages	“1”, “2”, “3”, “6”, “21”	40343
# of Characters	“1648”, “3152”	38516
# of Frames/Images	“23”, “21”, “1”	25851

Table 17 Most frequently extracted metadata items by Tika for the Office subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Last-Author	“”, “kim taylor”, “ktaylor”, “U.S. Census Bureau – Population Division”, “zolec300”, “FEMA Employee”, “XP Installer”, “Employee Name”	82958
Creation-Date	“2005-08-25T20:50:00Z”, “2004-01-27T23:25:00Z”	81549
Last-Save-Date	“2005-08-25T20:54:00Z”, “2004-01-27T23:25:00Z”	81091
Author	“”, “USDA”, “J. Scott Peterson”, “U.S. Census Bureau - Population Division”, “zolec300”, “tempuser”, “FEMA Employee”, “XP Installer”, “Employee Name”	79513
Application-Name	“Microsoft Word 10.0”, “Microsoft Word 9.0”, “Microsoft Office Word”, “Microsoft PowerPoint”, “gnumeric”, “SoftArtisans ExcelWriter”	78454
Company	“USDA”, “NCI”, “institut”, “Private”, “Self”	76715
title	“BLACK HUCKLEBERRY”	67086
Revision-Number	“6”, “2”, “4”	64274
xmpTPg	“NPages:2”, “NPages:1”, “NPages:21”	64071
Word-Count	“297”, “2106”	63740

Template	“Normal.dot”, “Normal”, “C:\Program Files\Microsoft Office\Templates\Blank Presentation.pot”	54832
Edit-Time	“600000000”, “1200000000”	53094
Last-Printed	“2005-08-24T20:41:00Z”, “2004-01-22T20:12:00Z”	48278
subject	“”, “Vaccinium deliciosum Piper”, “Adobe Captivate template”	42556

Table 18 Most frequently extracted metadata items by ExifTool for the PDF subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Linearized	“Yes”, “No”	137205
PDF Version	“1.0”, “1.1”, “1.2”, “1.3”, “1.4”, “1.5”, “1.6”, “1.7”, “1.39999”	137205
Page Count	“1”, “4”, “13”	136758
Create Date	“2002:04:25 13:02:24Z”, “2008:06:12 09:48:27-04:00”	135379
Producer	“”, “Acrobat Distiller 5.0 (Windows)”, “Acrobat Distiller 8.1.0 (Windows)”, “FDFMerge Lite 5.0.4 Windows SPDF_1096+ May 3 2004”, “Acrobat PDFWriter 3.02 for Windows NT”, “QuarkXPress(R) 7.3”	135005
Creator	“US Census Bureau”, “edocslib”, “PScript5.dll Version 5.2.2”, “J.David Wilson”, “ArcInfo 8.1 (Fri Mar 16 11:31:29 PST 2001)”	122290
Modify Date	“2002:04:25 14:14:17-03:00”, “2005:05:05 21:19:55Z”	120182
Title	“Census 2000 Profiles”, “Microsoft Word - 48428.doc”, “BLACK HUCKLEBERRY”, “C:\GAINSrvr\data\RS7044 Biofuel Annual_2007.PDF”	114372
Author	“US Census Bureau”, “edocslib”, “m1jas06”, “J.David Wilson”, “Admin”, “vhoward”	99200
Metadata Date	“2002:04:25 14:14:17-03:00”, “2005:05:05 21:19:55Z”	75565
Format	“application/pdf”, “application/postscript”, “application/x-indesign”, “Microsoft Word”, “image/jpeg”, “Microsoft Word 10.0”	65017
Document ID	“uuid:7798a834-eb15-4ea8-a026-6960a5507c53”, “adobe:docid:indd:a54ac75f-93b6-11dc-9c11-fae69af0e1e2”	64865
XMP Toolkit	“Adobe XMP Core 4.0-c316 44.253921, Sun Oct 01 2006 17:14:39”, “XMP toolkit 2.9.1-13, framework 1.6”, “3.1-702”	63851
Creator Tool	“PScript5.dll Version 5.2.2”, “Acrobat PDFMaker 8.0 for Word”, “QuarkXPress(R) 7.3”	59349

Table 19 Most frequently extracted metadata items by FIE for the PDF subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Format Version (major)	“100”, “110”, “120”, “130”, “140”, “150”, “160”, “170”	137283
Date Created/Sent	“2002/04/25 13:02:24Z”, “2005/05/05 17:19:55-04'00'”	120332
Program Name	“Acrobat Distiller 5.0 (Windows)”, “Acrobat Distiller 8.1.0 (Windows)”, “Acrobat PDFWriter 4.0 for Windows NT”, “QuarkXPress(R) 7.3”	115761
Date Saved	“2002/04/25 14:14:17-03'00'”, “2003/03/13 22:52:03”	104021
Software	“XPP”, “PScript5.dll Version 5.2”, “ACOMP.exe WinVer 1b43 jul 14 2003”, “PageMaker 6.5”	99225
Title	“Census 2000 Profiles”, “Microsoft Word - 48428.doc”, “BLACK HUCKLEBERRY”, “Type over this text with your Abstract Title (use initial caps)”	85047
Author/From	“US Census Bureau”, “edocslib”, “m1jas06”, “J.David Wilson”, “Admin”	76547

Table 20 Most frequently extracted metadata items by Tika for the PDF subsample

Metadata Item Name	Metadata Value Examples	Number of Files
xmpTPg	“NPages:1”, “NPages:4”, “NPages:13”	137005
producer	“Acrobat Distiller 5.0 (Windows)”, “Acrobat Distiller 8.1.0 (Windows)”, “Acrobat PDFWriter 4.0 for Windows NT”, “QuarkXPress(R) 7.3”	135118
created	“Thu Apr 25 23:02:24 EST 2002”, “Fri May 06 07:19:55 EST 2005”	134983
Creation-Date	“2002-04-25T13:02:24Z”, “2005-05-05T21:19:55Z”	134983
creator	“XPP”, “PScript5.dll Version 5.2”, “ACOMP.exe WinVer 1b43 jul 14 2003”, “PageMaker 6.5”	120245
Last-Modified	“2002-04-25T17:14:17Z”, “2005-05-05T21:19:55Z”	119988
title	“Census 2000 Profiles”, “Microsoft Word - 48428.doc”, “BLACK HUCKLEBERRY”, “Type over this text with your Abstract Title (use initial caps)”	114321
Author	“US Census Bureau”, “edocslib”, “m1jas06”, “J.David Wilson”, “Admin”	98574

Table 21 Most frequently extracted metadata items by PDFInfo for the PDF subsample

Metadata Item Name	Metadata Value Examples	Number of Files
Encrypted	“no”, “yes (print:yes copy:yes change:no addNotes:no)”, “yes (print:yes copy:no change:no addNotes:no)”	137341
Tagged	“no”, “yes”	137341
Form	“none”, “AcroForm”, “XFA”	137341
ArtBox	“0.00 0.00 612.00 792.00”	137341
TrimBox	“0.00 0.00 612.00 792.00”	137341
MediaBox	“0.00 0.00 612.00 792.00”	137341
BleedBox	“0.00 0.00 612.00 792.00”	137341
CropBox	“0.00 0.00 612.00 792.00”	137341
Optimized	“yes”, “no”	137341
Page size	“612 x 792 pts (letter)”, “792 x 1224 pts”, “595 x 842 pts (A4)”, “596.16 x 778.56 pts”	137341
Pages	“1”, “4”, “13”	137341
PDF version	“1.0”, “1.1”, “1.2”, “1.3”, “1.4”, “1.5”, “1.6”, “1.7”	137341
CreationDate	“04/25/02 13:02:24”, “05/05/05 17:19:55”	135804
Producer	“Acrobat Distiller 5.0 (Windows)”, “Acrobat Distiller 8.1.0 (Windows)”, “Acrobat PDFWriter 4.0 for Windows NT”, “QuarkXPress(R) 7.3”	135311
Creator	“XPP”, “PScript5.dll Version 5.2”, “ACOMP.exe WinVer 1b43 jul 14 2003”, “PageMaker 6.5”	120415
ModDate	“04/25/02 14:14:17”, “05/05/05 17:19:55”	120310
Title	“Census 2000 Profiles”, “Microsoft Word - 48428.doc”, “BLACK HUCKLEBERRY”, “Type over this text with your Abstract Title (use initial caps)”	114490
Author	“US Census Bureau”, “edocslib”, “m1jas06”, “J.David Wilson”, “Admin”	98736

4.7 Test Results: Garbage In, Garbage Out

Many file formats allow effectively arbitrary metadata fields to be stored within the byte stream, and the metadata extraction tools will find these and report them as metadata items.

The value of these fields depends entirely on the quality and consistency with which it is entered. For example, the "Author" metadata in a document will in some cases actually identify the author, but in others will contain a system user name, a generic name (such as "Employee" or "User"), the name of the author of a different document or template that the current one was created from, or just garbage text. Thus in most cases these metadata fields will not be consistently usable in any automated process, and so there is little value in extracting them. Other metadata fields are tags specific to the workflow processes of the organisation that produced a document, and have little value outside of that context.

Overwhelmingly, the metadata extraction test results demonstrate the importance of filtering the output of the tools to retain only the data that is likely to be relevant and useful to the context in which it is being used. Broadly speaking, technical metadata that is a required component of the file format (for example, the width, height and bit depth of an image) is the data that is extracted most consistently.

5 Unpacking and Mounting Utilities

Although not included as part of the test framework, some tools were indirectly tested as part of preparing the test data and are worth further comment.

5.1 WinCDEmu

WinCDEmu [26] is an open source Windows tool for mounting disc images. WinCDEmu can be integrated into the Windows desktop, but also has a batch mounting version that enables it to be used from scripts. WinCDEmu can be used to mount some of the cdrdao BIN/TOC images from the Prometheus collection.

5.2 Universal Extractor

Universal Extractor [25] is actually a collection of open source extraction tools with a unified GUI wrapper. Although the wrapper itself had some usability limitations, the Universal Extractor package was an easy way to download and install a very useful set of individual tools for the Windows platform, including:

- **InfoZip UnZip**. The only Windows tool I found that could unzip some of the larger zip files in the Prometheus collection.
- **bin2iso**. Can read a BIN/CUE pair and convert them to an ISO image.
- **7-zip**. Can extract files from an ISO image, and can extract files from a disk image of a USB flash drive, as well as several other container formats.
- **extract**: Can extract data from images of floppies.

5.3 cdrdao

cdrdao [5] is a tool for recording audio or data CDs. The Prometheus collection uses the cdrdao BIN/TOC format for some of its disc images. The Windows distribution of cdrdao

includes a command line utility called "toc2cue" which can convert the BIN/TOC to BIN/CUE format, which is recognised by other tools. One approach to accessing audio tracks from an audio CD in the Promethues collection is to use toc2cue to convert the BIN/TOC to a BIN/CUE, and then use bin2iso to extract the audio as WAV files.

6 Conclusions and Recommendations

6.1 File Format Identification Tools

6.1.1 Applicability

The conclusion from the testing results is that automatic file format identification both works and doesn't work. For a majority of cases, for files in one of the well known and well defined file formats, file format identification works well and gives useful results. However, regardless of the tools chosen, it is important to recognise that automatic file format identification remains inherently uncertain. There are files that can not be effectively recognised or distinguished with the methods available, and tools do make mistakes. Meaningful output on the accuracy of the identification reported by a tool may help to identify where the mistakes are more likely, but does not resolve the issue.

Therefore, I conclude that automatic file format identification should be used, but used with care. Where practical, the results should be confirmed by other means before significant decisions are made that rely on an automatic identification of the file format. The most important information that can be used to accurately determine the file format is the metadata about the processing history of the file: where did it come from? how was it created? why was it created? is it part of a collection or series of files with a particular purpose? what format did the author or producer intend it to have? This is metadata that is not typically available to file format identification tools, and is not part of the file system. In many cases, if this information were available, it could be used to bypass file format identification tools altogether.

Recommendation

1 Prefer using process history metadata to automatic identification if possible.

In particular, where files are being created by the Library as part of an established workflow, the file format specification should be derived from that workflow, and there would be no reason to run automatic identification tools. Where files are being gathered by a web harvest, the MIME type assigned by the web server might be considered an authoritative format classification.

Recommendation

2 Use automatic identification where process history is unavailable, but use the results as a guide or a clue, not a definitive answer. Implement manual checking processes where practical.

Recommendation

3 Develop processes and workflows that allow manual characterisation of files based on extrinsic information to interoperate with or override automatic identification. These could include, for example, easy batch update of the format metadata of related groups of files based on a shared processing history.

6.1.2 Tool Selection

Four file format identification tools were tested for this project. Any of the four could form a useful basis for a solution, but the clear stand-out tool was File Investigator Engine, in terms of coverage of file formats, flexibility of the API, accuracy reporting, providing background information on the formats, and collecting metadata. The potential weakness of this tool is the fixed set of file formats in each release that can not be locally extended. This weakness can be addressed by combining it with a more flexible tool.

Recommendation

4 Consider using File Investigator Engine as the basis for an automatic file identification solution, or at least use it as the standard by which to judge other proposed solutions.

Recommendation

5 Where a tool with a fixed set of formats is used as the basis for a solution, supplement it with an extensible open source or locally developed tool that can easily have formats added to cover cases of local significance that aren't covered by the commercial tool. Submit these cases back to the developer of the main tool for potential inclusion in a future release.

Of the two extensible, open source, tools tested, FIDO and libmagic, libmagic clearly has the better range of formats. However, FIDO benefits from its links to PRONOM and the digital preservation community and its use of an easily recognisable identifier. DROID could of course be considered instead of FIDO, although new versions of DROID are said to be more difficult to integrate into a batch process [14].

It should be technically possible to capture some of the pattern information from the libmagic database and export it in a format usable by FIDO, or visa-versa. The core file identification part of FIDO is a relatively small amount of Python code. This demonstrates that the same approach could be used as a basis for a locally developed tool if desired. It would technically be possible (but tedious) to modify the input files for libmagic such that it could output a unique format identifier instead of, or in addition to, its textual description.

Integrating two or more tools will necessarily require an approach to unifying the file format identifiers used to label file formats. Where new formats are being added locally because the existing tools don't recognise them, there will of course be no suitable identifier defined in any of the existing tools. It will be necessary to develop an identifier scheme that can include as a subset the identifiers from the existing tools in addition to new, locally defined, identifiers. An issue with PRONOM identifiers is that, being managed by an external authority, they will be slow to allocate additional identifiers.

The data from tool testing provides a basis for developing a mapping between the identifiers used by the four tools tested if required.

Recommendation

6 Develop an extensible solution based on one of the existing open source tools. Choose which one based on how easily it will be to integrate it into the surrounding development context. Only use it when the primary tool is unable to identify the file.

Recommendation

7 Develop an approach to unique identifiers for file formats that will allow integration of the results of multiple tools.

Recommendation

8 Find or develop a process to capture and import file format signatures from other tools.

6.1.3 Remaining Uncertainty

The four file format identification tools tested, and several others that were considered for testing, all concentrate on the issue of determining the format of a single file, considered in isolation. In some cases, files and directories grouped together make up a single compound object, and an identification and characterisation of this compound object would be a useful thing to have. This project did not identify any tools that could perform identification at the level of compound objects stored as separate files, although the Container signatures of FIDO/DROID are heading in that direction for compound objects packaged in a container format.

Recommendation

9 Find or develop a solution to the problem of identifying compound digital objects.

The test results confirm that some classes of files are hard to identify accurately. Text files, in particular, can exhibit considerable ambiguity. It is unrealistic to expect precise classification of these files, and where it is attempted the results will frequently be misleading. Internet text formats (HTML, XML, email, news) will be a common source of this confusion.

Recommendation

10 Develop policies and strategies for dealing with ambiguous file formats; deal with broad classes where possible, and expect precise classifications to be erroneous.

The test results also confirm that some files are not classifiable, or won't be classified by the current versions of the tools in use.

Recommendation

11 Develop policies and strategies for dealing with unidentified files, because there will be some.

File format identification tools work from a set of known file formats. New versions of the tools will recognise more file formats, and may have better rules and methods for identifying previously known file formats. Therefore, different versions of a tool may produce different results for the same file.

Recommendation

12 Develop policies and strategies for revising file classifications when new versions of tools are released.

6.2 Metadata Extraction

6.2.1 Applicability

Metadata extraction tools work well for a specific type of metadata: technical metadata that is intrinsic to the bitstream and part of the file format. Different metadata tools are specialised for particular file formats. Metadata tools tend to extract a lot of additional content that is not reliably useful, and needs to be filtered. Although the tools tested were generally robust, the requirement to read and decode the file format makes them vulnerable to being derailed by badly formed or unusual files.

Recommendation

13 Use metadata extraction tools to extract intrinsic technical metadata from files with specific formats. Choose specific tools to be used for specific formats. Other types of metadata will need to be generated or captured from workflow processes.

Recommendation

14 Develop filters for the output of metadata extraction tools to avoid capturing large volumes of unusable metadata.

Recommendation

15 Develop processes that allow for metadata extraction tools to fail or not terminate on badly formed files.

With reference to the "Repository Object Preservation Metadata Baseline for the DLIR Project" [15], note that most specified metadata items are extrinsic and will not be able to be extracted from files. Some intrinsic technical metadata is specified, but the document suggests this is incomplete ("it is envisaged that Collection Areas will provide additional file format specific technical information for their areas of expertise.") The following items may be extractable in some cases, but only for a limited range of file formats, and only if the creating application chooses to record the data.

- Creating Application, Creating Application Version
- Creation Date and Time
- Encryption
- Geo-tag Coordinates
- Recording Equipment Information

- Colour Profile, Colour Sub-sampling
- CODEC, CODEC Version
- Tracks / Channels, Track / Channel Relationships
- Bitdepth, Alpha Channel
- Bitrate, Sampling Rate, Frame Rate
- Compression
- Byte Order
- Interlacing
- Picture Format, Aspect Ratio

6.2.2 Tool Selection

Of the five tools tested, ExifTool was the most comprehensive across the spectrum of file types examined, in terms of the number of items reported. However, it is clear that not all "items" of metadata are equally useful. The current project did not have an explicit baseline of required technical metadata to test against, so it was not possible to determine which tools, if any, met the actual requirements for metadata extraction. As the metadata available for extraction is largely determined by the file format, such a baseline would need to broken down by file format. If a comprehensive statement of requirements was available, then further testing could determine the extent to which each tool met the requirements, and this could be used to choose between tools for specific purposes. The selection of tools for testing in this project was not comprehensive, and there would be benefit in testing more widely (but only if the statement of requirements was available).

Recommendation

16 Consider using ExifTool for general metadata extraction. Of the tools tested, it reported the most items and the largest variety of items across the classes of files examined.

Recommendation

17 Develop specific intrinsic technical metadata requirements for individual file formats, and conduct further testing on a wider range of tools to determine the most appropriate tool (or combination of tools) for each format.

6.3 Processing Strategies

Section 2.2 described one of the aims of this project as investigating complex processing workflows for complete object characterisation that involve iterative and/or recursive application of multiple tools: file format identification, unpacking and mounting, and metadata extraction. As described previously, these issues were not deeply investigated within the time available for the project. However, it is clear from the work that was done that these are important issues, and not tackled simply.

Recommendation

18 Complete characterisation of digital objects requires complex processing strategies involving recursive application of multiple tools, careful resource management, robustness in the event of badly formed files and other causes of tool failure, and efficient processing. This requirement should be taken into account when developing or evaluating a digital preservation solution.

The distinction between what is and isn't a container file is becoming increasingly blurred. For example, a modern Microsoft Office document is actually a compound object consisting of multiple files packaged using the PKZIP container format. Although this file should probably be characterised as a single document, it may be necessary to unpack it to extract metadata or find other hidden dependencies.

Recommendation

19 Develop policies and strategies for dealing with container and archive formats and other compound objects: which containers should be unpacked and how are they to be characterised?

The tool testing framework developed for this project, described in Section 2.3, was a preliminary attempt to investigate and implement some of the processing strategies needed for complex digital preservation scenarios. The framework was built around a central job queue implemented in a MySQL database table. This turned out to be a significant bottleneck that prevented the system from operating at maximum efficiency. It is probable that the overhead involved with managing the cooperation of multiple processes outweighed the benefits of parallel execution. Future work along these lines would do well to adopt either a much simpler approach (accepting the restrictions that imposes), or solve the processing issues more effectively.

References

[1] Adobe XMP Developer Center web site: http://www.adobe.com/devnet/xmp.html. Accessed 28th May 2012.

[2] Apache Tika web site: http://tika.apache.org/. Accessed 28th May 2012.

[3] Asger Blekinge. "Identification tools, an evaluation", blog post on "blekinge's blog", 23rd February 2012. http://www.openplanetsfoundation.org/blogs/2012-02-23-identification-tools-evaluation, accessed 16th July 2012.

[4] Kevin Ashley [not confirmed]. "Assessment of file format testing tools". Unpublished JISC report 2006, available from: http://www.jisc.ac.uk/media/documents/programmes/preservation/daat_file_format_tools_report.pdf. Accessed 18th July 2012.

[5] cdrdao web site: http://cdrdao.sourceforge.net/. Accessed 29th May 2012.

[6] Cygwin web site: http://www.cygwin.com/. Accessed 2nd May 2012.

[7] die.net linux man page for file command: http://linux.die.net/man/1/file. Accessed 2nd May 2012.

[8] die.net linux man page for libmagic library: http://linux.die.net/man/3/libmagic. Accessed 2nd May 2012.

[9] Digital Corpora Govdocs1 web site: http://digitalcorpora.org/corpora/files. Accessed 14th May 2012.

[10] DROID project web site on SourceForge: http://droid.sourceforge.net/. Accessed 2nd May 2012.

[11] ExifTool by Phil Harvey web site: http://www.sno.phy.queensu.ca/~phil/exiftool/. Accessed 28th May 2012.

[12] Forensic Innovations Inc. web site: http://www.forensicinnovations.com/. Accessed 2nd May 2012.

[13] ImageMagick web site: http://www.imagemagick.org/script/index.php. Accessed 28th May 2012.

[14] Johan van der Knijff and Carl Wilson. "Evaluation of Characterisation Tools. Part 1: Identification". SCAPE Technical Report, November 2011. Available online at http://www.scape-project.eu/report/scape-technical-report-evaluation-of-identification-tools, accessed 16th July 2012.

[15] Somaya Langley. "Repository Object Preservation Metadata Baseline for the DLIR Project". Internal National Library of Australia document available on the Digital Preservation Wiki: http://ourweb.nla.gov.au/apps/wiki/x/DbtFAQ. Accessed 10th July 2012.

[16] MediaInfo web site: http://mediainfo.sourceforge.net/en. Accessed 28th May 2012.

[17] The National Archives PRONOM web site: http://www.nationalarchives.gov.uk/PRONOM/Default.aspx. Accessed 2nd May 2012.

[18] The National Library of New Zealand Metadata Extraction Tool web site: http://meta-extractor.sourceforge.net/. Accessed 28th May 2012.

[19] Robert Neumayer, Hannes Kulovits, Manfred Thaller, Eleonara Nicchiarelli, Michael Day, Hans Hoffman and Seamus Ross. "On the need for benchmark corpora in digital preservation." In Proceedings of the 2nd DELOS Conference on Digital Libraries, December 2007. Available online at http://www.ifs.tuwien.ac.at/~neumayer/pubs/NEU07_delos.pdf, accessed 16th July 2012.

[20] Open Planets Foundation FIDO web site: http://www.openplanetsfoundation.org/software/fido. Accessed 2nd May 2012.

[21] Optima SC file identifier web site: http://www.optimasc.com/products/fileid/. Accessed 2nd May 2012.

[22] Oracle Outside-In Technologies web site: http://www.oracle.com/us/technologies/embedded/025613.htm. Accessed 2nd May 2012.

[23] Marco Pontello's TrID web site: http://mark0.net/soft-trid-e.html. Accessed 2nd May 2012.

[24] SWIG web site: http://www.swig.org/. Accessed 2nd May 2012.

[25] Universal Extractor web site: http://legroom.net/software/uniextract. Accessed 29th May 2012.

[26] WinCDEmu web site: http://wincdemu.sysprogs.org/. Accessed 29th May 2012.

[27] Xpdf web site: http://www.foolabs.com/xpdf/. Accessed 28th May 2012.

APPENDIX A: File Formats Identified in the Test Data Set by File Investigator Engine

Identifier	Description	Number Identified
3	Text File	77847
4	Graphics Interchange Format	19405
5	MS Windows Bitmap	7616
6	Amiga Interchange File Format Image	10
8	AutoDesk Animator Flic	37
9	GEM VDI Paint Image	9
11	PC Paintbrush Bitmap	462
12	PKZip Archive	226
13	MacBinary (Mac Data + Resource Fork)	1
15	DOS Program	26
19	Targa Bitmap Image	26
22	DOS Batch File	94
26	AutoCAD Drawing	8
28	BASIC Script/Source Code	44
30	DOS Program (Tiny)	31
44	Gzip Unix Archive	7088
46	Arc Archive	1
53	MS Windows Help Contents Table	3
56	Zoo Compressed Archive	1
58	dBase III/III+/IV/FoxBase+/FoxPro Datab	3856
66	Dr. Halo Palette	9
73	AutoCAD Drawing Exchange (ASCII)	28
77	MS Windows Program (32 bit)	367
79	LZexe Self Extracting Archive	1
89	GEM Write Format	4
95	Compiler Library (COFF)	255
99	MS Windows Cabinet Archive	266
100	MS Windows OLE Type Library	2
104	Lotus 123 Ver. 2 / Symphony 1.1 Workshe	3
105	Lotus 123 Ver. 3 & 4 Worksheet	1
111	MS Excel Worksheet/Template (OLE)	19796
114	MS Windows Icon	94
115	MS Windows Help File	30
133	Encapsulated PostScript Preview	96
140	MS Windows Internet Shortcut	4
142	Senddisk File	10
146	Tag Image File Format (Motorola)	336
154	InstallShield Install Script	4
157	WordPerfect Document	230
160	WordPerfect Graphic Image	9
163	WordPerfect Support File	2
164	MS PowerPoint Slides (OLE)	26152

166	MS Windows Wave Sound (Intel)	2041
171	Encapsulated PostScript Document	1544
172	Macintosh QuickDraw/PICT Image	12
176	Silicon Graphics RGB Bitmap Image	21
179	X11 BitMap	60
180	CALS Raster Image	2
185	JPEG File Interchange Format Image	76671
186	Portable BitMap Image (Binary)	13
187	Portable GreyMap Image (Binary)	16
188	Portable PixMap Image (Binary)	4
193	HP Printer Control Language File	1
200	Harvard Graphics Chart	3
206	Sun Raster Image	1
208	ACT! 2.0 Report	1
214	System Driver	3
215	MS Write Document	7
216	MS Word for Macintosh Document	100
217	MS Word for DOS/Macintosh Document	26
218	X Windows System Dump Image	1
222	MS Windows 3.x Screen Grabber	9
225	MS Windows Compiled Resources	616
229	MS Word 97-2003 Document (OLE)	39060
230	MS Windows Metafile (placeable)	88
234	MS Audio/Visual Interleave (Intel)	287
236	QuickTime Movie	65
241	Corel Draw Raster (Intel)	7
246	MPEG Animation	11
250	Toolbook Database	102
252	MS Windows 3.x Logo	3
258	Adobe Portable Document Format	137260
259	BinHex Archive	7
260	IBM OS/2 True Type Font	30
262	StuffIt Mac Archive	16
265	NeXT/Sun/UNIX Sound	24
269	MS Rich Text Format Document	1468
273	Portable BitMap Image (ASCII)	9
274	MS Compound Document (OLE) (General)	289
275	Portable GreyMap Image (ASCII)	22
281	Portable PixMap Image (ASCII)	7
283	PKZip Self Extracting Archive	2
287	IBM OS/2 Bitmap	18
288	MS Windows Program (16 bit)	74
290	MS Windows Driver (16 bit)	11
291	MS Windows Library (16 bit)	130
292	MS Windows Driver (16 bit)	3
297	MS Windows 3.x System Font	82

301	Adobe PostScript Document	9523
302	MS Windows Cursor	5
313	MS Windows Registry Import File	7
314	MS Windows Shortcut/Link	27
315	HyperText Markup Language	131160
316	Empty File	8640
320	MS Windows Color Palette (Intel)	6
321	Tag Image File Format (Intel)	1743
325	MS Windows Library (32 bit)	894
330	MS Access Database/Template/Addition	77
334	InstallShield 3.x Archive	21
336	MS Outlook Personal Information Store (	1
337	ICC Profile	2
338	MS Windows Compiled HTML Help Module	10
340	Adobe PostScript Font (Binary)	29
342	Extensible Markup Language	8965
343	Java Class (Compiled)	15
347	InstallShield 5 Cabinet Archive	23
349	MPEG Music File (+ID3v1 Tags)	182
351	Computer Graphics Metafile (Binary)	2
354	MS Language Character Set	33
357	MS Windows True Type Font	58
358	Adobe Printer Font Metrics	31
359	InstallShield Archive	11
368	MS Visual C++ DLL Exports File	239
369	MS Linker Database	1
372	MS Windows Help Full Text Search Cache	2
400	Adobe PhotoShop Image	62
401	Java Script Source Code File	289
405	Source Code Make File	100
406	C/C++ Source Code File	733
408	Printer Job Language Image	12
410	Adobe PostScript Document (PJL)	253
411	MS Developer Studio Project (ASCII)	28
412	Virtual Reality World (ASCII)	7
413	Virtual Reality World (Binary)	21
415	Cascading Style Sheet	179
416	MS Visual C++ Resource Script	3
418	Java Source Code File	185
423	MS Visual C++ Program Database	57
424	MS Developer Studio Workspace	10
429	Shockwave Flash Object	2808
433	Adobe Language Database	18
441	Adobe Illustrator Drawing	163
444	ANSI Text File	2
445	Python Tkinter Library Icons	24

447	Active Server Page	124
451	Comma Separated Values Text File	7892
455	Setup Information	208
456	Initialization File	297
458	WordStar Document	4
459	Printer Separator Page	7
468	Adobe Linguistics File	1
488	HTML + XML Namespace	3839
510	MS Visual Basic Class Module	1
532	Delphi Compiled Package Code	2
538	Evolution Email Message	2
546	Mutt Email Message	1
553	MS FoxPro Program File	68
556	Borland Paradox Primary Index	35
568	Borland Paradox Index	70
569	Computer Graphics Metafile (ASCII)	1
570	Extensible Style Language	32
574	Microsoft Outlook 2000 IMO Email Messag	6
575	MS Outlook Express Email Message	8
577	Pine Email Message	3
584	WinZip Self Extracting Archive	17
587	3D Studio Max Model Export (ASCII)	6
591	MS Office Macro Reference (OLE)	1
618	Common Gateway Interface Script	38
625	Adobe Font List	2
635	Corel PhotoPaint Image	30
636	Code Page Translation File	1
654	Stereo CAD-3D Objects Graphics Image	1
656	MS Visual Studio Properties	1
680	MS Visual Studio.NET Src Safe Code Cnt	7
686	SGML Document Type Definition	68
692	AutoDesk Web Graphics Image	47
703	Internet Message	1051
718	MS PowerPoint Slides (XML)	95
725	Fractal Image File	1
726	Flexible Image Transport System Bitmap	178
727	Flash Movie	4
730	FrameMaker Document	11
755	GenePix Array List	129
770	Gridded Binary Image	1
775	Hierarchical Data Format File (v4)	7
803	ISO 9660 CD-ROM Image (Data Mode 1)	68
807	Open Inventor 3d Scene (ASCII)	41
808	Open Inventor 3d Scene (Binary)	1
810	Paint Shop Pro Image Browser Cache	4
829	MS Access Lock File	15

836	HP Printer Control Language Image (PJL)	2
862	Berkeley UNIX Mailbox Format	414
863	Eudora Mailbox	1
870	Monarch Graphic Image	1
873	Machine Independent File Format Image	3
906	Object Oriented Graphics Library: Quadr	1
933	Eudora Email Message	24
938	NASA Planetary Data Systems Image	45
942	Pretty Good Privacy Key/Signature/Data	1
951	Macintosh Desktop Database	3
953	MS Setup Package	7
955	Perl Application	2280
961	Polygon Model Format	51
965	Portable Network Graphics Bitmap	4818
966	MacPaint Bitmap	12
968	Persistence of Vision Ray-Tracer	3
976	Lotus Freelance Graphics 97 File	1
987	Python Tkinter / UNIX Shell Script	259
1007	XML Resource Description Framework	17
1019	Red Hat Package Manager Archive	1
1033	MS Visual Source Safe Code Control	1
1051	Semicolon Divided Values File	108
1060	Standard Generalized Markup Lang	627
1063	ArcView GIS Shape	5453
1066	Pretty Good Privacy (PGP) Private Keyri	3
1077	MS Visual Studio.NET DB Discovery	2
1079	Structured Query Language Query	97
1080	Structured Query Language Report / Prog	4
1089	Thumbs Plus Database	281
1109	UU-Encoded File	3
1112	MS Visual Basic Project	3
1119	Khoros Visualization Image	9
1136	MS Write / Word Backup	29
1158	Advanced Visualizer 3D Object (ASCII)	13
1166	X11 Pixmap Image	28
1181	MS Datamap Index	1
1188	Flash Video	46
1196	MapInfo Map	230
1197	InstallShield Definition File	11
1204	MS Windows NT System Driver	1
1205	MS Datamap Index	81
1211	MS Windows Installer Package / Wizard	12
1212	DVD MPEG2 Video Object File	183
1214	MS Visual BASIC Source Code	6
1219	DVD Video Manager Data	52
1225	MS Visual BASIC Script/Header	1

1233	MS Windows Security Catalog	1
1239	MPEG Music File (+ID3v2 Tags)	576
1242	Text File: Unicode/DoubleByte/UTF-16LE	16
1245	Macintosh Disk Image	1
1248	MS Excel Spreadsheet (XML)	165
1249	MS Word Document (XML)	360
1254	Text File (UTF-8)	1176
1256	Source Code (General)	1327
1257	InterActual Installer Disk Identifier	3
1258	Tab Separated Values Text File	1829
1259	MS Windows Media Active Stream	743
1260	Pro/ENGINEER Geographic Image	1
1262	Internet Message (MIME)	81
1264	Scalable Vector Graphics Image	23
1266	QuickTime Xtra Plug-in	1
1293	InstallShield Index	2
1331	Adobe Portable Document (MacBinary)	108
1348	InstallShield Data	5
1352	Adobe Acrobat Resource CMap	2
1356	MS Publisher Document	1
1360	WordPerfect Document Template	5
1367	Generic Sound Sample	75
1374	MS Windows Application Usage Log	4
1377	NIST NSRL Hash Database	9
1390	PhotoStudio Image	1
1426	Bzip Archive V2	1
1432	Java Archive	33
1438	UFA Compressed Archive	11
1452	Pretty Good Privacy Public Keyring	3
1454	PestPatrol Scan Strings	1
1459	GEM Raster Image	2
1467	MS Visual Studio Solution	10
1534	AAC MPEG-4 Audio	16
1537	OGG Vorbis Compressed Audio	2
1660	JPEG-2000 Code Stream Bitmap	22
1673	Enhanced Compressed Wavelet	536
1693	Audio IFF Compressed Sound	1
1731	Macintosh Program (MacBinary)	1
1734	Text File (MacBinary)	1
1758	MS PowerPoint Slides (MacBinary)	7
1761	MS Word for Mac Document (MacBinary)	1
1766	MS Excel Spreadsheet (MacBinary)	3
1803	Assembly Source Code File	159
1808	MS C# Source Code	31
1811	MS Outlook Rich Text Formatted Message	2
1889	Modern ListGeo Output Image	106

1898	MathCaD Document	4
1910	MIME HTML Web Page Archive	5
1923	Unidata NetCDF Graphic Image	13
1943	Adobe Acrobat Installer Support File	2
1974	MySQL Database Index	64
2033	Photono-Software Stealther Skin	1
2051	MapInfo Spatial Table	237
2072	Virtual Calendar File	4
2102	DVM Movie	1
2132	XML Schema	17
2148	XML Paper Specification Document (Open	1
2169	VTeX Multiple Master Font Metrics	1
2177	MS Windows Visual Stylesheet (XML)	10
2179	OziExplorer Map (ASCII)	135
2186	Web Service Description Language	2
2189	MS Windows .NET Application Configurati	16
2208	MS Word 2007 Document (Open XML)	76
2209	MS Excel Spreadsheet (Open XML)	21
2210	MS PowerPoint Presentation (Open XML)	159
2264	UNIX Program/Program Library (32-bit)	1
2288	4D Creative Model (ASCII)	5
2294	XGL 3D Model	3
2312	Shrinkit/Nulib/NuFile Archive	101
2330	Extensible Markup Language (UTF-16LE)	5
2331	Extensible Markup Language (UTF-8)	455
2345	Text File (UTF-16BE)	1
2359	ArcExplorer Project	179
2360	Grace Project File	4
2385	Vicar Picture	3
2387	MySQL Generic Database Dictionary	1
2403	Personal Home Page Script	617
2407	Debian Linux Package	1
2438	AppleSingle MIME Format	5
2439	AppleDouble MIME Format	1697
2469	Google Earth Keyhole Markup Language	164
2484	Medical Waveform Description	1
2525	3D Studio 3.0 Image	9
2546	Alias/Wavefront Material Library	1
2551	Object Oriented Graphics Library: Objec	5
2557	Object Oriented Graphics Library: 4x4 T	706
2563	ACIS 3D Model	4
2596	Facility for Interactive Generation Fil	6
2602	MS Windows Media Player Play List	1
2603	Perfect Office Document	1
2604	The Bat! Email Message	1
2606	Yahoo! Mail Email Message	1

2612	OpenOffice Impress Presentation / Templ	1
2666	MS Compress 6.22 Archive	49
2667	MS Compress 5.0 Archive	97
2675	Floppy Disk Image / MBR (FAT16)	7
2678	Hard Disk Image/MBR (FAT32)	1
2682	MS Windows .NET Program (32 bit)	23
2686	MS Windows .NET Library (32 bit)	7
2689	DB/TextWorks Database Access Control Fi	1
2729	Extended Binary Coded Decimal Interchan	105
2738	MS Word 6.0/95 Document (pre-OLE)	206
2814	LDAP Data Interchange Format	1
2823	CER Internet Security Certificate	1
2826	Binary Property List	13
2830	HyperText Markup Language (UTF-16LE)	9
2831	HyperText Markup Language (UTF-8)	24
2852	Windows Policy Template	1
2909	DB/TextWorks Database Term and Word Ind	1
2920	MrSID Image	8049
2929	Big Tag Image File Format	258
2950	NetCDF CDL Metadata	20
2975	Andrew Toolkit CMU File	1
3002	Borland Paradox Database	35
3003	PFS: First Choice Document / PFS:Write	7
3006	MS Works Database 3 for Windows	4
3043	Zebra Metafile	22
3084	Pretty Good Privacy Signed Message (ASC	2
3085	Pretty Good Privacy Public Key Block (A	1
3086	Pretty Good Privacy Message (ASCII)	12069
3139	Linux Journalled Flash File System Imag	4
3158	UPX Compressed Executable	3
3168	InstallShield Self Extracting Archive	1
3172	3D Systems Stereolithography CAD Image	8
3173	3D Systems Stereolithography CAD Image	2
3176	PKZip Archive (Encrypted)	3
3195	Extensible 3D Model	1
3199	Apple Emulator 2IMG Disk Image (General	7
3204	ASIMOV2 Apple Emulator 2IMG Disk Image	2
3225	CGNS Advanced Data Format Database	16
3240	ACE/gr Parameter Data (ASCII)	1
3278	Palm OS Application	1
3297	Mobipocket eBook	19
3299	MS Rich Text Format Document (Mac)	31
3315	Wyko Vision Dataset (ASCII)	14
3316	Google Earth Keyhole Markup Langage (Co	157
3317	MS FrontPage Document (XML)	63
3318	Netscape Browser Bookmarks	5

3319	Web Script Source Code	12
3346	Tgif Drawing	6
3350	RISC OS Executable	2
3385	Impulse Tracker Sample	1
3405	Apple Serialized Typed Stream Data (Mot	1
3408	Interface Builder User Interface Resour	13
3410	Apple Property List	36
3435	DB/TextWorks Database	1
3436	DB/TextWorks Database Directory	1
3437	DB/TextWorks Database Textbase Structur	1
3460	Mac OS X Folder Information	40
3466	ArcInfo Coverage Export	328
3488	Earth Resource Mapping Satellite Image	353
3525	Windows CD-ROM Autorun	113
3526	Google Earth Compressed Keyhole Markup	63
3545	Commodore Compressed Archive	1
3553	ArcInfo Binary Image	13
3560	Seamless Image Graphic	278
3563	Netscape Email Message	9
3650	ArcMap GIS Project	2
3667	SnagIt Capture Image	1
3677	SQLite Database	1
3687	Mac OS X Package Bill of Materials	1
3688	Ruby Script	1
3702	DVD Video Title Set	152
3712	Text File With Formatting Codes	457
3713	Flight Recorder Data	504
3714	Radiance 3D Scene Octree	6
3715	Mac Draw Image	1
3716	ArcView DOQ Image	1
3717	NOAA-PMEL BBIS Data	38
3718	PrimeOCR Output	103
3719	Linux Patch	26
3720	NASA Imaging Radar Data	7
3721	Adobe Acrobat Distiller Log	1
3722	SigmaPlot Exchange File	5
3723	Erdas Image (HFA)	1261
3724	MetaMap Technology Transfer	1
3725	National Weather Service Forecast	2
3726	Princeton Transport Run Log	13
3727	Geological Survey Metadata	453
3737	Cumulate Draw Image	2
3757	Sybase Compressed Data	7
3762	GenBank Sequence Record	116
3787	HDF5 Archive	1
3854	DB/TextWorks Database Index	1

3962	Mac OS X Program (PowerPC)	33
3965	WordPerfect Document (OLE)	2
3967	LIST Interchange File Format	8
3968	PROP Interchange File Format	2
3970	Macromedia Director File (Intel)	2
3978	MagicDraw UML Project	4
4014	MS PowerPoint 2007 Theme (Open XML)	2
4015	Random or Encrypted Data (Headerless)	179
4016	Compressed Data (Headerless)	43
4051	NMEA GPS Log	3
4063	Adobe Portable Document Format (PDF/A-1	65
4064	Adobe Portable Document Format (PDF/X)	64
4067	DB/TextWorks Database Terms and Words	1
4103	Wiped Data (zeroes)	11
4109	HyperText Markup Language 5	13