Backing up digital preservation practice with empirical research: the National Library of Australia's Digital Preservation Knowledge Base

Gareth Kay, Libor Coufal, Mark Pearson

Abstract

This article introduces the National Library of Australia's Digital Preservation Knowledge Base which helps the Library to manage digital objects from its collections over the long term. The Knowledge Base includes information on file formats, rendering software, operating systems, hardware and, most importantly, the relationships between them. Most of the work on the Knowledge Base over the last few years has been focused on the mapping of functional relationships between file formats, their versions and software applications. The information is gathered through unique empirical research and is initially being recorded in a multiple-worksheet Excel file in a semi-structured format, though development of a prototype graph database is underway.

Keywords

file formats, software applications, functional relationships, digital preservation, knowledge bases

Introduction

The National Library of Australia (henceforth the Library) has an ongoing project to develop a knowledge base detailing, among other things, relationships between software applications and file formats – information which is currently missing in existing technical registries.

The project involves detailed empirical research that looks into the capabilities of selected software applications with respect to selected file formats. The research is predominantly format-driven since the primary long-term goal is to be able to access content stored in digital files. File formats which have been identified as present in the Library's digital collections are the highest priority and primary focus of the research but information about other formats is also gathered on the way.

For each major abstract content type (images, textual documents, videos, spreadsheets, maps etc.), we investigate the capabilities of selected software applications with respect to their associated file formats. The chosen applications may be freely available or proprietary in nature.

Details such as release dates, versions, vendor support, licensing status and dependencies are recorded both for formats and applications. Due to business needs the data gathered from the research is initially being recorded in a multiple-worksheet Excel file in a semi-structured format. Development of a prototype graph database together with software modules capable of importing data from the Excel file is taking place in parallel with the empirical work.

While Excel is not a suitable platform for a production knowledge base, its use in the development phase does have some advantages. Firstly, it allows us to get up to speed quickly without the need to spend too much time and effort upfront on developing an underlying data model and a database application. Secondly, as our understanding of the problem domain improves through empirical contact with it, we can experiment with changes to our data model at very little cost. When we

come across aspects of the software/file-format relationship which we judge might be significant to future preservation decision making but which the current iteration of the model provides no structured way to record, we can adapt the model accordingly.

Two very useful by-products of the empirical work are: a growing corpus of files in various formats and format versions containing known content which we have created ourselves and which we can usefully employ in testing software package capabilities; and a growing collection of VMWare virtual machine images for various current and legacy operating system environments.

The knowledge base is system- and strategy- agnostic, meaning it should support our work regardless of which preservation system we use and without preference for one preservation strategy over another. The long-term strategic goal is to build machine-readable knowledge bases to aid us in determining our level of support for different file formats; analysing the NLA's digital collection materials for preservation risks; and planning and executing preservation actions, be it migration or emulation, on digital objects which comply with the documented preservation intents for those objects.

Context

The National Library of Australia, like many cultural institutions worldwide, has a mandate to collect and preserve Australia's heritage. The Library's role, as defined by the National Library Act 1960, is to ensure that documentary resources of national significance relating to Australia and the Australian people, as well as significant non-Australian library materials, are collected, preserved and made accessible either through the Library itself or through collaborative arrangements with other libraries and information providers.

The legal mandate is further reflected in the current Corporate Plan for the years 2016 - 2020. The Strategic Priority 1 - Build the nation's memory - outlines the Library's aspiration to "enable Australians to understand their diverse social, cultural and intellectual histories by collecting, describing and preserving Australian publications and unpublished collections—in print **and digital** forms—so that they can be enjoyed by current and future generations." (National Library of Australia, 2016)

In addition to traditional published and unpublished physical collections and materials, which amount to over 10 million items or 256 km of storage space, the Library has acquired almost five petabytes of digital material over the past four decades. The Library's analogue, digitised and born-digital content span several collections, from oral histories to web archives.

The importance of digital preservation for the long-term survival of its digital collections has been recognised very early by the Library and a small unit was established in mid 1990s, at a time when digital preservation was still unknown to most. Acknowledged as a world leader, the digital preservation unit has advanced the theories of digital preservation and explored the challenges that lie ahead in papers on *obsolescence* (Pearson and Webb, 2008), *level of support* (Pearson, 2012) and *preservation intent statements* (Webb, Pearson and Koerbin, 2013). The concept of preservation intent statements in particular guides preservation planning decisions about methods, quality assurance and accountability in relation to identified significant properties and forms the foundation of the Library's benchmark test corpus, which will be explored further in this article.

The Library's thinking on digital preservation has been further shaped by its involvement in a number of internal and external projects over the past few decades, among them the work on the Automatic Obsolescence Notification System II project for Australian Partnership for Sustainable Repositories (Pearson and Walker, 2007); a first DRAMBORA risk assessment of the Library digital preservation capabilities (Long, 2008); the development of Prometheus, a system for processing

content off physical carriers (Elford et al., 2008); the creation of Mediapedia[1] to help identify risks to carriers (Del Pozo, Elford and Pearson, 2009); and most recently, the National and State Libraries Australasia[2] (NSLA) Technical Registry (McKinney et al., 2014).

All this work highlighted gaps in existing sources of technical digital preservation information, such as the well-known PRONOM[3] registry, including the lack of any meaningful risk metrics, and, together with the recognised need to preserve and manage the wealth of knowledge accumulated by its digital preservation staff, informed the conception and development of the Library's Digital Preservation Knowledge Base.

NLA Digital Preservation Knowledge Base

The Library's Digital Preservation Knowledge Base is the first practical step to equipping the Digital Preservation unit with essential knowledge about the file formats present in the Library's collections and their relationships with software applications. It has been designed to be used by digital preservation practitioners as a tool to answer key questions regarding the various aspects of a digital object, and provide information that could assist in performing preservation actions to maintain access to the content of digital objects. For example, we have format version X: which software application(s) can support it and are there additional software dependencies or rendering issues involved? Or, we have software Y: which file formats does it support? This knowledge is crucial for enabling the digital preservation unit to better assist and advise collection managers on any issues that could jeopardize the "health" of, or affect access to, digital objects in their collections. It will also allow us to plan any future preservation actions by providing information about rendering

[1] http://mediapedia.nla.gov.au

[2] http://www.nsla.org.au/projects/digital-preservation

[3] http://www.nationalarchives.gov.uk/PRONOM/Default.aspx

environments and dependencies for emulation or potential migration pathways between formats.

The Knowledge Base is actually comprised of several smaller knowledge bases, each providing specific information on a particular domain: file formats, rendering software, operating systems, and hardware (see Fig. 1). However, it is only through the combined power of the individual knowledge bases that we can build the complete picture and start to use it to assess the *level of support* of a particular file format, monitor *file format obsolescence*, or explore potential *migration pathways* from Format A to Format B via Software X, just to name a few applications. Drawing relationships between the many aspects integral to a digital object thus unlocks information that could only be speculated about before.

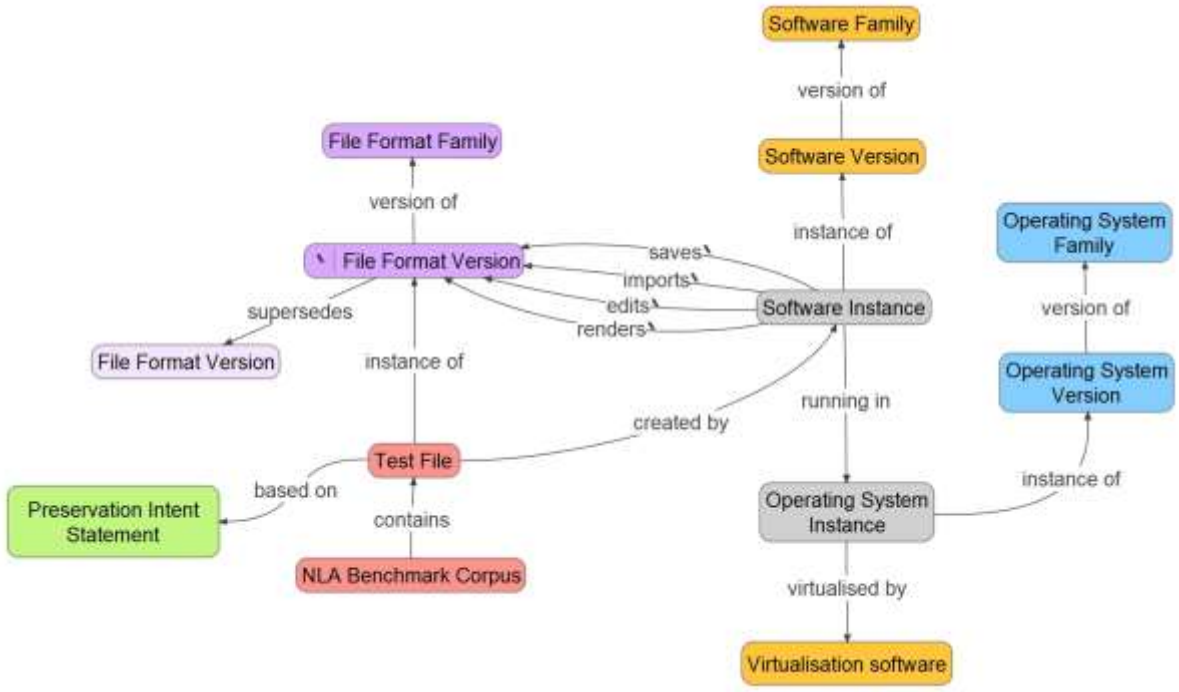

Figure 1: Directed graph showing some of the entities and relationships in the Knowledge Base data model

We are currently planning to start utilising the Knowledge Base to produce two regular, quarterly reports with the working titles *Obsolescence Watch* report and *Collection Health* report. The former report will synthesise the information on the level of support for individual file formats and their

versions from the Knowledge Base to establish the risk of file format obsolescence. The latter report will bring together the risk ratings from the *Obsolescence Watch* report and the file-format composition of our digital collections to answer the question "what do we know about the health of the individual collections?". For example, it may tell us that, in a particular collection, 99 % of the formats are rated as *safe* or *watch*, while 1 % is rated as *at risk*. This information will then be used to prioritise further digital preservation work.

Most of the work on the Knowledge Base over the last few years has been focused on the mapping of relationships between file formats, their versions and software applications. While there are other components of the Knowledge Base which are only partially populated and will need further development, it is the mapping of these relationships that is considered the most important aspect of the Knowledge Base, and by far the most immediately relevant and useful to the Library. It is also, as far as we are aware, a unique undertaking, producing comprehensive research not available elsewhere. We acknowledge and accept that, due to limited resources available, the other aspects of the Knowledge Base will have to be worked on gradually over time, or may be sourced from other existing resources.

Data sources

The Knowledge Base in the beginning was designed to be a place to store information about file formats, software and other components related to a digital object. This information was sourced from online resources, and relevant information was put into a structured form in the Knowledge Base. However, once we started to map relationships between file formats and software applications we realised that even the vast amount of information on the internet could not provide the exact information we were looking for.

For example, to know which file formats and their versions a particular software application supports requires a lot of "digging around" to get some useful information. The trustworthiness of

that information has to be questioned as well. Vendor documentation, if available, might contain the information we seek, but much of the time the level of detail rarely gets below the format family level: that is, format versions are not always specified. This may not seem like a large issue, but it is exactly the kind of information that is highly sought after by digital preservation practitioners.

To illustrate this, according to Microsoft's online documentation for Word 2013, the application supports the binary file format for Word 97 to Word 2003. It does not state anywhere that Word for Windows 95, or 6, or 2 or 1 Documents are supported, when in fact they are, once Word 2013 has been configured in a certain way. If a Word for Windows 1 Document profile is set up in the Knowledge Base, and no relationship is linked between it and the Word 2013 application, then the information would be misleading. If a user were to rule out using Word 2013 as a tool to access content stored in a Word for Windows 1 Document based on that documentation, they would not be aware that Word 2013 actually offers excellent support and faithful rendering of this older format.

This realisation lead us to change how information for the Knowledge Base is gathered. Rather than just relying on vendor documentation, we run a particular software application and check which file formats and their versions it supports. In doing so, we soon started to note discrepancies between vendor documentation and actual software functionality in terms of file format support. For example, it is not uncommon for software applications to list support for file formats or their versions that are not mentioned in vendor documentation, or vice versa. As another example, the vendor documentation may be much more detailed in terms of support for file format versions, whereas the software application might refer to formats generally, like Presentations (.ppt or .pptx).

Which source of information do we then trust and use in the Knowledge Base? Either way, inaccurate data could find its way into the Knowledge Base and potentially mislead users. It quickly became clear that there was only one solution: test and verify the relationships.

Mapping functional relationships

A key function of the Knowledge Base is to map out the capabilities of software applications in relation to the file formats they are (or claim to be) able to handle. To gather this data, we investigate certain functional relationships for each software/format combination. These relationships are used to describe capabilities exhibited by an application in relation to a format. Currently, we investigate four relationships: *import, render*, *edit*, and *save*. These relate to whether an application can parse a given format and build a "meaningful" internal representation of its content; render that internal representation; allow a user to make changes to the content; and save it to the format, respectively. Table 1 below provides more detailed definitions of these relationships.

Functional Relationship	Definition
Import	The software either has a built-in capability or requires a plug-in to input a format, interpret the data within said format, then create an internal representation of its content
Render	The software either has a built-in capability or requires a plug-in to re-present the *imported* content in a way which allows the user to meaningfully interact with the content, such as play/pause/fast-forward an audio/video file, or to view/zoom/rotate an image, etc.
Edit	The software either has a built-in capability or requires a plug-in that allows the user to meaningfully change the *rendered* content.
Save	The software either has a built-in capability or requires a plug-in to save either newly created or *imported* content into a particular format.

Table 1: Definitions of functional relationships

The process of documenting these functional relationships involves first harvesting information by running the software and recording the formats listed in the typical *Open/Import* and *Save as/Export*

menus, or any other means the application uses to get formats in and out. Such entries in the Knowledge Base are assigned a confidence value of *untested*, as we do not know whether the software can actually open or save the listed formats (or versions thereof). For file formats which are present in the Library's collections, hence considered high priority by the Library, the functional relationships are further empirically tested with the aid of the NLA Benchmark Corpus (described below). Such entries are assigned a confidence value of *tested* and are more detailed in nature.

To prioritise the mapping work, i.e. to answer the questions of which file formats we need to map and in which order, we used a file format analysis conducted in 2014 over a small collection (just over 2 TB, 1.3 million files) of born-digital objects that included about 360 distinct versions of different file formats, using DROID[4], which gave us a rough idea of what could be in the Library's digital collection.

The results from the DROID analysis were broken down into general content types: images, documents, spreadsheets, presentation, audio, video, etc. Each content type was then sorted by frequency. It was at this point we faced a tough question: do we start mapping the less frequent, not so common file formats first, or start with the most popular file formats? Both approaches have their merits, but ultimately it was agreed that the low-hanging fruit would be mapped first. The justification for that was that we would be able to map most of the file formats that exist within the Library's digital collection with little hassle, given that there would be a lot of information and software applications available to make the work easy. This way, we would quickly cover the formats which, together, represent the majority of our collections; then once those formats have been mapped, we can focus on the "long tail" (see Table 2). Of course, the counter argument is that the file formats present in the "long tail" might be the ones that are more likely at risk of obsolescence. However, the arguments of economies of scale and the most efficient use of limited resources

[4] http://www.nationalarchives.gov.uk/information-management/manage-information/policy-process/digital-continuity/file-profiling-tool-droid/

prevailed.

Frequency (no. of files)	No. of Formats	Total % of Files	Total % of Formats
100k >	4	59%	1%
10k – 99k	11	21%	3%
1k – 9k	20	4%	5%
100 – 999	46	1%	13%
50 - 99	15	0.08%	4%
10 – 49	70	0.13%	19%
1 – 9	198	0.05%	54%

Table 2: Frequency of file formats in the analysed sample

Before the mapping process, we conduct non-technical research on each file format. This entails identifying all file format versions of a particular file format, a timeline of when each version was released, and software applications that can support them in some way. Once done, we search online forums to record issues particular to the file format family, or specific to a format version. These might include rendering issues users encounter in a particular software application, or features/functions that present problems, etc. This information not only acts as a flag for us, but also helps us to design test files with those troublesome characteristics, features or functions present.

The selection of software applications to map file formats and their versions that are likely to exist within the Library's digital collection is not to be taken lightly. There are good applications that offer excellent support for file formats, as well as not-so-good applications that struggle to render file formats they purportedly support. Some software applications might also have particular strengths and weaknesses. For example, an application might accurately and faithfully render a file format, though might not be suitable for migration.

Software selection takes into consideration what is in the Library's Windows-based *Standard Operating Environment* (SOE), both for staff computers and computers available to the public in the reading rooms, which typically includes 'popular' applications for accessing more frequently-used file formats. Any software in the Library's current SOE that supports any priority format is automatically

selected for mapping, regardless of other factors which might normally exclude software from selection, such as adoption or overall rendering quality, as it is already being used by Library users to access our digital content. The mapping process aims to identify all file formats and their versions supported by selected applications, and then assess the overall quality of rendering and level of support for the priority file formats among them. This in turn provides feedback to the Library's collection areas and IT about the suitability of such software for future SOE rollouts.

Other suitable candidates for mapping are selected from within the Digital Preservation unit's extensive Software and Hardware Library[5] of over 1300 software titles, or from other sources including online. The selection currently focuses on Windows-based applications as this is the environment in which the Library operates, although this may be reconsidered in the future. Software applications are assessed by their perceived support for priority file format under investigation, and this is usually done by reviewing vendor documentation or trialling the software, if that option is available, with the main focus on the rendering of file formats.

Where legacy versions of a software application exist, there is a temptation to map them all. For example, Adobe Photoshop has 18 major versions spanning nearly thirty years, and each version has a Windows and Macintosh release. If we have access to these, do we map them all? Although mapping all versions for all platforms of a particular software application would provide great insight into the rise and fall of file formats, we simply do not have the time or resources to do the work. More importantly, it does not need to be done as our main use case is to establish what support current applications have for various file formats.

We therefore decided to map the most recent software applications first and, if they do not support a particular version of a priority file format or if there are clear rendering issues, only then would we

[5] The Software and Hardware Library has expanded in size over the years mostly from generous internal and external personal donations, IT hand-me-downs from legacy Library SOE, as well as a small number of purchased acquisitions.

consider mapping an earlier version of the software. For example, for Microsoft's PowerPoint Presentation format, of which the Library has PowerPoint for Windows versions 3, 4, 95, 97, 2000, 2002, 2003, 2007+, we initially mapped Microsoft PowerPoint for Windows 2016. The mapping showed that this application does not support PowerPoint for Windows 2, 3, 4 and 95. We then worked backwards through many previous versions of PowerPoint until we found the one(s) which can support the remaining format versions, at which point we stopped.

The selected software applications are installed in their own pristine virtual environment running in a contemporary operating system in VMware Workstation. The decision to run applications in virtual environments rather than on standard computers is due to a few reasons. We observed very early on that when two versions of Microsoft Office were run on the same computer, Microsoft Word of the earlier version, Office 95, suddenly had the ability to import and render Word 97-2003 Documents. Support for these later documents is clearly unintentional and could be ignored in inputting that relationship into the Knowledge Base, but what other unintentional "enhancements" are not as obvious? Setting up virtual machines avoids the risk of unintentionally corrupting the Knowledge Base with such "enhancements".

Each virtual machine runs an operating system. From the many Windows and Mac operating systems in our Software and Hardware Library, we select the operating system the application would have been using when released onto the market. Installing an application in its contemporary operating system reduces the risk of introducing any unintentional effects a modern operating system might have on an older application. Using virtual machines makes this task easier and less time-consuming. By using virtual machines, we also remove the issues of sourcing, running and maintaining legacy computer hardware. For example, if we needed to run an application in Windows 3.11 without using virtual machines, we would have to source working computer hardware to do that and then maintain that hardware environment for as long as we want the application we are mapping running, which includes dealing with hardware failure. On top of that, we are able to save a

snapshot of every application we map so that we can revisit that application in the exact same environment should the need arise.

The goal of the mapping process is simple: run and explore a software application and write down all the file formats, and formats versions where possible, that the application supports, not just the priority file format(s) we are interested in mapping, and at the same time capture the degree of support the application has for each file format by linking it to the functional relationships. We aim to map all versions of priority file formats to a given application, even if the application does not list those versions. For non-priority file formats, i.e. file formats that currently have not been identified to exist within the Library's collections, only those file format versions listed by applications will be mapped, but we will not expand the mapping to include all existing versions.

Most of the information we are looking for is in the *Open* or *Import* dialogue box, which indicates the functional relationships *Import* and *Render,* and *Save* or *Save as* dialogue box, which indicates the functional relationship *Save*.[6] Not all applications are that straightforward, however, as there is more than one way of getting file formats in and out of applications, and this warrants a thorough exploration of the application at hand. Let us take a closer look at what this process does. Below is a screenshot of the *Open* dialogue box within the Microsoft Word 2010 application.

[6] *Edit* is assigned to a file format (version) when an application can *Import*, *Render* and *Save* it by default, based on the assumption that editing is possible. Further testing of prioritised file formats will confirm whether editing is actually possible.

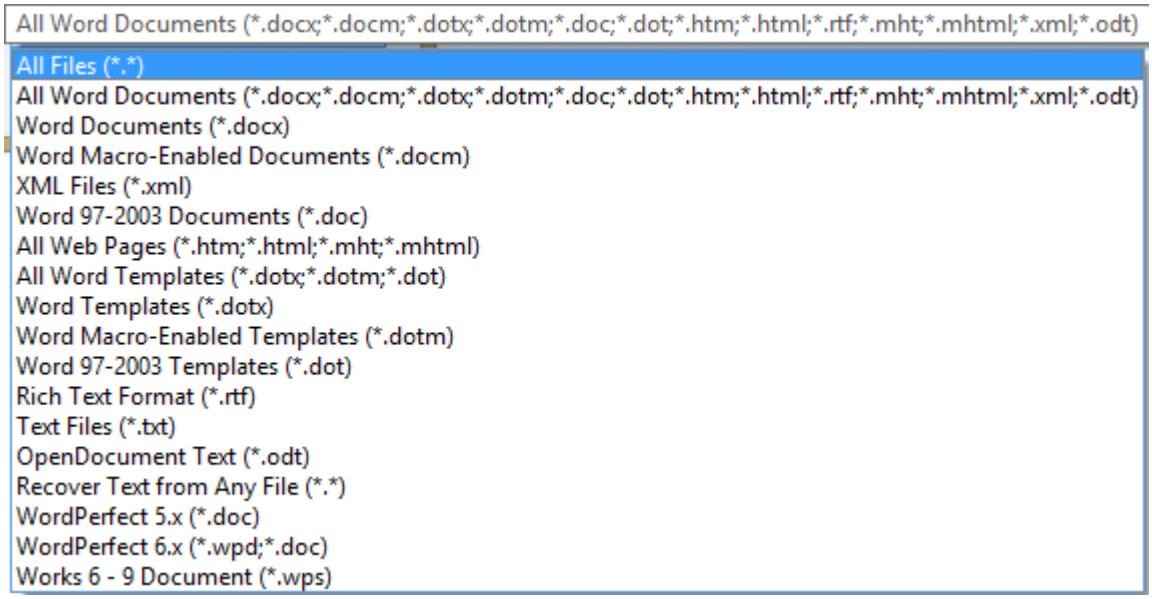

Figure 2: 'Open' dialog box within Microsoft Word 2010

Microsoft Word 2010 has provided the user with a list of file formats, their extensions and some format version information. We take file extension, format and version information and put it into the relevant columns in the mapping spreadsheet. Each of these entries is assigned the functional relationships *Import* and *Render* (See Fig. 3).

Extension	Format name and version	Standardised format name	Standardised format version	Format Content Type	Level of Priority	Software instance	Imports	Renders	Edits	Saves	Confidence Value
doc	Word Document	Word for Windows document	1	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes[1]	No	No	Tested
doc	Word Document	Word for Windows document	2.0c	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Tested
doc	Word Document	Word for Windows document	2000	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Tested
doc	Word Document	Word for Windows document	2003	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Tested
doc	Word Document	Word for Windows document	95	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Tested
doc	Word Document	Word for Windows document	97	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Tested
doc	Word Document	Word for Windows document	2002	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Tested
doc	Word Document	Word for DOS document	5.5	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Tested
doc	Word 97-2003 document	Word for Windows document	97-2003	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
doc	WordPerfect 6.x			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Untested
doc	WordPerfect 5.x			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Untested
doc	Word Document	Word for Windows document	6	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Untested
doc	Word Document	Word for Mac Document	4	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Untested
doc	Word Document	Word for Mac Document	5	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Untested
doc	Word Document	Word for Mac Document	5.1	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Untested
doc	Word Document	Word for Mac Document	6.0/95	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Untested
doc	Word Document	Word for Mac Document	98	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Untested
docm	Word Macro-Enabled	Word for Windows Macro-	2007	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
docx	Word Document	Word for Windows document	2007	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Tested
docx	Word for Windows		2010	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Tested
dot	Word Template 97-2003	Word for Windows template	97-2003	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
dotm	Word Macro-Enabled	Word for Windows Macro-	2010	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
dotx	Word Template	Word for Windows template	2010	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
htm	web page filtered			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
htm	Web page			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
htm	HTML			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
html	web page filtered			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
html	Web page			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
html	HTML			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
mht	Single File Web Page			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
mht	HTML			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
mhtml	Single File Web Page			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
mhtml	HTML			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
odt	OpenDocument text format	OpenDocument text format	1.2	Document	1	Word 2010 (Windows 8 Pro)	Yes[1]	Yes	No	No	Tested
odt	OpenDocument text format	OpenDocument text format	1.1	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Tested
odt	OpenDocument text format	OpenDocument text format	1	Document	1	Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Tested
rtf	Rich Text Format			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
txt	Text file			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
wpd	WordPerfect 6.x			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	No	No	Untested
wps	Works document 6.0-9.0			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
xml	Word XML document 2003	Word for Windows XML	2003	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
xml	Word XML document	Word for Windows XML	2010	Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested
xml	XML file			Document		Word 2010 (Windows 8 Pro)	Yes	Yes	Yes	Yes	Untested

Figure 3: Screenshot of Mapping Spreadsheet

The column *Format name and version* is populated by format information given to us by the application. However, just recording what the application gives us is not enough; in fact, most of the time, applications understate which formats they support because they do not list all versions of a particular file format. For example, Word 2010 states that it opens Word Documents, but what if it does not open Word for Windows 1 Documents or Word for Mac 4 Documents? Or, what if the opposite were true, and Word 10 does open Word for Windows 1 Documents? Any digital preservation practitioner would surely benefit from that information. That is why, for priority formats, we attempt to map all existing versions of a format, if known, against the given application.

Once a file format (version) has been mapped to a software application in the spreadsheet, it is assigned a confidence value of either *untested* or *tested*. The idea behind the confidence value is to inform users whether a particular functional relationship between the file format (version) and

software application has been verified by the person inputting that information. By default, all new records are assigned the *untested* value to make it clear that no one has done any verification. At the moment, we are only interested in testing file formats that exist within our collections, and as such, most of the records in the Knowledge Base will be *untested* as they are non-priority file formats that have been mined from software applications during the mapping process.

The two columns *Standardised format name* and *Standardised format version* are a new feature recently introduced into the Knowledge Base to begin to address the issue of file formats being referred to by different names, which happens frequently. File format names might include a vendor's name (e.g. PDF/Adobe PDF), or be an acronym, as opposed to a fully-spelled version of a name (e.g. VWPG/Vector WordPerfect Graphics), or use multiple, either similar or completely unrelated, name variants or aliases (e.g WebP/Weppy file format), just to name a few. At the extreme end, we have the case of, for example, Paintbrush Bitmap Image which can be referred to by any of the following names: PCX, ZSoft IBM PC Paintbrush, Zsoft Publisher's Paintbrush, PC Paintbrush format from ZSoft Corporation, Zsoft PCX image; but which is distinct from ZSoft IBM PC multi-page Paintbrush/Mulitpage PCX format. Having multiple variants of file-format names would make querying the Knowledge Base very hard, if not impossible, and so we have begun to standardise priority file formats, taking into consideration how they are named in official vendor documentation or in repositories like PRONOM.

The mapping process is repeated until all file formats, and any versions present, are recorded in the spreadsheet, and all supported functional relationships are noted.

Non-format related data are also captured during the exploration process and populated in the *Instance* profiles for the software version, operating system version, and virtualization software version being used. These instances record detailed versioning of software, build numbers, and licensing.

NLA Benchmark Corpus

The testing process has evolved over the years from a simple process that was merely interested in confirming a software's ability to open a file format and nothing more, to one that is specifically designed to not only test a software's support for a given file format, but also identify potential rendering issues that collection and digital preservation managers should be aware of.

The initial testing process involved sourcing tests files from the internet. The only thing we were concerned about was support for the format itself, not for the rendering of its content. Such a simple approach, although convenient, has limitations. Hypothetically, say we need a Microsoft Word for Windows 1 Document to test that Microsoft Word for Windows 2016 can open and render its content, and we come across some site which says that it may have a Microsoft Word for Windows 1 Document. How do we know that that file really is the format and version it is purported to be? Yes, we can run file identification and characterisation tools over the file, which may or may not confirm it, but if the results prove to be inconclusive, which happens often, then doubt will creep its way into the Knowledge Base.

We needed to come up with a solution to minimise doubt and increase confidence in the test files we use and decided to make use, once again, of our extensive software library and virtual machines to create our own test files. By doing this, we are able to not just remove a lot of doubt surrounding the file format the test file is in - not all, of course, as applications might not necessarily specify the file format version it exports - but also provide information on exactly how the test file was created, which software application it was created in, which operating system the software that created the test file was running in at the time, and the virtualization software used to run the operating system.

It also occurred to us that, given we can create our own test files, we could add carefully designed content with extensive formatting and features to the test files to make them as authentic as possible, as if they were files within the Library's collection. By adding this "rich" content, we could

begin to record any rendering or other digital preservation issues we observe when testing a file format and its relationships to software applications. This is how the NLA Benchmark Corpus was born.

The NLA Benchmark Corpus (henceforth the Corpus) is a development that no one envisaged, yet has become a key component of the mapping project and Knowledge Base. The Corpus offers confidence when testing functional relationships because each test file is linked to the file format (version) it is and the software application that was used to create the test file in the Knowledge Base. This information is provided in the Corpus manifest, and includes other information such as date created, checksum, and the results of file format identification.

Being able to offer the provenance of a test file, the technical environment in which it was created, and a much higher degree of confidence as to which file format and version it is makes this Corpus highly valued. But that is not all. Where the Corpus stands out is in what goes into the test files themselves. As discussed above, when we used to source test files from the internet, it was not uncommon to come across test files with nothing but a one sentence statement as means of content. Such content is useless if we want to explore the types of issues we might encounter when opening a given format in an application. The test files in the Corpus, however, are carefully crafted to include a large variety of content and formatting, with current preservation intent statements in mind, so that we can observe and get a good understanding of which features and characteristics are supported, or not, by whichever software application we choose to test.

The test files are designed in such a way that the content is self-describing, that is, a user is told what they should expect to see (Fig. 4). For example, "**this sentence should be in bold**" is a clear indication that that sentence should be in bold formatting, and if not, there is a rendering issue.

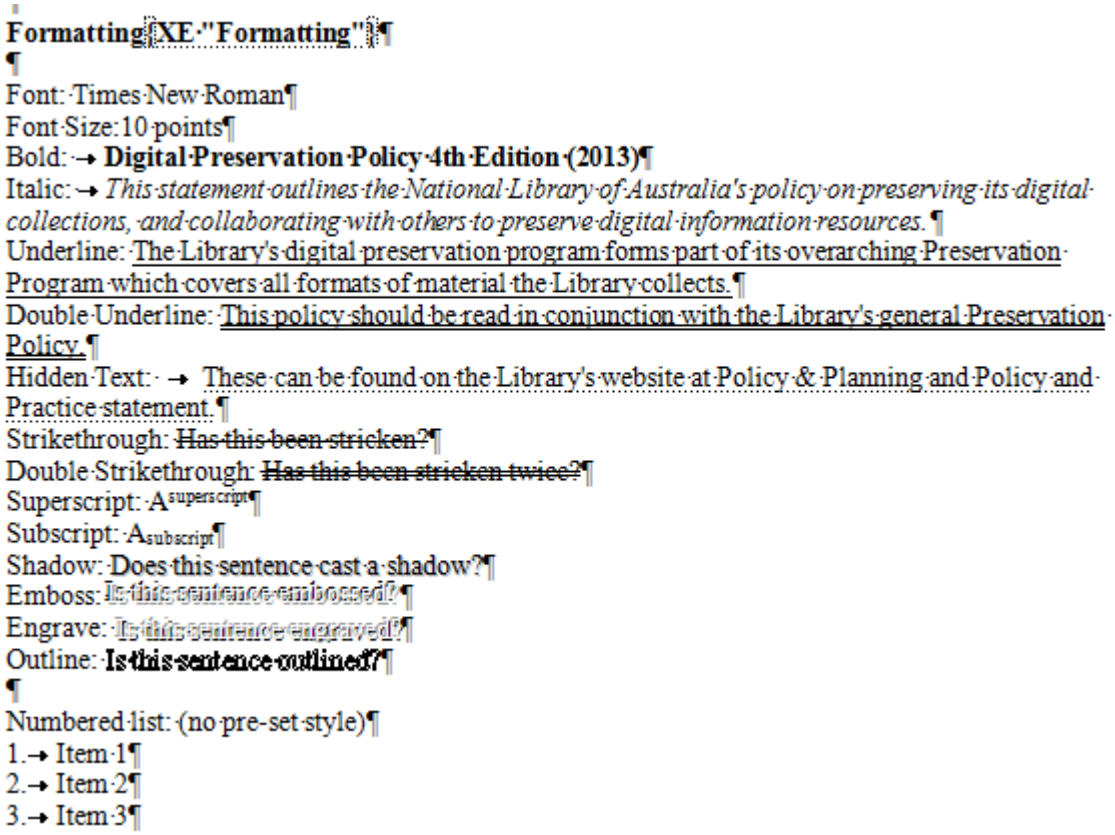

Figure 4: Screenshot of part of a Microsoft Word for Windows 97 test file

Screenshots of each test file as it is being rendered in the application that created it are taken as well and bundled with the test file. That way, we can provide a visual indication of how the test file should be rendered. We acknowledge that screenshots are useful, but they may not represent a one hundred percent faithful reproduction of the test file. Even how a monitor is configured might affect how a test file is rendered, and this is a limitation that has yet to be addressed.

During the process of investigating the functional relationships, issues with applications, which could have an effect on the support for file formats, sometimes arise. Examples could include rendering issues; discrepancies between documented and actual software functionality; software/hardware dependencies; installation issues; or the inability to preserve certain properties of content which may have been deemed significant by the preservation intent statements associated with the content type. These issues are currently recorded in free-text format in the *preservation notes* field but it is becoming clear that there is a need to record these issues in a more structured way which

would be amenable to (semi)automated querying.

Graph database

While Excel is an excellent tool for recording semi-structured empirical data with very low overheads for getting started, it is not a full database management system and as such does not provide a powerful querying language, tools for describing data model schemas or multi-user access. Therefore, at some point we have to replace the Excel spreadsheet with a full database system to facilitate data entry, querying, reporting and the ability to integrate the Knowledge Base into our processes.

As the Fig. 1 shows, our problem domain is all about relationships between entities such as software instances, format versions, operating systems etc. It makes sense to look for a database management system which supports modelling this kind of structure directly. This is exactly what graph databases do.

In graph databases, entities such as people or businesses are represented by *vertices* or *nodes* which roughly correspond to *records*, *relations*, or *rows* in a relational database. Vertices can have *properties* and can be linked to other vertices by lines which are called *edges*, *graphs* or *relationships.* Edges represent the relationship between vertices and are “the key concept in graph databases, representing an abstraction that is not directly implemented in other systems. Meaningful patterns emerge when examining the connections and interconnections of nodes, properties, and edges.” (Wikipedia contributors, 2017)

Graph databases are much more flexible than relational or document databases as “in a graph, each vertex is seen as an atomic entity (not simply a "row in a table") that can be linked to any other vertex or have properties added or removed at will. This empowers the data modeler to think in

terms of actors within a world of complex relations as opposed to, in relational databases, statically-typed tables joined in aggregate." (Apache TinkerPop, 2017) They are therefore ideally suited to problem domains where the schema has not been strictly defined or may be a "moving target".

Representing the Digital Preservation Knowledge Base as a graph database is conceptually quite straightforward. Vertices are used to model the *things* in the model domain, which naturally differentiate into broad classes such as:

- *Software family* - e.g. Adobe Photoshop
- *Software version* - e.g. Adobe Photoshop CS4
- *Software edition* - e.g. Adobe Photoshop Professional
- *Format family* - e.g. Portable Document Format (PDF)
- *Format version* - e.g. PDF 1.4
- *Operating system family* - e.g. Windows, Linux, Mac OS X
- *Operating system version* - e.g. Windows XP

With the *things* in our problem domain modelled as vertices in the graph database, the *functional relationships* described in the section *Mapping functional relationships* can be represented as edges between those vertices. Since edges may contain arbitrary properties just like vertices, we can store additional characteristics of the particular functional relationship such as "*is this function supported (yes|no)*"; "*has the function been tested?*"; "*are there notes associated with the relationship?*" as properties of these edges.

We started to work on implementing the Knowledge Base as a prototype graph database. From the many database management systems which support the property graph model, we chose OrientDB.[7] Currently, we have an automated import mechanism in place and the data have been migrated from

[7] http://orientdb.com/

the Excel worksheet into the graph database. As the next step, we are planning to test the querying functionality on a number of selected use cases. Provided the results are satisfactorily, we will then develop a graphical user interface for data entry, which would allow us to decommission the Excel spreadsheet, and implement the querying functionality.

Conclusion

The NLA Digital Preservation Knowledge Base is not just another research project; it is actually used to support everyday digital preservation practice at the Library. Thanks to the information from the Knowledge Base we can start informing the collection managers about the "health" of their collections and any preservation risks.

Our digital preservation system, Preservica[8], has been in production since October 2016 and we are currently ingesting digital publications from the legal deposit stream but more born-digital content, both published and unpublished, from other content streams will start arriving very soon. With that, the variety of file formats we will have to deal with will grow exponentially and so will the importance of the Knowledge Base.

To date, the mapping part of Knowledge Base contains just over 12 thousand entries for file-format/software pairs, 2,525 of which are for priority file formats, and 1,327 of which have been tested. Altogether, 32 priority file-format families and their numerous versions have been mapped to 130 software instances. However, we have only scratched the surface and more mapping will be required as more priority file formats are identified within our collections and as new software applications or versions become available for mapping.

Small, incremental changes and improvements to the processes have always been made as the

[8] http://preservica.com/

mapping went along. Once we started to look at the data, we identified data quality as one of the issues which lead to regular quality assurance processes being built into the system. In addition, a more formal review is now underway, looking more holistically at the current processes and the underlying data model. These include the need to review whether the current definitions of the functional relationships are adequate and whether there are additional functional relationships that should be mapped; the standardisation of file format names to remove ambiguity and improve consistency; the reworking of the preservation notes field, which is currently free-text, to make it much more structured; and the development of other domains of the Knowledge Base, which have been underdeveloped due to the focus on the mapping component.

While the outcomes of this project will provide practical benefits for the National Library of Australia, we would like to explore the possibility of releasing the Knowledge Base and Corpus for the benefits of the wider digital preservation community. This would allow us, in turn, to seek feedback to address current gaps, enhance the product and even find contributors who could add their own data to the Knowledge Base. The Knowledge Base will complement and extend the information from other technical registries and could eventually become part of a larger digital preservation ecosystem, such as the proposed NSLA Digital Preservation Technical Registry.

References

Apache TinkerPop (2017) Apache TinkerPop. Available at: http://tinkerpop.apache.org/ (accessed 18 April 2017).

Del Pozo N, Elford D and Pearson D (2009) Mediapedia: Managing the Identification of Media Carriers. In: *DigCCurr 2009 proceedings*, Chapel Hill, NC, USA, 1-3 April 2009, pp. 76-78. Available at: https://www.slideshare.net/natlibraryofaustralia/mediapedia (accessed 16 April 2017).

Elford D et al. (2008) Media Matters: developing processes for preserving digital objects on physical

carriers at the National Library of Australia. In: *74th IFLA General Conference and Council*, Québec, Canada, 10-14 August 2008. Available at: https://archive.ifla.org/IV/ifla74/papers/084-Webb-en.pdf (accessed 18 April 2017).

Long AS (2008) Building trust: pilot preservation audit of the National Library of Australia Digital Repository. Report, National Library of Australia.

McKinney P et al. (2014) Reimagining the Format Model: Introducing the Work of the NSLA Digital Preservation Technical Registry. *New Review of Information Networking* 19(2): 96-123. Available at: http://www.tandfonline.com/doi/full/10.1080/13614576.2014.972718 (accessed 18 April 2017).

National Library of Australia (2016). Corporate Plan 2016-2019. Available at: http://www.nla.gov.au/sites/default/files/corporate_plan_2016-2020.pdf (accessed: 16 April 2017).

Pearson D (2012) The Adventures of Digi: Ideas, Requirements and Reality. In: *Future Perfect 2012*, Museum of New Zealand Te Papa Tongarewa, Wellington, 26-27 March 2012. Available at: https://www.nla.gov.au/content/the-adventures-of-digi-ideas-requirements-and-reality (accessed 13 April 2017).

Pearson D and Walker M (2007) Report of the Format Notification and Obsolescence Service (AONS II). Report, Australian Partnership for Sustainable Repositories, November. Available at: http://apsr.anu.edu.au/aons2/report.pdf (accessed: 16 April 2017)

Pearson D and Webb C (2008) Defining File Format Obsolescence: A Risky Journey. *The International Journal of Digital Curation* 3(1): 89-106. Available at: http://www.ijdc.net/index.php/ijdc/article/view/76 (accessed: 18 April 2017).

Webb C, Pearson D and Koerbin P (2013) 'Oh, you wanted us to preserve that?!' Statements of Preservation Intent for the National Library of Australia's Digital Collections. *D-Lib Magazine*, 19(1/2). Available at: http://www.dlib.org/dlib/january13/webb/01webb.html (accessed 18 April 2017).

Wikipedia contributors (2017) Graph database. *Wikipedia, The Free Encyclopedia.* Available at: https://en.wikipedia.org/wiki/Graph_database (accessed 18 April 2017).