DIGITAL PRESERVATION CHALLENGE

Solutions Report

Alex Mason
alex.mason@durham.ac.uk

July 2008

Contents

1 Introduction

The Digital Preservation Challenge is a contest organised by Digital Preservation Europe (DPE). The challenge presents five realistic scenarios involving so called "digital objects", with the task to develop digital preservation solutions to provide access to the digital objects, and hence preserve them.

This report contains my attempts to complete each of the five scenarios. I've attempted to document the exact method I employed to complete the scenarios, along with any tools and resources I used along the way. The appendices catalogue the tools and resources, as well as provide my own conclusions on their suitability for the tasks they were employed.

I am an undergraduate student reading Computer Science at Durham University, having recently completed my first year. I have no prior knowledge of digital preservation, and so the entire subject matter of this contest is new to me. I've tried to capture how I went about each scenario, including any mistakes I made and red herrings I followed, simply to give a better picture on how a solution was arrived at.

The majority of the search was carried out on a Dell Inspiron 1721 notebook, running Windows Vista and Ubuntu Linu via dual boot. A few things required the use of an older version of Microsoft Office, and so a DELL Latitude D510 running Windows XP with Microsoft Office 2000 was used for those.

All custom programming was written using the Perl programming language[1] and tested against perl version v5.10.0. I feel perl is the perfect tool for custom preservation scripts as it is: cross platform, allows for rapid development time as a dynamic language, and provides a formidable amount of library code via CPAN[2] which provide a wide range of access to all sorts of formats.

2 Scenario walkthroughs

Each walkthrough consists of the scenario description and the list of tasks to be completed, followed by a detailed analysis of the methods employed to complete each task of the scenario.

[1]http://www.perl.org
[2]http://www.cpan.org

2.1 Legacy Application File

2.1.1 Scenario Description

Your company archivist discovered an old tape in a store-room. The content is not known but the label “Master Backup” suggests that is highly valuable. There were four types of files on it, one type of text documents, one type of graphics and two unknown file types. You are asked to identify the unknown file types and display the content of the given sample files in such a way that they may be used in a different application. You are also asked to design an appropriate preservation strategy that will facilitate access to such records, and that can be applied, as far as possible, in an automatic manner. Moreover, you are asked to estimate the cost/effort required to deploy the strategies you propose. Task

Your task is to:

1. Identify the type of content of the unknown files.
2. Propose one or more suitable preservation strategies and provide a thorough description that highlights their advantages and disadvantages.
3. Implement a preservation strategy capable of mass handling of files of this type, giving an estimate of the cost/effort for deploying the strategies.
4. Apply the preservation strategy to the objects and display the files.
5. Analyse the benefits and the shortcomings of the preservation strategy.

2.1.2 Scenario Solution

‘logo Pur.ltp’ (found in the ‘graphic’ folder) was the first file I attempted to identify.

I first attempted to identify it via its file extension using first FILExt[3] and then wotsit[4]. Unfortunately the extension wasn’t recognised in either database, nor did a Google such for the extension bring up any clues as to the file format.

My next port of call was the ImageMagick identify utility[5]:

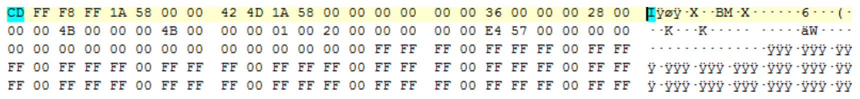
```
>identify "logo Pur.ltp"
identify: no decode delegate for this image format.
```

[3]See Appendix A.3
[4]See Appendix A.4
[5]http://www.imagemagick.org/www/identify.html

which was similarly unhelpful.

At this point, I moved on to the other files and came back to the logo at the end. From identifying the other files prior, I put an estimated time of backup of the logo file at around the early 90s in terms of the scenario. This being the case, I tried an extensive Google search for the file extension, coming up with a somewhat obscure file[6] identifying the extension as a disk image for a Zx Spectrum[7] emulator. I was highly dubious this would lead to anything, the file being only 63.2kb in size, and the Zx Spectrum being a slightly obscure choice to preserve an image. I was correct in my scepticism, and fell back on the assumption ltp stood for "long term preservation", as Google thew up numerous references to this acronym, which fit with the scenario.

I then attempted to identify the file from content, opening it in a hexadecimal editor[8] (see figure 1).



```
CD FF F8 FF 1A 58 00 00  42 4D 1A 58 00 00 00 00  00 00 36 00 00 00 28 00  Íÿøÿ·X··BM·X······6···(·
00 00 4B 00 00 00 4B 00  00 00 01 00 20 00 00 00  00 00 E4 57 00 00 00 00  ··K···K·····  ·····äW····
00 00 00 00 00 00 00 00  00 00 00 00 00 00 FF FF  FF 00 FF FF FF 00 FF FF  ··············ÿÿÿ·ÿÿÿ·ÿÿ
FF 00 FF FF FF 00 FF FF  FF 00 FF FF FF 00 FF FF  FF 00 FF FF FF 00 FF FF  ÿ·ÿÿÿ·ÿÿÿ·ÿÿÿ·ÿÿÿ·ÿÿÿ·ÿÿ
FF 00 FF FF FF 00 FF FF  FF 00 FF FF FF 00 FF FF  FF 00 FF FF FF 00 FF FF  ÿ·ÿÿÿ·ÿÿÿ·ÿÿÿ·ÿÿÿ·ÿÿÿ·ÿÿ
```

Figure 1: Hex edit view of the first few bytes of 'logo Pur.ltp'

The file contained obvious padding, and so I concluded it was a raw image type, uncompressed and representing a fixed size raster image. After unsuccessfully trying to identify what looked like a header using the first few hex bytes (CD FF F8 FF 1A 58), I moved to searching for the distinctive repetitive buffer pattern, "FF FF FF 00 FF FF FF". Such a pattern, I suspected, was idiosyncratic of the format, as I would have expected a blander pattern of "00 00 00 00" or a fixed repeated symbol.

Googling for this pattern revealed a C header file entitled,"Microsoft Windows bitmap (BMP) file definitions."[9] This file also provided a handy description of the file format, and header:

```
/*********************************************
* Uncompressed 16-color BMP
*
* 00000  42 4D        id          'BM'  \
* 00002  66 01 00 00  filesize    358   |
* 00006  00 00 00 00  reserved          |
* 0000A  76 00 00 00  headersize  118   |
```

Noticing the start of the header "42 4D" was located 8 bytes from the start of the 'logo

[6]http://melkor.dnp.fmph.uniba.sk/ garabik/readme.txt
[7]http://en.wikipedia.org/wiki/ZX_Spectrum
[8]http://www.editpadpro.com - more a text editor with hex editing capabilities
[9]http://david.tribble.com/src/bmp/bmp.h

Pur.ltp' file, I removed the first 8 bytes, which fixed the file. In summary, file was in Windows 16-color BMP format, with a corrupted header, which could be fixed by removing the first 8 bytes from the header.

In hindsight, I should have followed up the ltp = "long term preservation" lead more thoroughly, and made the obvious deduction that it was a file in one of the major image formats suitable for long term preservation, only slightly corrupted.

Methods of digital image preservation appear multiple times throughout the challenge, and are similarly pronounced in general digital preservation. As such, Appendix B focuses on image formats suitable for preservation, and their advantages/disadvantages. Suffice to say, I felt the Windows bitmap image itself was not a suitable preservation format, and so opted for migration[10] to an image format suitable for preservation.

Following from the conclusions drawn, I developed an example program which is capable of mass handling converting files into Long Term Preservation (LTP) suitable formats[11]. The program relies on the ImageMagick library to be installed, as well its accompanying Perl module. The script accepts a list of files on the command line to convert, but in a real world situation if that were not sufficient it could be easily changed to accept a directory, in which it searched for and converted files.

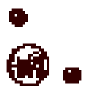


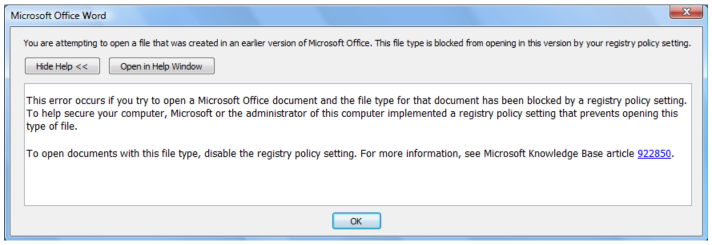
Figure 2: "logo Pur.bmp" converted to a PNG

'NEWS.doc' (found in the 'text' folder) was the second file I attempted to identify. As '.doc' is an extremely common file extension, used by Microsoft Word, I assumed the file was in a Microsoft Word format and so attempted to open it immediately using Microsoft Word 2007, which resulted in an error (see figure 3).

I resolved this by creating a Word 2007 specific 'trusted location'[12] which allowed me to open the file.

The file (see figure 4) is a 13 page newsletter of sorts from Digital Preservation Europe. Saved in an evidently old and deprecated version of the Microsoft Word document format, I suggest migration as the most viable method of preservation.

[10]See Appendix C.3
[11]See Appendix D.1.1
[12]http://office.microsoft.com/en-us/help/HA100319991033.aspx#15



You are attempting to open a file that was created in an earlier version of Microsoft Office. This file type is blocked from opening in this version by your registry policy setting.

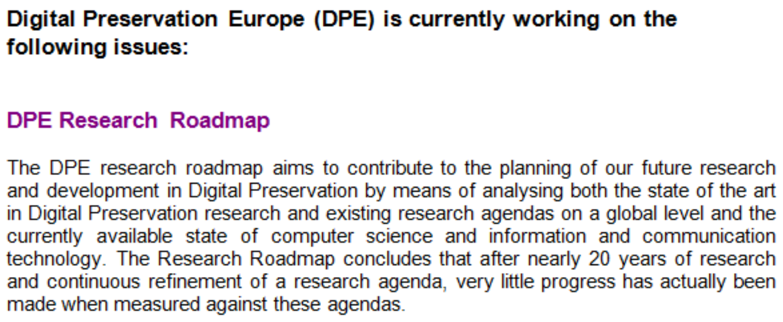
This error occurs if you try to open a Microsoft Office document and the file type for that document has been blocked by a registry policy setting. To help secure your computer, Microsoft or the administrator of this computer implemented a registry policy setting that prevents opening this type of file.

To open documents with this file type, disable the registry policy setting. For more information, see Microsoft Knowledge Base article 922850.

Figure 3: Error message from Microsoft Word 2007

An alternative would be emulation[13] via the use of a Virtual Machine running an older version of Windows and older versions of Microsoft Office installed, but this does not scale well with lots of documents of different versions of the '.doc' format, and could be cumbersome to implement unless absolutely necessary.

However, I advise some caution be taken before making assumptions regarding the quality of the reproduction of the documents by newer versions of Microsoft Word. Upon examining the document, there are minor signs of inconsistencies with the formatting of the document, for example changes in font in an itemised list, and the implied omission of information on the last page, with it ending with 'or visit'. If reproduction of the document needs to be exact, it may be prudent to opt for an emulation route if the conversion of the documents to newer formats does not preserve the formatting and structural content of the document well enough. That said, I will continue under the assumption that Microsoft is capable of reproducing its own formats to as high a quality as possible, and employ a migration strategy accordingly.



Digital Preservation Europe (DPE) is currently working on the following issues:

DPE Research Roadmap

The DPE research roadmap aims to contribute to the planning of our future research and development in Digital Preservation by means of analysing both the state of the art in Digital Preservation research and existing research agendas on a global level and the currently available state of computer science and information and communication technology. The Research Roadmap concludes that after nearly 20 years of research and continuous refinement of a research agenda, very little progress has actually been made when measured against these agendas.

Figure 4: Snippet of the file displayed in Word 2007

As far as migration is concerned, clearly using software developed by the company responsible for the old format is advantageous for ensuring optimal quality of the conversion. The old Microsoft Word document formats are proprietary and badly documented to the outside world, and so whereas it may be possible to migrate via the use

[13]See Appendix C.1

of another package, for example OpenOffice[14], in an optimal scenario Microsoft Word itself should be used.

This being the case, I have developed a tool suitable for mass conversion of such documents to newer formats (see Appendix D.1.2). This 'Word Document Converter' tool interfaces with Microsoft Word, which needs to be installed on the system, which performs the actual conversion. The full range of Word input and output formats are available via the tool, and so you can convert old documents, plaintext files etc to the Word 97-2003 format, the new OOXML format[15], or even PDF (and PDF/A) if using Microsoft Word 2007 with the appropriate add-in.

It should be noted that if opening files of an extremely old format such as 'NEWS.doc', the conversion will fail unless a registry change is applied or a *trusted location* is set up to allow Word to open the files.

Whereas the conversion tool can convert hundreds of documents to a particular format, it's currently bound by the speed at which Microsoft Word can convert each file, as it waits until Word is finished before proceeding to the next file. Depending on how much resources Word takes when converting files, and how well it can handle multiple conversions simultaneously, it may be a better option to adopt threads in the conversion tool, which could potentially speed it up substantially.

For a migration strategy, it's important to decide which output format would be best for files such as 'NEWS.doc'. The conversion tool can handle every format Word can handle, but not all of them are suitable for preservation.

The Florida Digital Archive's 'Recommended Data Formats for Preservation Purposes' document lists the old binary '.doc' formats as "Low Confidence", and even the new OOXML format and PDF as only "Medium Confidence"[1]. Ian Barnes in "Preservation of word processing documents" goes even further, condemning OOXML, RTF, and even PDF[2]. Barnes takes issue with the proprietary nature of OOXML, as well as the Zip compression system used by OOXML and ODF, pointing to the potential for catastrophic failure if corruption of any sort occurs.

Barnes' concludes that the best option would be to employ a custom XML strategy, using the DocBook standard or similar[16]. His reservations aside, for mass migration of documents, hand converting them to DocBook would not be feasible, and OOXML would be a fair choice of format. At the time of writing, OOXML is in the process of becoming an ISO standard[17], which means it will be an open format, maintained by the ISO, and not require a license for use[12].

OOXML will be supported as a standard, and implemented via Microsoft Office for the

[14]http://www.openoffice.org

[15]http://en.wikipedia.org/wiki/Office_Open_XML

[16]http://en.wikipedia.org/wiki/DocBook

[17]http://www.iso.org/iso/catalogue_detail?csnumber=45515

foreseeable future. As the standard is XML, it could potentially be converted to other XML standards, such as ODF or future standards via XSLT[18].

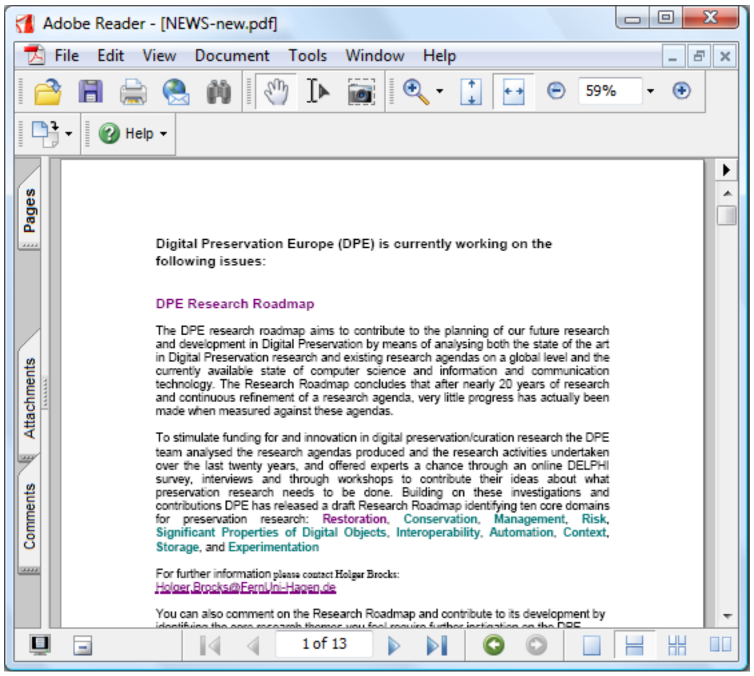
Despite its merits, I think there is a better migration format choice than OOXML, which is PDF/A. Both OOXML and PDF/A are only available in Word 2007 (or above), with the latter being available via a free add-in. PDF/A[19] is a subset of the PDF standard, designed with LTP suitability in mind[18]. It's also an ISO standard (ISO 19005-1:2005), and is a 'preferred format' in the LOC's 'Sustainability of Digital Formats' project[16], and has been given a "High Confidence Level" by the FDA[1].

Therefore, I recommend mass file migration via the use of the 'Word Document Converter' with the output type of 'pdfa' (PDF/A). This format requires Word 2007 with the appropriate PDF add-in, but would ensure maximal quality of the migrated document, as well as assurances to its viability in a LTP scheme, thanks to the *ISO 19005* standard.

This migration strategy offers the advantages of far greater reliability in the format, a format that could be directly accessed and reproduced as digital output without the need for any intermediate steps, as well as far smaller file sizes than say TIFF, as well as the option to embed metadata automatically, as well as manually. Also, the format is very efficient and cost effective[9].

There are a couple of drawbacks to this strategy, namely the migration is to an inherently read-only format, which would make it difficult to get the document into a flexible editable format, although this could be seen as an advantage from an archival standpoint. Furthermore, the reproduction of the document in PDF/A format may not exactly match the original, although differences should be minor and, on the whole, the conversion is very reliable.

[18]http://en.wikipedia.org/wiki/XSLT
[19]http://en.wikipedia.org/wiki/PDF/A


Adobe Reader - [NEWS-new.pdf]
File Edit View Document Tools Window Help
59%
Help
Pages
Attachments
Comments
1 of 13

Digital Preservation Europe (DPE) is currently working on the following issues:

DPE Research Roadmap

The DPE research roadmap aims to contribute to the planning of our future research and development in Digital Preservation by means of analysing both the state of the art in Digital Preservation research and existing research agendas on a global level and the currently available state of computer science and information and communication technology. The Research Roadmap concludes that after nearly 20 years of research and continuous refinement of a research agenda, very little progress has actually been made when measured against these agendas.

To stimulate funding for and innovation in digital preservation/curation research the DPE team analysed the research agendas produced and the research activities undertaken over the last twenty years, and offered experts a chance through an online DELPHI survey, interviews and through workshops to contribute their ideas about what preservation research needs to be done. Building on these investigations and contributions DPE has released a draft Research Roadmap identifying ten core domains for preservation research: Restoration, Conservation, Management, Risk, Significant Properties of Digital Objects, Interoperability, Automation, Context, Storage, and Experimentation

For further information please contact Holger Brocks:
Holger.Brocks@FernUni-Hagen.de

You can also comment on the Research Roadmap and contribute to its development by

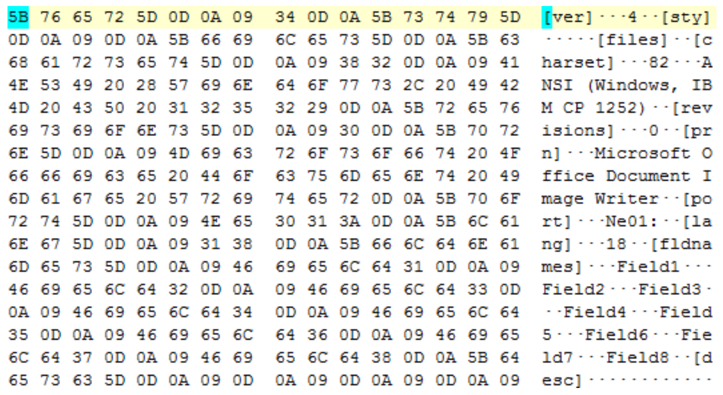
Figure 5: Converted ‘NEWS.doc’ in PDF/A format

‘NEWS’ (found in the ‘unknown 1’ folder) was the third file I attempted to identify. With no file extension, I began by trying to detect the file type using DROID, which was unsuccessful. I then opened the file in a hex editor (see figure 6) and searched Google using keywords from the ASCII view that looked related to the file format. Searching for “+fldnames +charse” reveals multiple files bearing the ‘.SAM’ file extension. FILExt lists this as an ‘Ami Pro Document’[20].

As the format is related to word processing, and hence Microsoft Word, I immediately began looking for a converter, which led me to Microsoft knowledge base article 208863[21]: “How to convert documents between Ami Pro 3.0/3.01 and Word 2000”. The converter pack listed does not seem to be available any more, but several other sources on the internet pointed to an alternate location to download the converter[22].

After installing, I attempted to open the renamed ‘NEWS.sam‘ in Microsoft Word 2007. Unfortunately this resulted in a blank page, and after researching the issue on the web, I have concluded the problem lies with Word 2007, as when I installed the converter on a machine running Word 2000, I could successfully open the file (see fig-

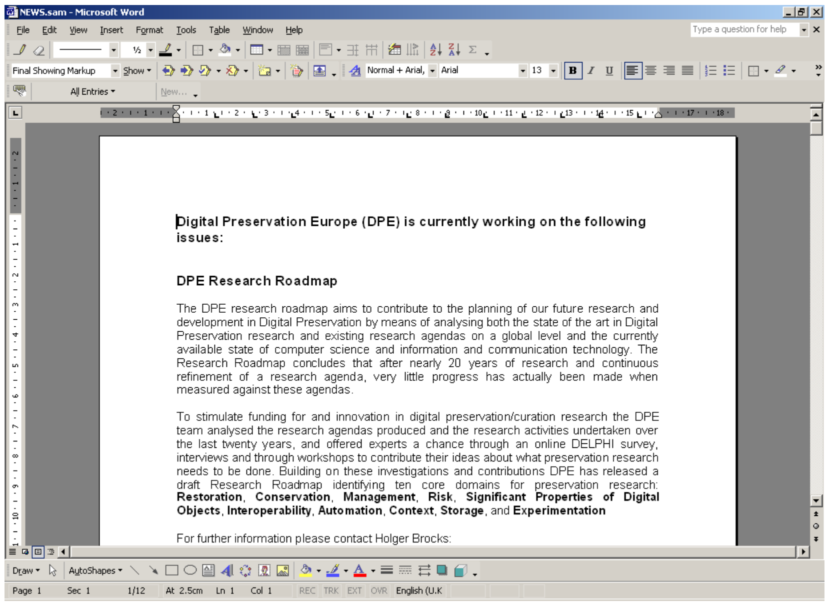
[20]http://en.wikipedia.org/wiki/Am%C3%AD
[21]http://support.microsoft.com/kb/208863
[22]http://www.gmayor.com/downloads.htm



```
5B 76 65 72 5D 0D 0A 09  34 0D 0A 5B 73 74 79 5D  [ver]···4··[sty]
0D 0A 09 0D 0A 5B 66 69  6C 65 73 5D 0D 0A 5B 63  ·····[files]··[c
68 61 72 73 65 74 5D 0D  0A 09 38 32 0D 0A 09 41  harset]···82···A
4E 53 49 20 28 57 69 6E  64 6F 77 73 2C 20 49 42  NSI (Windows, IB
4D 20 43 50 20 31 32 35  32 29 0D 0A 5B 72 65 76  M CP 1252)··[rev
69 73 69 6F 6E 73 5D 0D  0A 09 30 0D 0A 5B 70 72  isions]···0··[pr
6E 5D 0D 0A 09 4D 69 63  72 6F 73 6F 66 74 20 4F  n]···Microsoft O
66 66 69 63 65 20 44 6F  63 75 6D 65 6E 74 20 49  ffice Document I
6D 61 67 65 20 57 72 69  74 65 72 0D 0A 5B 70 6F  mage Writer··[po
72 74 5D 0D 0A 09 4E 65  30 31 3A 0D 0A 5B 6C 61  rt]···Ne01:··[la
6E 67 5D 0D 0A 09 31 38  0D 0A 5B 66 6C 64 6E 61  ng]···18··[fldna
6D 65 73 5D 0D 0A 09 46  69 65 6C 64 31 0D 0A 09  mes]···Field1···
46 69 65 6C 64 32 0D 0A  09 46 69 65 6C 64 33 0D  Field2···Field3·
0A 09 46 69 65 6C 64 34  0D 0A 09 46 69 65 6C 64  ··Field4···Field
35 0D 0A 09 46 69 65 6C  64 36 0D 0A 09 46 69 65  5···Field6···Fie
6C 64 37 0D 0A 09 46 69  65 6C 64 38 0D 0A 5B 64  ld7···Field8··[d
65 73 63 5D 0D 0A 09 0D  0A 09 0D 0A 09 0D 0A 09  esc]············
```

Figure 6: Hex edit view of the first few bytes of ‘NEWS’

ure 7). It is clear that the file holds the same content as the ‘NEWS.doc’ file, and by comparing the two I’m fairly confident the reproduction is to a high standard.


NEWS.sam - Microsoft Word
File Edit View Insert Format Tools Table Window Help
Type a question for help
Final Showing Markup
Show
Normal + Arial,
Arial
13
All Entries
Draw
AutoShapes
Page 1 Sec 1 1/12 At 2.5cm Ln 1 Col 1 REC TRK EXT OVR English (U.K

Digital Preservation Europe (DPE) is currently working on the following issues:

DPE Research Roadmap

The DPE research roadmap aims to contribute to the planning of our future research and development in Digital Preservation by means of analysing both the state of the art in Digital Preservation research and existing research agendas on a global level and the currently available state of computer science and information and communication technology. The Research Roadmap concludes that after nearly 20 years of research and continuous refinement of a research agenda, very little progress has actually been made when measured against these agendas.

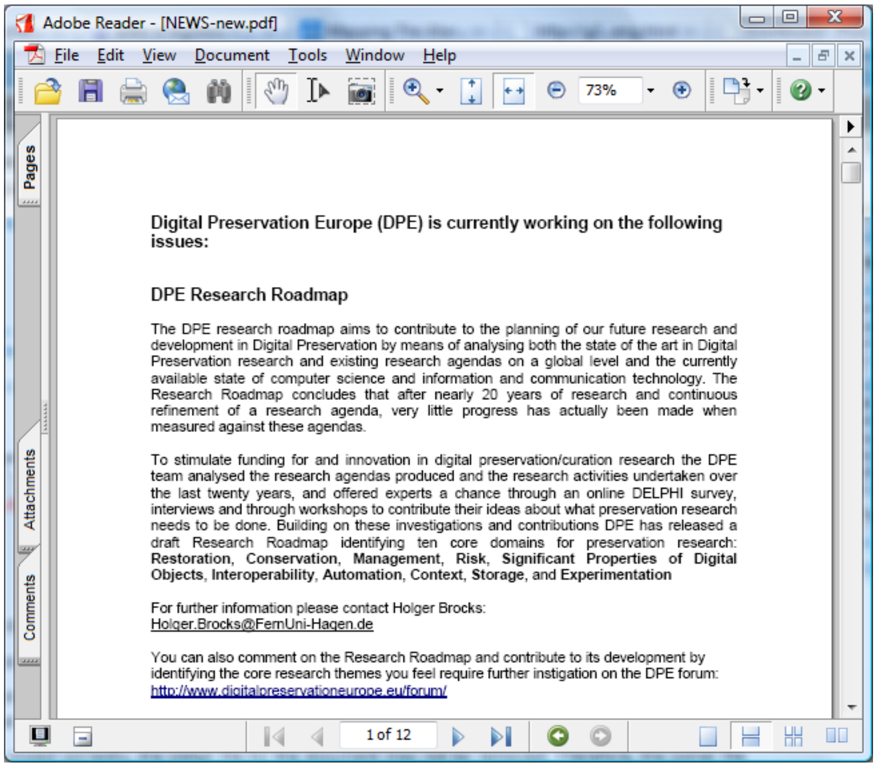
To stimulate funding for and innovation in digital preservation/curation research the DPE team analysed the research agendas produced and the research activities undertaken over the last twenty years, and offered experts a chance through an online DELPHI survey, interviews and through workshops to contribute their ideas about what preservation research needs to be done. Building on these investigations and contributions DPE has released a draft Research Roadmap identifying ten core domains for preservation research: **Restoration**, **Conservation**, **Management**, **Risk**, **Significant Properties of Digital Objects**, **Interoperability**, **Automation**, **Context**, **Storage**, and **Experimentation**

For further information please contact Holger Brocks:

Figure 7: ‘NEWS.sam’ opened in Microsoft Word 2000

In terms of preservation, as Ami Pro is an obsolete word processing format, the same arguments apply as for ‘NEWS.doc’. That being the case, the ‘Word Document Converter’ could be used as part of a migration strategy, as long as Microsoft Word is installed on the PC with the Ami Pro converter mentioned above. As with the Word document, I would strongly recommend the final migration format be PDF/A.

As the PDF add-in only works on Word 2007, and the Ami pro converter only works on versions below 2007, the document converter should be used convert the Ami Pro file to the latest Word native document format supported by that version of Word, and then that file converted to PDF/A via Word 2007 (see figure 8).


Adobe Reader - [NEWS-new.pdf]
File Edit View Document Tools Window Help
73%
Pages
Attachments
Comments
1 of 12

Digital Preservation Europe (DPE) is currently working on the following issues:

DPE Research Roadmap

The DPE research roadmap aims to contribute to the planning of our future research and development in Digital Preservation by means of analysing both the state of the art in Digital Preservation research and existing research agendas on a global level and the currently available state of computer science and information and communication technology. The Research Roadmap concludes that after nearly 20 years of research and continuous refinement of a research agenda, very little progress has actually been made when measured against these agendas.

To stimulate funding for and innovation in digital preservation/curation research the DPE team analysed the research agendas produced and the research activities undertaken over the last twenty years, and offered experts a chance through an online DELPHI survey, interviews and through workshops to contribute their ideas about what preservation research needs to be done. Building on these investigations and contributions DPE has released a draft Research Roadmap identifying ten core domains for preservation research: **Restoration, Conservation, Management, Risk, Significant Properties of Digital Objects, Interoperability, Automation, Context, Storage, and Experimentation**

For further information please contact Holger Brocks:
Holger.Brocks@FernUni-Hagen.de

You can also comment on the Research Roadmap and contribute to its development by identifying the core research themes you feel require further instigation on the DPE forum:
http://www.digitalpreservationeurope.eu/forum/

Figure 8: Converted 'NEWS' in PDF/A format

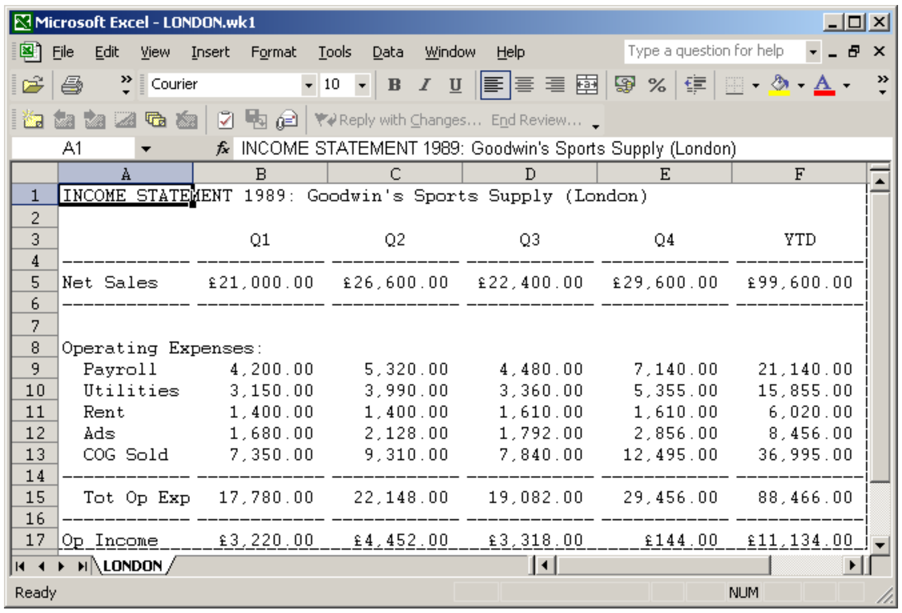
'LONDON', 'QSSOURCE' & 'QSTARGET' (found in the 'unknown 2' folder) were the last files to be identified. With no file type, DROID[23] providing its trademark "The format could not be identified" message, and all other identification tools at my disposable drawing a blank: I again resorted to using a hex editor and searching Google for promising strings. Searching for "+ISAAF +ISAPP" returned results referencing 'Lotus 123'[24] and its .wk4 format[25].

As no copies or free downloads of 'Lotus 123' were evident, and IBM seemingly still trying to sell a later generation of it, I searched for a converter to turn the files into Excel format, or similar. I found a proprietary shareware converter solution called '

[23]See Appendix A.1
[24]http://en.wikipedia.org/wiki/Lotus_1-2-3
[25]http://filext.com/file-extension/wk4

ABC Amber Lotus 1-2-3 Convertor', and an add-in for Microsoft Excel provided by Microsoft[26].

The Microsoft add-in failed to function on Excel 2007, which seems to be somewhat of a running theme of Office 2007 providing no backwards compatibility with anything. I instead turned to Excel 2000, where the add-in successfully installed and could load up the files (see figure 9).



Microsoft Excel - LONDON.wk1
File Edit View Insert Format Tools Data Window Help
Type a question for help
Courier 10
Reply with Changes... End Review...
A1
LONDON
Ready
NUM

A1: INCOME STATEMENT 1989: Goodwin's Sports Supply (London)

	A	B	C	D	E	F
1	INCOME STATEMENT 1989: Goodwin's Sports Supply (London)					
2						
3		Q1	Q2	Q3	Q4	YTD
4	------------	------------	------------	------------	------------	------------
5	Net Sales	£21,000.00	£26,600.00	£22,400.00	£29,600.00	£99,600.00
6	------------	------------	------------	------------	------------	------------
7						
8	Operating Expenses:					
9	Payroll	4,200.00	5,320.00	4,480.00	7,140.00	21,140.00
10	Utilities	3,150.00	3,990.00	3,360.00	5,355.00	15,855.00
11	Rent	1,400.00	1,400.00	1,610.00	1,610.00	6,020.00
12	Ads	1,680.00	2,128.00	1,792.00	2,856.00	8,456.00
13	COG Sold	7,350.00	9,310.00	7,840.00	12,495.00	36,995.00
14	------------	------------	------------	------------	------------	------------
15	Tot Op Exp	17,780.00	22,148.00	19,082.00	29,456.00	88,466.00
16	------------	------------	------------	------------	------------	------------
17	Op Income	£3,220.00	£4,452.00	£3,318.00	£144.00	£11,134.00

Figure 9: 'LONDON.wk4' loaded in Excel 2000

For preservation of the Lotus 1-2-3 file type, migration to a more popular format would be a best case scenario as long as the conversion process is able to preserve the vast majority of the original information. As far as I'm aware, at the time of writing there is no open standard for digital spreadsheet preservation, and so for a migration route, I believe the best option would be to convert to the most well supported format.

As with the previous items, if no migration route is able to adequately preserve the information contained in the original, emulation would be necessary, although I do believe that the Excel converter is capable enough and so will not be pursuing emulation. However, I would strongly recommend a copy of the original files be preserved, as well as the migrated copy.

For mass handling of the documents, I developed a tool very similar to the 'Word Doc-

[26]http://www.microsoft.com/Downloads/details.aspx?FamilyID=dd0697d5-c621-4cdc-981c-4ac92071700d

ument Converter', which I gave the inspired title of the 'Excel Document Converter'[27]. This converter interfaces with Excel in exactly the same way as the 'Word Document Converter' interfaces with Word, and provides a near identical user interface, with a -t flag and a list of files or a directory via a -d flag.

One issue with conversion is that some spreadsheet formats (like Lotus 1-2-3), provide functionality to link values from other sheets. Clearly when migrating to a new format, it would be preferable for these links to remain in tact. This being the case, the converter provides a -u flag which will attempt to do just that.

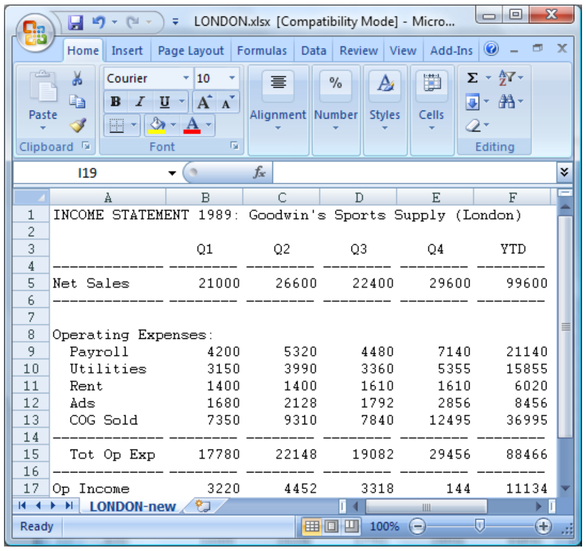
The -u flag works by seeing if the link source file exists in the same directory as the file being converted, and if so whether or not that file has been converted. If the original exists, it will either update the link to the converted copy, or if the file has not yet been converted, it pushes the file to the end of the conversion queue, and re-runs, at which point the file will have been converted. This simplistic approach while very effective, can be brought into an infinite loop by two files linking from each other, to form a deadlock[28]. In these cases, links would have to be resolved manually.

In the long term, which final format the files should be converted to would depend on the purpose of the archiving. If it is enough to simply view a static snapshot of the spreadsheet, then PDF conversion (Excel 2007) would be a good choice. A better choice would be PDF/A, but unfortunately it appears the programmers of Microsoft Excel neglected to add an option to export to PDF/A to their API[29]. That being the case, to get PDF/A files would have to be converted to PDF/A by hand, or a macro program built to simulate this.

PDF while a good preservation format, would be lossful in the case of formulas and anything dynamic about the spreadsheet. This is akin to saving the HTML output of a web page, not the scripts that back-end generated it. If dynamic content is involved, I would recommend the latest OOXML format used for the conversion, although as the add-in for Lotus 1-2-3 and other obsolete formats does not work on this version, an intermediate format would have to be used. That is to say; Excel 2000, for example, should convert the original file to its latest file format, and then that used as input for Excel 2007, which converts it to OOXML. This should be a relatively safe operation, as compatibility between recent native Excel formats is almost flawless.

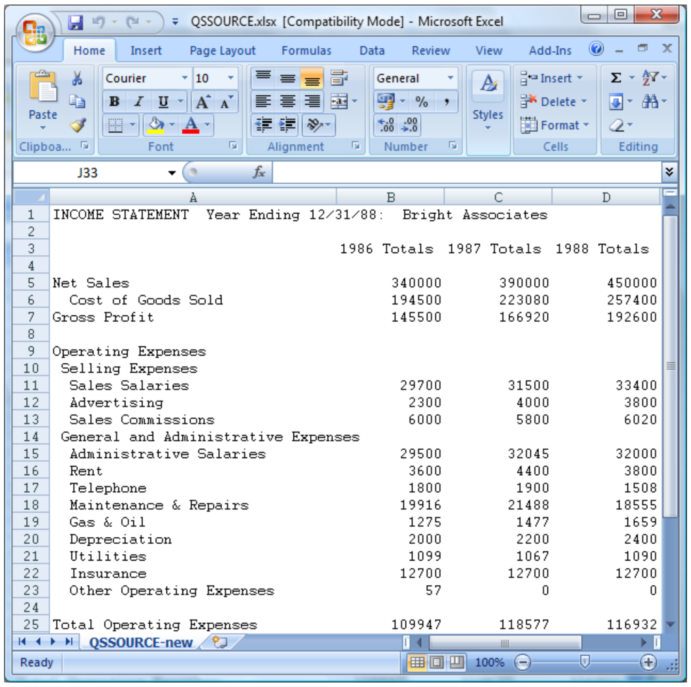
As OOXML is not a preservation standard, if following that route it would be prudent to encapsulate[30] the files with a copy of Microsoft Excel capable of reading them, or at least the OOXML specification pages, to ensure in the long term the files have a reader capable of understanding them fully.

[27]See Appendix D.1.3
[28]http://en.wikipedia.org/wiki/Deadlock
[29]http://msdn.microsoft.com/en-us/library/bb238907.aspx
[30]See Appendix C.2



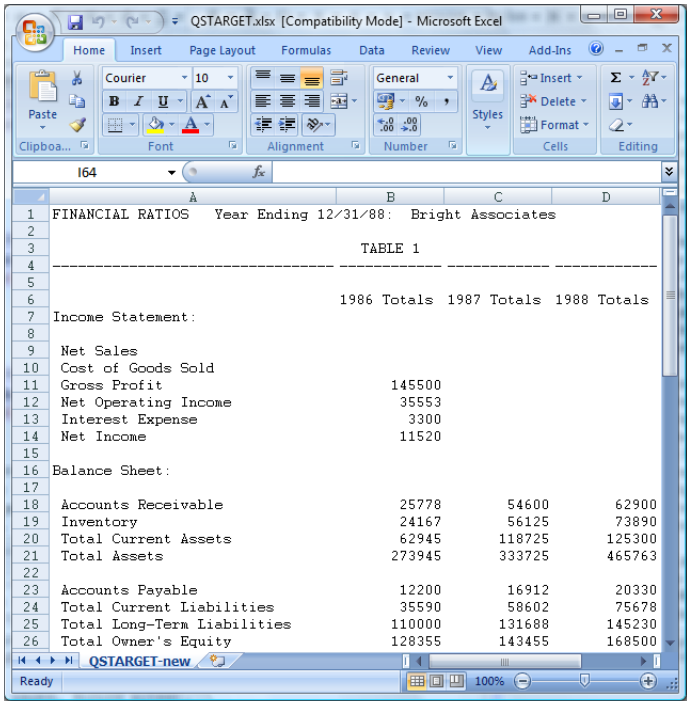
	A	B	C	D	E	F
1	INCOME STATEMENT 1989: Goodwin's Sports Supply (London)					
2						
3		Q1	Q2	Q3	Q4	YTD
4	--------------	---------	---------	---------	---------	---------
5	Net Sales	21000	26600	22400	29600	99600
6	--------------	---------	---------	---------	---------	---------
7						
8	Operating Expenses:					
9	Payroll	4200	5320	4480	7140	21140
10	Utilities	3150	3990	3360	5355	15855
11	Rent	1400	1400	1610	1610	6020
12	Ads	1680	2128	1792	2856	8456
13	COG Sold	7350	9310	7840	12495	36995
14	--------------	---------	---------	---------	---------	---------
15	Tot Op Exp	17780	22148	19082	29456	88466
16	--------------	---------	---------	---------	---------	---------
17	Op Income	3220	4452	3318	144	11134

Figure 10: ‘LONDON’ in exported XLS format



	A	B	C	D
1	INCOME STATEMENT Year Ending 12/31/88: Bright Associates			
2				
3		1986 Totals	1987 Totals	1988 Totals
4				
5	Net Sales	340000	390000	450000
6	Cost of Goods Sold	194500	223080	257400
7	Gross Profit	145500	166920	192600
8				
9	Operating Expenses			
10	Selling Expenses			
11	Sales Salaries	29700	31500	33400
12	Advertising	2300	4000	3800
13	Sales Commissions	6000	5800	6020
14	General and Administrative Expenses			
15	Administrative Salaries	29500	32045	32000
16	Rent	3600	4400	3800
17	Telephone	1800	1900	1508
18	Maintenance & Repairs	19916	21488	18555
19	Gas & Oil	1275	1477	1659
20	Depreciation	2000	2200	2400
21	Utilities	1099	1067	1090
22	Insurance	12700	12700	12700
23	Other Operating Expenses	57	0	0
24				
25	Total Operating Expenses	109947	118577	116932

Figure 11: Snippet of ‘QSSOURCE’ in exported XLS format



	A	B	C	D
1	FINANCIAL RATIOS Year Ending 12/31/88: Bright Associates			
2				
3		TABLE 1		
4	--------------------------------	------------	------------	------------
5				
6		1986 Totals	1987 Totals	1988 Totals
7	Income Statement:			
8				
9	Net Sales			
10	Cost of Goods Sold			
11	Gross Profit	145500		
12	Net Operating Income	35553		
13	Interest Expense	3300		
14	Net Income	11520		
15				
16	Balance Sheet:			
17				
18	Accounts Receivable	25778	54600	62900
19	Inventory	24167	56125	73890
20	Total Current Assets	62945	118725	125300
21	Total Assets	273945	333725	465763
22				
23	Accounts Payable	12200	16912	20330
24	Total Current Liabilities	35590	58602	75678
25	Total Long-Term Liabilities	110000	131688	145230
26	Total Owner's Equity	128355	143455	168500

Figure 12: Snippet of 'QSTARGET' in exported XLS format

For the migration of the three files, I have chosen OOXML (XLSX extension) as the migration format, due to the formulae used in the spreadsheets. In figure 12, several fields (e.g Gross Profit) are linked to the corresponding fields in the QSSOURCE worksheet (figure 11). Other formulae in QSTARGET worksheet could not be correctly converted, due to (I suspect) fields moving offsets during the conversion. Had PDF been used as the output format, the information would be lost, however with OOXML, the formulae are accessible and can be fixed manually. The same applies for formatting, such as monetary values.

I have tried to use as much free and open source technology as possible with the preservation strategies, and so the image conversion should incur no licensing overhead. The other documents require potentially two separate versions of Microsoft Office, and Windows to go along with it. Those costs are fixed and unknown, and would generally be under $1000 dollars.

Labour costs would be potentially higher for the document and spreadsheet formats, as depending on the desired quality levels, some things would have to be hand converted, and files checked for quality control. We can be reasonably safe in assuming image conversion could occur without such reservations, however.

Physical storage costs have not been taken into account, under the assumption that hard drive space is relatively cheap and it is doubtful these types of files could take up enough room to make a noticeable impact on a budget. Also, I would assume a system would be in place if mass handling was involved, and such costs would have been laid out up front before hand anyway.

2.2 Images from a Legacy Computer Gaming Platform

2.2.1 Scenario Description

An image archive received a donation from an artist representing working material from his early years. While ingesting the data into the image archive repository, the system failed to identify some of the file formats. The artists cannot remember the name of the particular application or the computer platform. He also found a related file for one type of the images, but he does not know what the file is. Can you display the images? Include the images in your report in an appropriate form.

Your task is to:

1. Identify the application and computer platform that the files come from.
2. Display the images in an appropriate form.
3. Propose valid preservation alternatives pointing out their advantages and disadvantages.
4. Implement one or more of the preservation strategies (for example emulation or migration).
5. Evaluate and compare their performance.

2.2.2 Scenario Solution

Of the three files: 'GRID.PIC', 'SKY.PIC', 'XLART.XFD', I decided to at first concentrate on the latter. I searched FILExt for the 'XFD' extension, with the third result being 'Atari Disk Image', which fits perfectly with the scenario.

Armed with the knowledge that an XFD file was an Atari Disk Image, I then searched the Web for a program that could run it. I found one in the form of the Xformer 2000 Atari emulator[31].

[31] http://www.emulators.com/xformer.htm

Unfortunately Xformer 2000 continually crashed (possibly due to being run on Vista), and so I quickly moved to another emulator: atari++[32] (See figure 13).

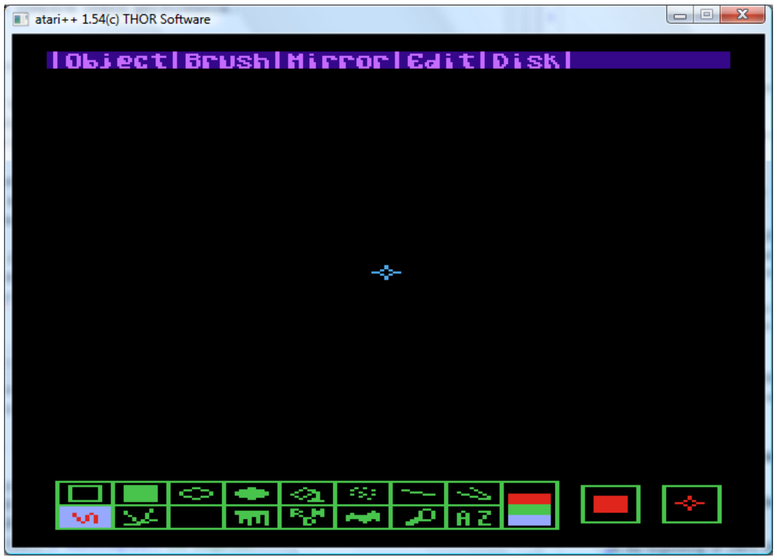

atari++ 1.54(c) THOR Software
|Object|Brush|Mirror|Edit|Disk|
A Z

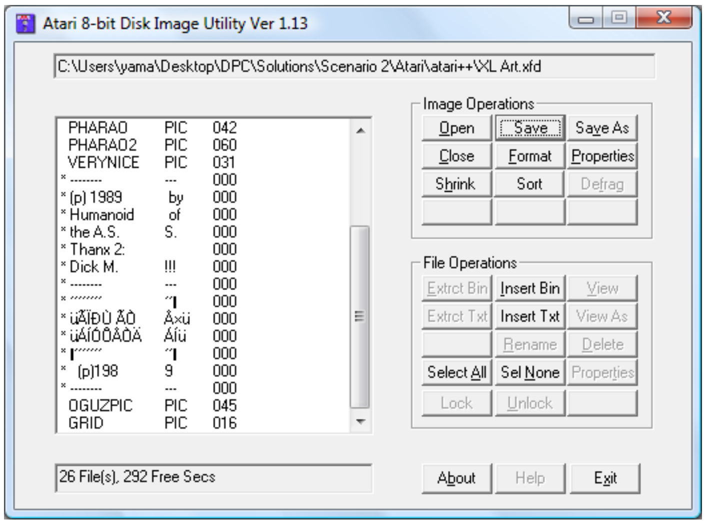
Figure 13: XL-ART running under atai++

At this point I attempted to load the pictures, but unfortunately I could not find a method of accessing the host hard drive from XL-ART running on the emulator. To get around this issue, I searched for a tool which would allow me to edit the XFD disk image directly, and place the PIC files onto the image. I found such a tool (AtrUtil) on ‘Ken’s Atari 8-bit Page’[33], and attempted to open ‘XLART.XFD’ using it. Unfortunately the tool was unable to open the file, however I verified that the tool could open other XFD files, and so drew the conclusion that the problem was specific to that particular XFD. That being the case, I searched Google for another XL-ART XFD file, hoping it would fair better. Eventually I located one, and proceeded to add ‘GRID.PIC’ to it using AtrUtil (see figure 14).

With ‘GRID.PIC’ on the disk image file, I could load it using XL-ART (see figure 15).

Using this technique, I also loaded ‘SKY.PIC’ (see figure 16). I then saved ‘SKY.PIC’ as a ‘MIC’ file via the Disk menu of XL-ART, as I was curious as to what affect this would have as a picture. Transferring it from the disk image via AtrUtil, I noticed in a hex editor the format was similar to that of the ‘CAR’ and ‘EINSTEIN’ files, with regards to the use of the null bytes (hexadecmial 00) as a marker, and a file size of exactly 7684 bytes. Assuming ‘CAR’ and ‘EINSTEIN’ could too be opened by XL-ART, I gave them both MIC extensions, and successfully loaded them via XL-ART

[32]http://www.math.tu-berlin.de/ thor/atari++
[33]http://atari.ksiders.tzo.com/a8emulators.html - AtrUtil95


Atari 8-bit Disk Image Utility Ver 1.13

C:\Users\yama\Desktop\DPC\Solutions\Scenario 2\Atari\atari++\XL Art.xfd

```
 PHARAO      PIC   042
 PHARAO2     PIC   060
 VERYNICE    PIC   031
* --------   ---   000
* (p) 1989    by   000
* Humanoid    of   000
* the A.S.    S.   000
* Thanx 2:         000
* Dick M.    !!!   000
* --------   ---   000
* ''''''''    ´'I  000
* üÄÍÐÙ ÄÖ   Äxü   000
* üÄÍÖÖÄÖÄ   Äíü   000
* I''''''''   ´'I  000
*  (p)198     9    000
* --------   ---   000
 OGUZPIC     PIC   045
 GRID        PIC   016
```

26 File(s), 292 Free Secs

Figure 14: AtrUtil editing ‘XL ART.xfd’

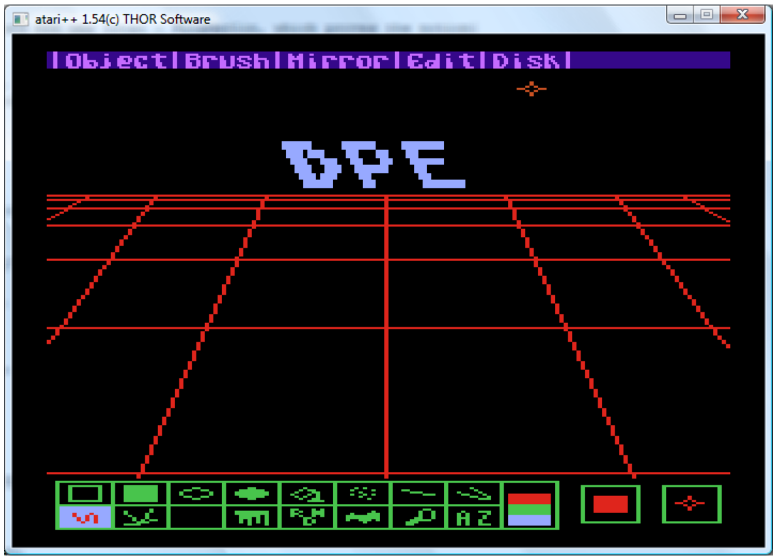

atari++ 1.54(c) THOR Software
|Object|Brush|Mirror|Edit|Disk|
DPE
A Z

Figure 15: ‘GRID.PIC’ loaded in XL-ART

(see figures 17 and 18).

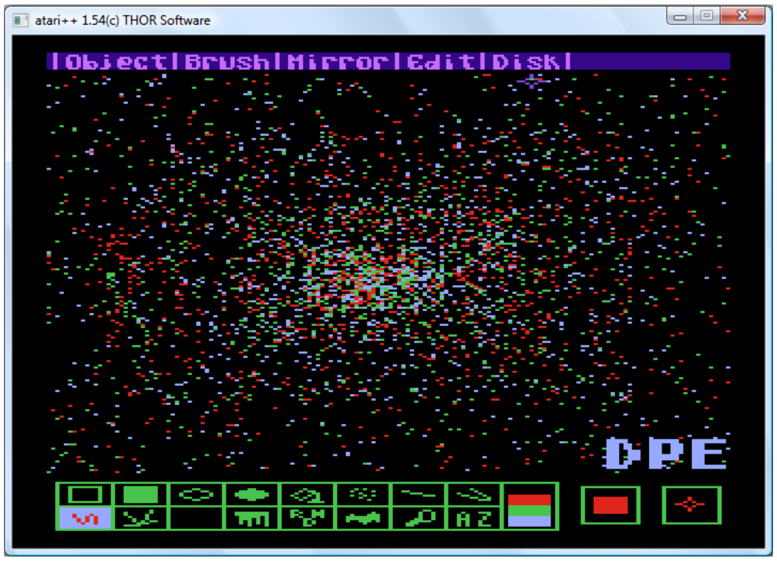

atari++ 1.54(c) THOR Software
|Object|Brush|Mirror|Edit|Disk|
DPE

Figure 16: ‘CAR.PIC’ loaded in XL-ART

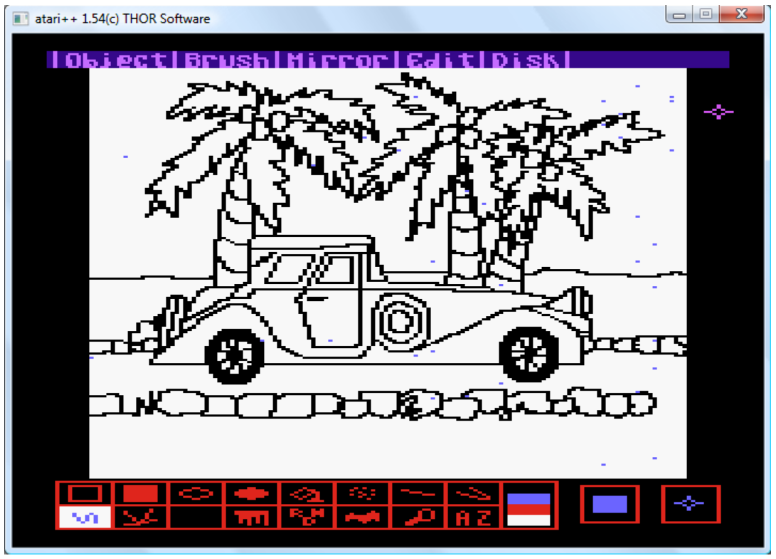

atari++ 1.54(c) THOR Software
|Object|Brush|Mirror|Edit|Disk|

Figure 17: ‘CAR.MIC’ loaded in XL-ART

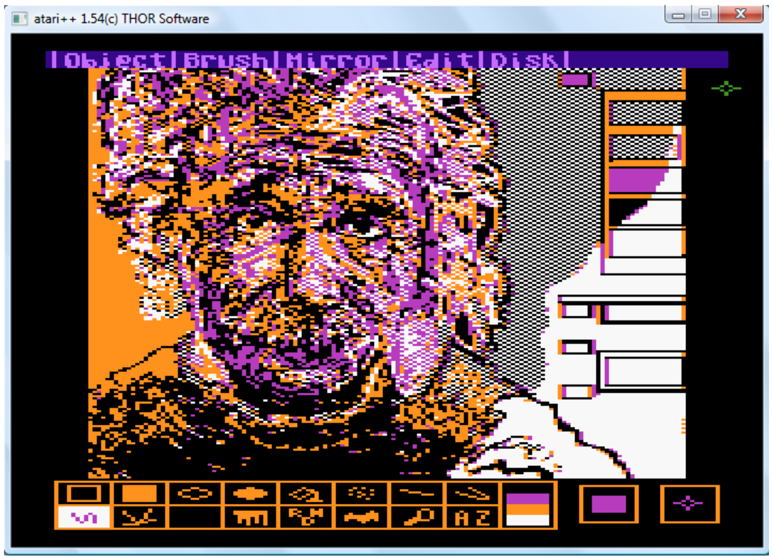

atari++ 1.54(c) THOR Software
|Object|Brush|Mirror|Edit|Disk|

Figure 18: 'EINSTEIN.MIC' loaded in XL-ART

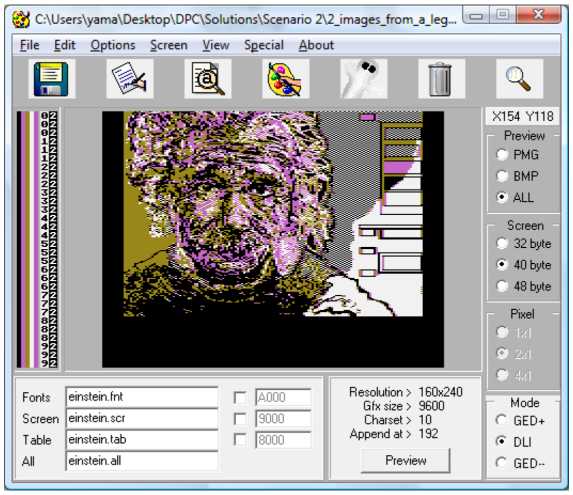
The two obvious preservation strategies for these files are emulation and migration[34]. Emulation would require substantial effort, and is certainly not my preferred option. Namely, it would require an Atari emulator (atari++ as our example), an image file viewer for the emulator (XL-ART as our example), and a method of making the images accessible. Issues could arise with LTP due to the need to run the emulator, and so the emulator itself would almost certainly need to be included with its host operating system for long term preservation, for example a Virtual Machine (VM) running Windows Vista. There is also the problem of making the images accessible to the image file viewer. I could not locate a method of mapping a local drive running on the host system to something understandable by an image file viewer running on the emulator. Without this, files would have to be added to a disk image (which can be problematic due to a maximum size of images and other limitations), which would be laborious and potentially hard to automate without custom tools. Furthermore, viewing the archived material would require the emulator to be run, and the interface of the image file viewer to be navigated, and so instructions and documentation would need to be written up to explain the functionality of the viewer. In short, whereas emulation would be possible, the effort required to implement it as a strategy is not conducive to a fast or efficient preservation strategy.

Migration appears a better option, as long as a migration route is possible without losing any image quality from the original. My preference would be for a Perl command line utility to perform a conversion, but unfortunately whereas I did find obscure spec-

[34]See Appendix C.1 and C.3

ifications for both the MIC[35] and PIC[36] formats, I was not able to write a converter from Antic[37] E graphic codes to their RGB equivalents.

Instead, I located a Windows application called Graph2Font[38] which could load both formats (see figure 19) and save them as BMPs. From there it would be trivial to convert the BMPs to PNG, for example by using the 'Image Converter'[39]. It is noticeable that the images shown in Graph2Font have different colour shades to that of the atari++ XL-Art view. After some study, it is evident that the fault lies with atari++'s reproduction, and that the Graph2Font color palette is the correct one.



Fonts: einstein.fnt ☐ A000
Screen: einstein.scr ☐ 9000
Table: einstein.tab ☐ 8000
All: einstein.all

Resolution > 160x240
Gfx size > 9600
Charset > 10
Append at > 192

Preview: PMG BMP [✓ ALL]
Screen: 32 byte [✓ 40 byte] 48 byte
Pixel: 1x1 [✓ 2x1] 4x1
Mode: GED+ [✓ DLI] GED--

Figure 19: 'EINSTEIN.MIC' loaded in Graph2Font

Migration via Graph2Font and the 'Image Converter' provides an exact replica of the image in a format suitable for long term preservation, with no software operational costs, and at a high rate of efficiency. Emulation for this kind of preservation would offer no obvious benefits, and would require an order of magnitude more effort, not to mention increased costs.

[35]http://g2f.atari8.info/instrukcja_eng.html
[36]http://www.pokey.nl/xoops/modules/megazine/index.php?op=viewarticle&artid=213
[37]http://en.wikipedia.org/wiki/ANTIC
[38]http://g2f.atari8.info/
[39]See Appendix D.1.1

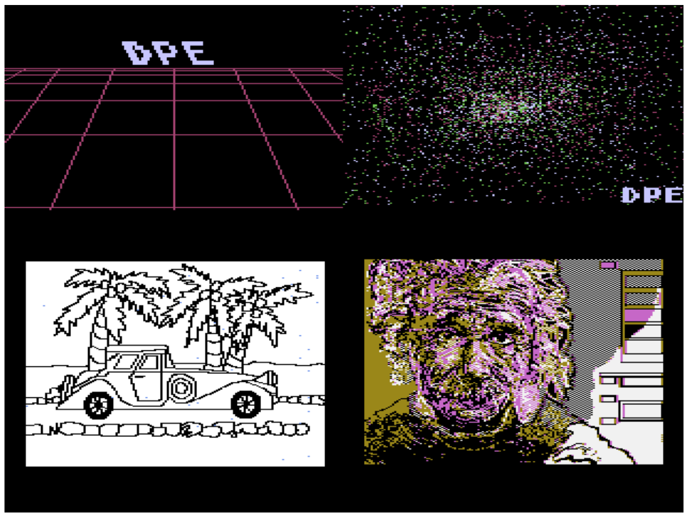


Figure 20: All 4 pictures in PNG format

2.3 Obsolete Database

2.3.1 Scenario Description

In the beginning of 2003, the Porto Regional Archive (ADP) initiated a project called DigitArq. The goal of the project was to bring together its various finding aids, previously scattered throughout the archive in many different forms and formats, into a single centralised repository based on international standards such as ISAD(G) and EAD. The planned repository would enable the standardisation of all archival procedures and the development of new data services such as search mechanisms and description tools. However, in late 2007 a fire broke out in the server room destroying the server that held all the information produced over the last 25 years. In addition to this all the backup tapes that were kept in a cabinet in the adjoining room were destroyed. Around 80% of the information had been synchronised with a similar repository at the National Archives in Lisbon, and this information was easily recovered. The other 20% had been migrated from an old database that was still kept at the archive but had not been used since 1990. The ADP staff were unable to use the database, so they decided to hire a digital preservation expert to do the job.

Your task is to:

1. Identify the database system that is necessary to interpret the provided data files.
2. Provide access to the data of the database.

3. Propose an adequate procedure to migrate the records from the old system to the new one (please provide output files, scripts or standards to support your answer).

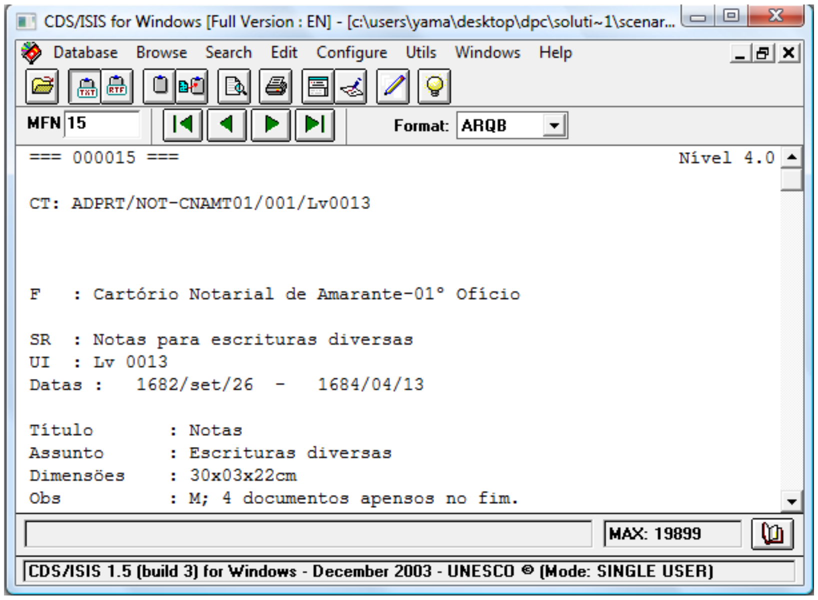
4. Propose a disaster recovery plan so that if a similar incident takes place the disruption will be kept to a minimum.

2.3.2 Scenario Solution

The format of the obsolete database system was easily found by searching for a collection of file extensions in Google: "+mst +fdt +fst +ifp +database". The first entries all relate to CDS/ISIS[40], the database format.

The Wikipedia page for CDS/ISIS shows the CDS/ISIS software package is developed by a company called UNESCO. Their website provides a Windows software package called WinISIS capable of opening CDS/ISIS databases[41].

Access to the database is provided through the use of the WinISIS software package (see figure 21).


CDS/ISIS for Windows [Full Version : EN] - [c:\users\yama\desktop\dpc\soluti~1\scenar...
Database Browse Search Edit Configure Utils Windows Help
MFN 15
Format: ARQB
MAX: 19899
CDS/ISIS 1.5 (build 3) for Windows - December 2003 - UNESCO © (Mode: SINGLE USER)

```
=== 000015 ===                                            Nível 4.0

CT: ADPRT/NOT-CNAMT01/001/Lv0013



F    : Cartório Notarial de Amarante-01° Ofício

SR   : Notas para escrituras diversas
UI   : Lv 0013
Datas :   1682/set/26   -   1684/04/13

Título        : Notas
Assunto       : Escrituras diversas
Dimensões     : 30x03x22cm
Obs           : M; 4 documentos apensos no fim.
```

Figure 21: Screenshot of WinISIS displaying a record

[40]http://en.wikipedia.org/wiki/CDS_ISIS
[41]http://www.unesco.org/isis/files/winisislicense.html

Access to the database alone via the use of a Windows executable is limited in scope, and so migration is necessary to make use of the data in the new system.

WinISIS provides an "Export to XML" facility which can convert the entire database to XML format. However, as WinISIS is platform restricted, I have developed a Perl script to perform a similar task (see Appendix D.2.1).

The CDS/ISIS Exporter Perl script can convert a CDS/ISIS database to the same XML format as used by WinISIS with the default settings. It can also export the database to MAchine-Readable Cataloging (MARC) format[42].

Migration wise, it may be possible to migrate directly from the XML representation of the database to an international standard such as ISAG(G) or EAD, although information is sparse. I could not locate any sources to do this directly, which was part of the motivation for writing the 'CDS/ISIS Exporter'. Whereas its XML exporting capabilities are clearly not on par with WinISIS' built in facilities, the script can be easily modified to change the output to something that maybe easier to convert to an established format.

Building on this, (US)MARC exporting is also available. MARC is another internationally recognised standard, designed by the US Library of Congress[43]. Its wide adoption for storing bibliographic data, which is a not too dissimilar field to that of DigitArq, makes it a logical candidate format to aid migration.

It may be possible for the MARC format could be used directly in the new system, but if not the format is far more accessible than the CDS/ISIS XML format, and so would provide an easier migration[44] route.

From the MARC file, it seems perfectly possible to convert the data into formats such as EAD, either directly by perhaps using the tools built for the MALVINE Project[45] in the 'MALVINE SGML Feasibility Study'[46]. Another option would be to convert the MARC file to MARCXML, and convert to EAD[47] by using a crosswalk[13, 3]. As MARCXML and EAD are both XML standards, this could be achieved more easily using Extensible Stylesheet Language Transformations (XSLT)[48].

In terms of the scenario, I will propose a disaster recovery plan that builds upon what was implemented previously. Several things need to be taken into account to ensure no data loss occurs if a disaster strikes:

[42]http://en.wikipedia.org/wiki/MARC_standards
[43]http://www.loc.gov/marc/
[44]See Appendix C.3
[45]http://www.malvine.org/
[46]http://helmer.hit.uib.no/malvine/EADpage.html
[47]http://en.wikipedia.org/wiki/Encoded_Archival_Description
[48]http://en.wikipedia.org/wiki/XSL_Transformations

- As well as regular backup tapes being kept near the server for easy replacement, more backups should be stored at fixed intervals and stored externally e.g four backup tapes stored in a different building, the oldest one being updated on a weekly basis with a full backup.
- A redundancy system employed whereby each month a backup is loaded onto a cheap dedicated system to ensure that the backup files are working and can be restored from, and providing a small safety net in case the worst happens.
- Continue synchronising the repository with the Lisbon system, and if possible extend that to other similar systems.

In the scenario the disruption was caused by the oversight of underestimating the scope of possible damage, and hence not keeping external backups. I would assume an archive project had at least basic disaster recovery precautions that are the same throughout the IT industry, and so I have focused on the special needs of the particular project.

2.4 Electronic Art

2.4.1 Scenario Description

Founded in 1987, the Prix Ars Electronica is an interdisciplinary platform for digital art and media culture. The Prix Ars Electronica is one of the most important awards for creativity and pioneering spirit in the field of digital media. With the rapid change of software tools and frameworks for multimedia authoring their artworks are in danger of becoming inaccessible and unusable. You have been asked to preserve four of these historical digital artworks for future generations and to develop appropriate digital preservation strategies. Task

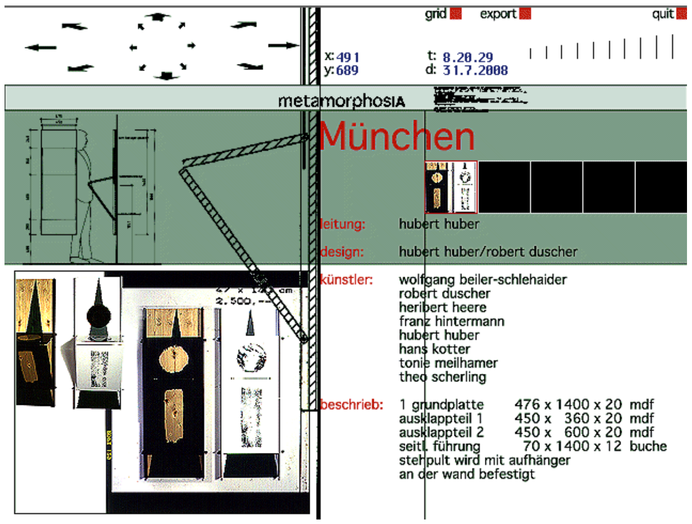
Your task is to:

1. Display the multimedia art, provide screenshots and a description of the steps taken.
2. Decide which aspects of the artworks to preserve, and identify their significant properties.
3. Develop a set of different preservation strategies for the four pieces of multimedia art provided, that have the potential to address different aspects of the artwork.
4. Point out the differences in the strategies with respect to the characteristics of the preserved artworks and their suitability.

5. (Optional) Implement part of the preservation strategies you have developed and submit code for this.

2.4.2 Scenario Solution

The first piece of artwork, Metamorphosia, I displayed natively in Windows Vista merely by running the 'MIA97.exe' executable, with no changes necessary (see figure 22).



x: 491
y: 689

t: 8.20.29
d: 31.7.2008

metamorphosIA

München

leitung: hubert huber

design: hubert huber/robert duscher

künstler: wolfgang beiler-schlehaider
robert duscher
heribert heere
franz hintermann
hubert huber
hans kotter
tonie meilhamer
theo scherling

beschrieb:

1 grundplatte	476 x 1400 x 20 mdf
ausklappteil 1	450 x 360 x 20 mdf
ausklappteil 2	450 x 600 x 20 mdf
seitl. führung	70 x 1400 x 12 buche

stehpult wird mit aufhänger
an der wand befestigt

Figure 22: Screenshot of 'Metamorphosia'

The second piece of artwork, 'Cyberinstallation Botschaft An Europa 2098', I displayed natively in Windows Vista by using the compatibility mode and setting it to 'Windows 95', I also set the colour depth to 256 colours (see figure 23).

The third piece of artwork, 'Cyborg's Eye', I displayed using Mozilla Firefox 3[49] (see figure 24), although it functions in Microsoft Internet Explorer 7 as well. To run, merely open the 'intro.html' page in either of the browsers.

The final piece of artwork, 'Interactive Paint', I displayed natively in Windows Vista(see figure 25), after fixing an error that prevented the program from loading by creating an empty text file named *kolofon.txt* in the same directory as the main executable. Windows Vista seems capable of running the basic program, although some of the interactive nature of the colouring of pictures seems to be distorted, and using the compatibility mode could not resolve this.

[49]www.mozilla.com/firefox/

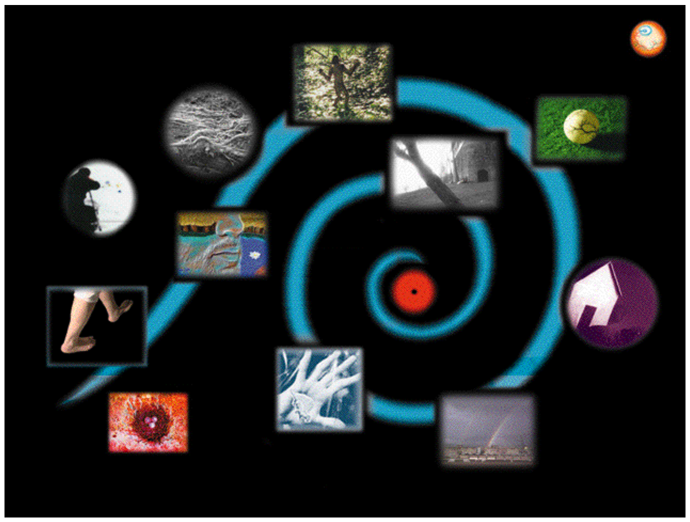


Figure 23: Screenshot of ‘Cyberinstallation Botschaft An Europa 2098’

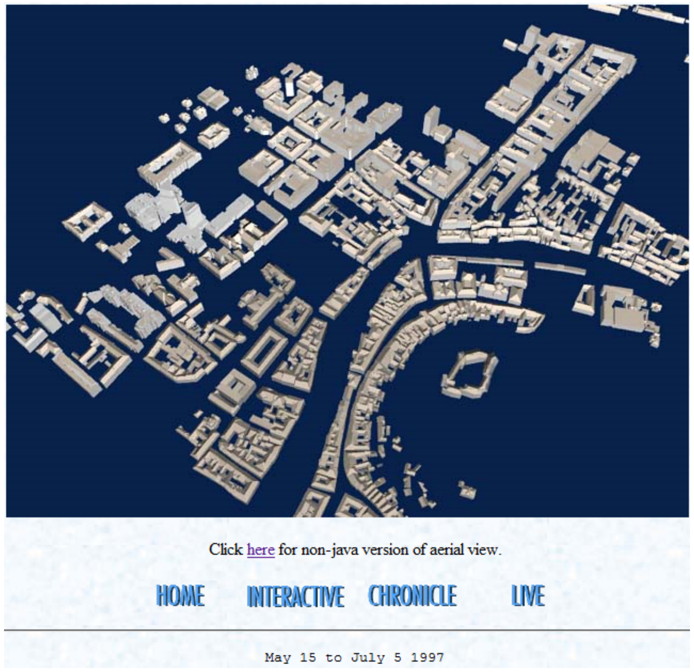
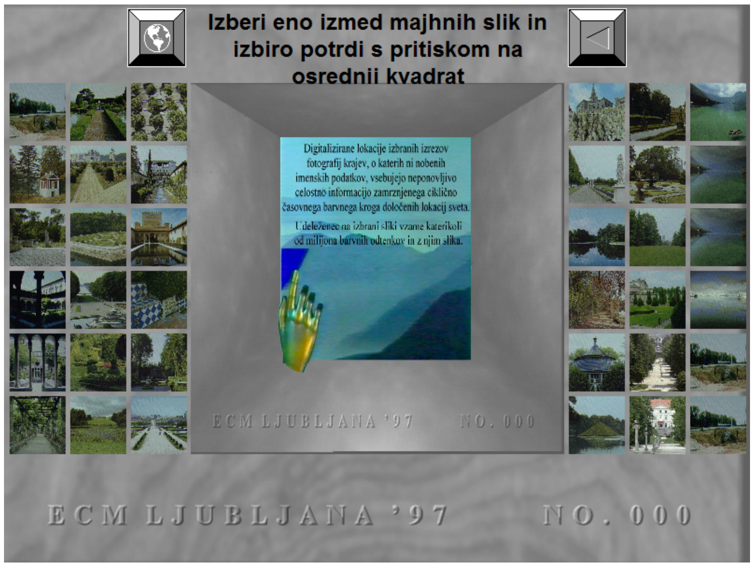


Figure 24: Snippet of ‘Cyborg’s Eye’ running on Firefox

As is often the case, the only two main preservation strategies for these pieces of artwork are emulation or migration. Due to the interactive nature of the pieces, they


Izberi eno izmed majhnih slik in izbiro potrdi s pritiskom na osrednii kvadrat

Digitalizirane lokacije izbranih izrezov fotografij krajev, o katerih ni nobenih imenskih podatkov, vsebujejo neponovljivo celostno informacijo zamrznjenega ciklično časovnega barvnega kroga določenih lokacij sveta.

Udeleženec na izbrani sliki vzame katerikoli od milijona barvnih odtenkov in z njim slika.

Figure 25: Screenshot of 'Interactive Paint'

cannot adequately be represented in a non-computational medium.

Emulation can take many forms, and so one preservation strategy I propose for all of the pieces is full environmental reproduction and emulation. This would involve using a virtual machine (VM) with Windows 95 running as the operating system. Appropriate plugins would have to be installed on this environment to show some aspects of the artwork. For example, the 'Cyborg's Eye' requires the use of an obsolete ActiveX browser component for 'Surround Video', and 'Metamorphosia' may require the Macromedia Shockwave plugin[50]. This strategy would require the least amount of work and effort to implement, and should be able to capture all aspects of the artwork with the correct environment. However, the artwork would be tied to the emulation making it difficult to decouple aspects of it, file sizes would be large, and there would be an overhead upfront cost of purchasing virtualisation equipment/software and licensing any software required to emulate the environment.

As a example of how this could be implemented, as all the pieces presumably are designed to run on a typical computer of the day, a VMWare virtual machine could be set up with a copy of Windows 95. This would provide the native APIs and library code capable of running the executable files. Software of the time, for example Macromedia Shockwave and the browser plugins would have to be sought, although Adobe holds archived shockwave versions[51], and the other plugins can be found via Google

[50]http://www.adobe.com/products/shockwaveplayer/

[51]http://www.adobe.com/shockwave/download/?P1_Prod_Version=SWArchive7.02 from Internet Explorer

searches.

In the majority of cases, migration would be preferable to emulation if the 'essence' of the original could be captured. For each piece of artwork, I will propose a migration strategy, listing any shortcomings and caveats, along with comparison to the general emulation strategy listed above. From there, I will draw a conclusion as to whether migration is the more viable option, and if so give an example of how it could be conducted.

Metamorphosia Metamorphosia is an abstract art 'experience' for lack of a better word. The program provides an interface of sorts to different sections which consist of a montage of motion, images, sounds, and interactivity. Its written for a German audience, but presentation wise resembles a subdued version of a Larry Carlson[52] piece.

The artwork has a difficult to identify interactive element, where content can seemingly be moved around at will forming and opening new content through rules I fail to comprehend. I do not have any migration recommendations, a cursory look at the format files provides no clues, and so I am unable to tell how much of the dynamic content is generated by the executable, and how much can be extracted. From the *strings*[53] tool, I have determined the executable was compiled using the WATCOM compiler[54], although this information in itself is not useful to aid migration.

Due to this lack of information and uncertainty, the only viable option I see is emulation as described above in the general purpose emulation strategy. Attempting a migration strategy without knowledge of the internal workings would be folly, and may result in catastrophic loss of information. At a best case scenario, you'd have an indeterminate amount of content that may or may not be salvageable to any degree.

Cyberinstallation Botschaft An Europa 2098 is a cross between a menu based segmented image viewer, and a slide show player. The main menu gives access to different areas of the artwork, that contain text and images.

Basic research of the file types show that it was crated using Macromedia[55] Director, now Adobe Director[56]. The DXR files are protected Adobe DIR files, which appears to make editing them impossible. However the main executable, 'DIOS_W95.exe' has a packed DIR file which can seemingly be extracted using a tool called 'Swift Xena Pro'[57].

[52]http://www.larrycarlson.com
[53]http://linux.about.com/library/cmd/blcmdl1_strings.htm
[54]http://en.wikipedia.org/wiki/Watcom_C_compiler
[55]http://en.wikipedia.org/wiki/Macromedia
[56]http://en.wikipedia.org/wiki/Adobe_Director
[57]http://www.buraks.com/swifty/xena.html

Full migration looks problematic, as even if you could update the main director file, the encrypted content files are of a much older version, and it is unclear whether or not it is possible to ‘mix & match’.

Instead, partial migration could be possible through the use of screenshots and moving the navigation to a broader format. For example, the DXR files are playable in Internet Explorer 7 (with the Adobe Flash[58] plugin), showing trainsitions of screens between each section. That is to say, whereas in the program you would traverse a sections content using the menu, when opened in Internet Explorer, each section is transitioned in term after a fixed delay.

Hence, if a screenshot were somehow taken of each of the separate pages from the DXR file (or the images extracted, which looks possible), through manual navigation of the program (and by merely viewing the navigational icon thumbnail pictures and matching it to the corresponding slide), you could map out the navigation of the system. Instead of screenshots, it may be possible to use dirOpener[59] to extract information from the DXR files, and load this into Adobe Director to extract the images directly.

Assuming the images were extracted, you could then put each separate slide on a web page, and link between them to emulate the effect of the original program. This, combined with the background sound from the extracted DIR file, would be a migration of the full content of the original.

The migration strategy outlined above is time consuming, and would require considerably cost and effort. However, it is preferable to emulation in terms of LTP and accessibility of the content, and so would be my recommended choice, assuming the images could easily be extracted by the use of dirOpener or similar tools. It has the advantage of being cross platform, and suitable for LTP if the image format was PNG or similar[60]. On the other hand, the reproduction would not be the original, but I think with the possibility of all of the content being migrated and the core navigation system conceptually preserved, this is vastly outweighed by the benefits.

Cyborg’s Eye is an interactive website which provides a virtual tour of Ljubljana, through the use of Java applets for ‘aerial views’, images, surround video, and live streams.

The live streams functionality does not work, evidently due to the web provider being shut down, no doubt years ago. The surround video also did not function in Firefox or Internet Exploer by default, and there’s one VRML[61] file that also didn’t function.

[58]http://en.wikipedia.org/wiki/Adobe_Flash
[59]http://j-roen.blogspot.com/2005/11/diropener.html
[60]See Appendix B
[61]http://en.wikipedia.org/wiki/VRML

Disregarding the live interactive element, the majority of the content is in the form of the applet (or images if Java is not installed), surround video and general images make up the bulk of the content. Whereas emulation could restore all of these, HTML in itself is very well supported and potentially available without the need for emulation, as are the other items.

The major obstacles for migrating are the surround sound files. The Java applet should is compiled into bytecode, and therefore runnable on any Java certified vVM. This means no migration is necessary, and by very virtue of the VM, we can be reasonable confident the applet will continue to work in the future.

The surround video uses obsolete technology, in the form of SVH files. This technology seems primarily routed at the very early days of the modern internet, and so is badly documented. However, the surround videos are accessible from tInternet Explorer 7 using an ActiveX control.

In the HTML source, the code to invoke the control is as follows:

```
CLASSID="clsid:7142BA01-8BDF-11CF-9E23-0000E8A37440"
CODEBASE="bin/svideo.cab#version=1,0,0,180"
```

The CLASSID property identifies the control, and the CODEBASE property points to its physical location in terms of a URI. I found a working control by searching for the CLASSID, '7142BA01-8BDF-11CF-9E23-0000E8A37440' and downloaded the corresponding CAB[62] file, and placing it in a 'bin' folder, I created in the 'cye' directory.

The CODEBASE property also includes 'version' which dictates what version of the control must be used. If the version of the control does not match the one specified, the control is not loaded. To get around this limitation, I executed a small perl program in the 'cye' directory to remove these version references, allowing the control to load (should be one line):

```
>perl -pi.bak -e "BEGIN{@BAK=map{$_.$^I}@ARGV=glob qq(@ARGV)}
  s/#version=1,0,0,180//g; END{ unlink @BAK or warn $!,@BAK}"
  sv_*.html
```

When run from the 'cye' directory, it will edit all the HTML files containing references to the surround video ActiveX control, and remove the version string.

With this done, the surround video files are accessible from Internet Explorer (see figure 26). Internet Explorer 7 will ask if the control should be run, via a yellow bar

[62]http://en.wikipedia.org/wiki/Cabinet_(file_format)

at the top of the page. The control should be allowed and enabled to make use of the embedded surround video.

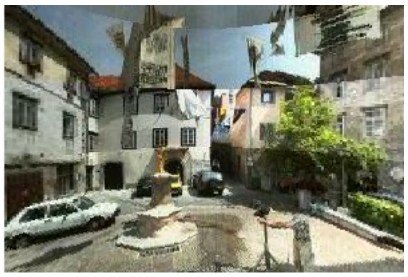


Figure 26: Surround video in ‘Cyborg’s Eye’

Migrating the surround video files would be problematic, and so I propose the migration strategy make use of ActiveX controls, with additional documentation in to form basic encapsulation[63]. Losing the surround video would be an unacceptable loss for the artwork, and so the only alternative is to attempt to migrate the format. This could be done by recording a full cycle of the surround video, and providing it in AVI[64] or QuickTime[65] format. The recording could be done with a utility like FRAPS[66], however a recording would lose the interactive element of the surround video and some of the video, as you would be unable to change direction, and the video would only show one angle.

The VRML file is accessible in Firefox and Internet Explorer (see figure 27) via a browser plugin, such as Cosmo Player[67]. As VRML is an ISO standard[68] with several implementation players, it should be accessible in the future. However, the standard has been replaced by the X3D[69], and so it could be migrated to the newer format, although as there’s only one VRML file and it contains very little content, this may not be justifiable.

[63]See Appendix C.2
[64]http://en.wikipedia.org/wiki/AVI
[65]http://en.wikipedia.org/wiki/QuickTime#QuickTime_file_format
[66]http://www.fraps.com/
[67]http://cic.nist.gov/vrml/cosmoplayer.html
[68]http://www.web3d.org/x3d/specifications/vrml/
[69]http://en.wikipedia.org/wiki/X3D

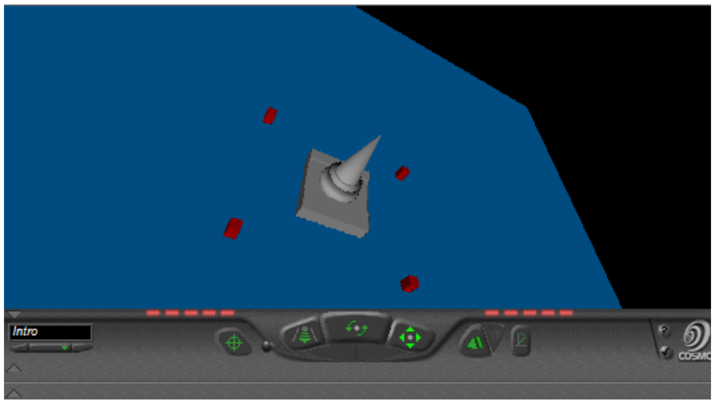


Figure 27: VRML 'Parlament' in 'Cyborg's Eye'

In conclusion, I'd recommend migration, as all major aspects of the artwork are accessible using modern technology. The core of the artwork is HTML, a very well supported and implemented standard, and the surround video still works using the same browser plugin technology, a decade later. That said, I'd recommend including basic documentation on VRML and the surround video format, along with a browser with plugins installed.

Interactive Paint is, as the name suggests, an interactive paint program at heart. The concept appears to be that the program was originally designed to be used by a touch screen system, it provides a multilingual interface to pick a picture from a selection, edit it (i.e change the colouring in some manner), sign it, and print it off. The image manipulation is somewhat broken in Vista, and so I can only guess what the effects are on the image, although while editing some audio files are played with a French speaker (inexplicably French regardless of language chosen) remaking on something (as to what remains a mystery to your humble author, who has next to no knowledge of the language). The finished image is then supposedly printed off, although this component didn't work for me, but then again I don't have a printer.

The core of this artworks seems to be the interactive element, where the user is creating artwork using the pictures as a base. This being the case, whereas it would be perfectly possible to compress the bitmap images to PNG format and the WAV files to FLAC[70], the interactive element is lost.

Without emulation, it would be very difficult to migrate the interactive elements of this piece of artwork, given it relies heavily on an executable. Basic disassembly and string

[70]FLAC is a free lossless audio compressor, although its viability for LTP is still under scrutiny)

searching[71] shows it was created using an antiquated version of Delphi[72]. Porting the application to ensure it will run on new and later platforms would be extremely difficult and unreliable.

From a migration standpoint, the only way to ensure the interactive element is preserved, is to literally recode the executable using newer technology such as Java[73]. However, this would be a complex and arduous task, and faithful reproduction would be challenging. It is even debatable whether this would still constitute the same piece of artwork, although I believe the artistic value is contained in the concept and choice of audio and visual elements, rather than the executable format.

In conclusion, putting the reliance on later versions of Microsoft Windows to maintain backwards compatibility aside, emulation is the only viable route to keep the interactive element of the artwork.

For migration, I'd recommend converting the images to PNG format, possibly converting the WAV files to FLAC, depending on your confidence of the lossless format, and documenting the interactive functionality and concept, as well as including the original executable and related files in archived format.

I believe migration a better route for preservation, as although it would make the program unusable (with the incompatible image files), the concept would be preserved, and the file size would be much lower. I believe thorough documentation of the program (with screenshots of an example use), would suffice in terms of preservation, along with the media content, and the tradeoff is worth the loss of the interactive functionality.

2.5 Web Archiving

2.5.1 Scenario Description

Your company wants to preserve their website to document their growth and evolution over time. You are asked to analyse different preservation strategies for websites. The developed strategies will be applied to two internet domains and to analyse their advantages and disadvantages. Task

Your task is to:

1. Devise at least two preservation strategies for websites, highlighting their respective advantages and disadvantages.

[71] http://linux.about.com/library/cmd/blcmdl1_strings.htm
[72] http://en.wikipedia.org/wiki/Delphi
[73] http://java.sun.com

2. Harvest the two following internet domains documenting the date, time and method of harvesting:
    - www.digitalpreservationeurope.eu with a depth of 3 (approximately 15MB of data)
    - www.rai.it with a depth of 2 (approximately 40MB of data)
3. Apply the identified preservation strategies to the harvested websites, compare and document the results (for example storage size, processing time, presentation quality). Give an estimate of the resources (such as time, storage, effort, costs) required to deploy the strategies.

2.5.2 Scenario Solution

Web archiving is an extremely complex task, in a very new part of digital preservation. Not many web archive exists, and so I will propose two general systems suitable for archiving based on different needs.

The first system is designed at simply displaying a website in the typical environment of the times, with no additional information on metadata. In short, a snapshot of the frontend of a website with no special effort made to accomodate exotic data types. Fundamentally, harvested data from a website in the age of dynamic and server generated pages means that any archive can only be a brief snapshot of a sites state, potentially crippled of all its functionality in the case of so called 'Web 2.0'[74] sites, that rely on Ajax[75] technologies. As This being the case, I feel for a snapshot of a site, it is not worth the effort or resources to ensure maximum compatibility when so much will be lost anyway in terms of the backend.

This system is designed to be easy to implement, generic, and as easy to operate as possible. It's separated into two parts, harvesting of the information and viewing it at a later time, following the KISS principle[76].

To harvest the data, I have opted for using the GNU 'wget'[77] utility. *wget* is command line based, cross platform, and simple to use. This allows it to be potentially incorporated into a larger application, and its open source nature allows tailoring if required.

The *wget* options employed are as follows:

```
>wget -r -l DEPTH -p -k http://www.example.org
```

[74]http://en.wikipedia.org/wiki/Web_2.0

[75]http://en.wikipedia.org/wiki/Ajax_(programming)

[76]http://en.wikipedia.org/wiki/KISS_principle

[77]www.gnu.org/software/wget

The -r flag specified that links from the initial page should be followed, and the -l flag specifies to which depth (hops). DEPTH should be replaced by a number, with the default being 5. To map an entire domain, this would be set to 0 or -1. The -p flag ensures all the files (e.g images) required to view a given HTML page are also downloaded. Finally, the -k flag rewrites all links in terms of relative paths, which would allow the website to be viewed ‘offline’, as it were.

These options form a basic generic web spider capable of harvesting a website to a specific depth. The only change to the pages is the rewriting of the links, which is required for viewing. It is an intentional decision not to do anything further to the page, following the KISS principle, as too many changes may corrupt the website or contaminate the content.

While harvesting the example websites with this method, I noted that harvesting the DPE website to depth 3 resulted in far more files than advertised (over double when I cancelled the harvest), and so I reduced the depth of 2. The opposite was noticed with *www.rai.it*, and so its depth was raised to 3.

The harvested content from the DPE and rai.it websites take up 7.36mb and 13.5mb on disk respectively. No compression is used in this strategy for simplicity, and because of the risk of a slight corrupted of an encrypted file leading to catastrophic data loss.

The harvested sites are viewable as-is via a web browser, with any links to the same domain[78] rewritten to point to the local files. This forms the basis of the preservation, and the first part of the strategy.

The second part of the strategy involves ensuring the data remains viewable. For this, I recommend the use of a virtual machine. As an example, VMWare offer a ’browser appliance’[79], which is a small virtual machine designed for browsing, including Firefox running under a lightweight Ubuntu[80] system. As a VMWare virtual machine, it is cross platform and able to run on all major operating systems that have the free VMWare installation. Preservation can take the form of a website hosting the harvested domain in folders, and being accessed by the virtual machine, or the purchase of a full VMWare license, and then editing the virtual machine as to include the harvested domains on file.

In conclusion, this strategy provides a simplistic and generic method of website preservation, useful for the small scale archiving of website snapshots to be viewed on a whim. It is not ‘heavy duty’, does not store metadata on the harvested sites, and provides no method of ensuring data on the sites can be viewed. It relies on the fact that a well designed website worth archiving should not rely on anything exotic, but use well established technologies that would be included with the VM viewer.

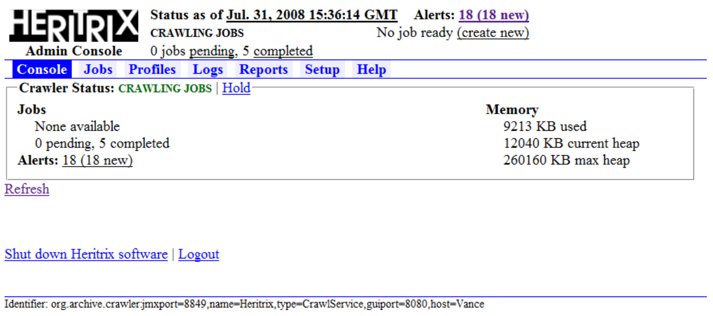
[78]subdomains are not treated as the same domain, to change this consult the *wget* documentation

[79]http://www.vmware.com/appliances/directory/browserapp.html

[80]http://en.wikipedia.org/wiki/Ubuntu

The second strategy is more involved, and so I will explain the basic concept and omit a specific implementation due to the range of uses available.

This strategy uses open source technology, and relies on the wisdom of the internet archive[81] as a base. Sites are crawled using Heritrix[82], an opensource 'industrial strength' Java webcrawler, developed and maintained by the internet archive. Heritrix provides a web based UI for crawling (see figure 28).



Status as of Jul. 31, 2008 15:36:14 GMT   Alerts: 18 (18 new)
CRAWLING JOBS   No job ready (create new)
0 jobs pending, 5 completed

Crawler Status: CRAWLING JOBS | Hold

Jobs
None available
0 pending, 5 completed
Alerts: 18 (18 new)

Memory
9213 KB used
12040 KB current heap
260160 KB max heap

Identifier: org.archive.crawler:jmxport=8849,name=Heritrix,type=CrawlService,guiport=8080,host=Vance

Figure 28: Heritrix Web UI

The internet archive provides the biggest and oldest website archive database, which has a public frontend called the 'Wayback Machine'[83]. Heritrix is used to crawl websites, in a highly configurable manner, and store the results in an ARC[84] file. This file can then be queries and data extracted by open source tools[85].

This strategy provides far more archival information, such as MIME types, date of crawl, and a checksums to ensure integrity. The ARC format also allows entire domains to reside in one file, resorting to a compression format that is susceptible to corruption causing massive loss of data.

Viewing of the ARC files depends on exactly what you wish to gain from them. This strategy is not for simply viewing the content again, but much broader in scope, for example collecting statistics of tracking change over time. As such, it's difficult to provide examples without defining a limited scope.

Using the mature and well maintained codebase of the internet archive is a clear preferable choice for advanced archiving. As with any opensource code, things can be configured and tailored towards specific needs, as well as no costs due to licensing.

[81]http://crawler.archive.org
[82]http://crawler.archive.org
[83]http://www.archive.org/web/web.php
[84]http://www.archive.org/web/researcher/ArcFileFormat.php
[85]http://www.archive.org/web/researcher/tool_documentation.php

resources and costs of the two strategies are not really comparable as I've aimed them at different purposes. The first one is small scale and designed for snapshots, using free technology, and designed to be as automated as possible. In a general scenario, resources could be flexible as the harvesting tool allows configurable delays between requests, and so processor resources would be minimal, as would the machine specifications required to perform the harvesting. Time and effort should also be minimal, as the strategy is designed to be extremely simple, easy to set-up, with minimum hassle.

The second strategy may require considerably more resources and effort, if used on an enterprise scale. Depending on what is desired from the archive, custom tools may have to be developed, or existing ones modified, and a frontend for the data put in place. That said, the harvesting is again taken care of by Heritrix, and with the option to set up custom harvesting profiles, the set-up cost would be a one time event.

The results of the two are not comparable, they apply to different needs on different scales. On a simple scale, Heritrix would allow compression to be used more easily, resulting in smaller files, containing more information, and better crawling. However the information would not be immediately accessible, and a big overhead would be incurred providing access to it for a user.

Appendices

A File Identification Tools/Resources

A.1 DROID

Description from the website[86]: *DROID (Digital Record Object Identification) is a software tool developed by The National Archives to perform automated batch identification of file formats. Developed by its Digital Preservation Department as part of its broader digital preservation activities, DROID is designed to meet the fundamental requirement of any digital repository to be able to identify the precise format of all stored digital objects, and to link that identification to a central registry of technical information about that format and its dependencies.*

DROID is signature based, with the database containing information on file extensions, and magic numbers[87]. It requires Java[88] to run, and provides a Graphical User Interface (GUI).

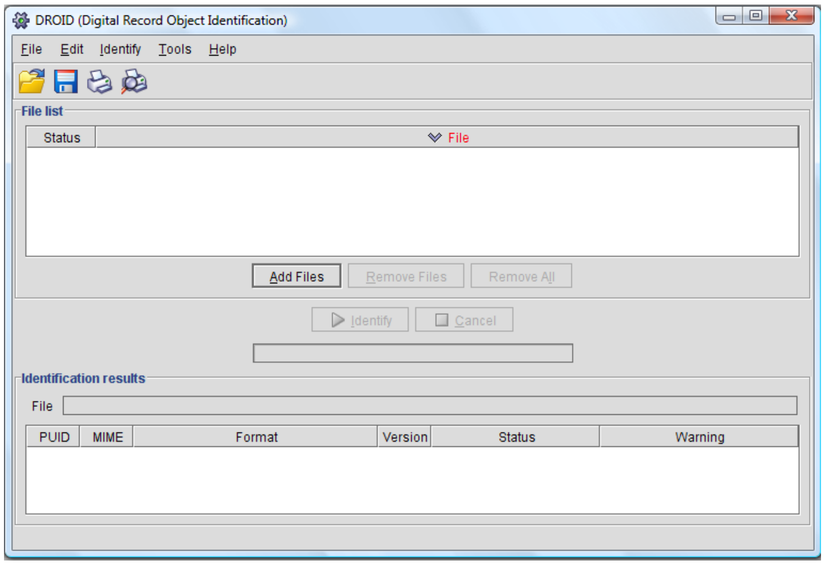


Figure 29: Screenshot of the DROID 3.0

What DROID lacks in identification prowess it makes up for in usability and aesthetic

[86]http://droid.sourceforge.net
[87]http://en.wikipedia.org/wiki/File_format#Magic_number
[88]http://java.sun.com

value. Not to put too fine a point on it, it's lots of icing without any cake. Out of all the files I attempted to identify, it failed to identify a single one. Given a known file, e.g one with a well known extension, it works, but with more exotic files - no such luck. It appears the magic number database is extremely limited, as well as being heuristically simplistic, as was evident in the Scenario 1 identification of the graphics file. The header was slightly corrupted, but even a cursory check past the first few bytes for a match would have been successful.

Overall, DROID appears a good idea in theory, with a good framework to support it. In practice, its database is too small to be of much value. Hopefully in the coming years it will expand, and add better heuristics to go with its polished looks.

A.2 file

The Linux *file*[89] utility attempts to classify files using a range of methods. It works by conducting language tests, file system tests and magic number tests. For most file identification, it is the magic number tests which are relevant, and which provide *file* with the information it needs to classify a file.

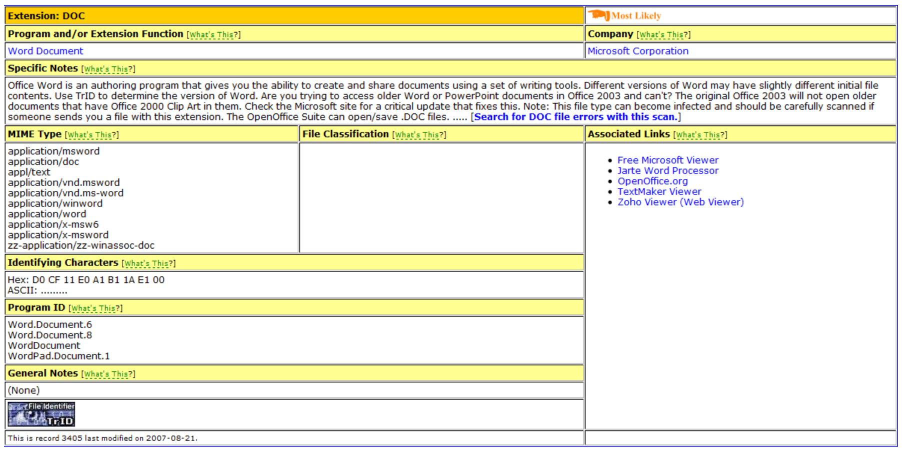
file is an old utility, included on all popular Linux distributions, which makes it perfect for quickly trying to detect the format of a file. Its large database of common format magic numbers means it should correctly identify the majority of common file formats in use on Linux systems.

However, the tool is not really suited for identifying exotic or very old Windows platform-dependent files.

A.3 FILExt

FILExt (http://filext.com) is a website dedicated to cataloguing known file extensions. It allows you to search for a file extension, and returns all the known matches for that extension. The results page displays lots of useful information about each result, such as the file type name, originating company, mime type, and useful notes about the file format (see figure 30).

[89]http://linux.about.com/library/cmd/blcmdl1_file.htm



Extension: DOC | Most Likely

Program and/or Extension Function [What's This?]

Word Document

Company [What's This?]

Microsoft Corporation

Specific Notes [What's This?]

Office Word is an authoring program that gives you the ability to create and share documents using a set of writing tools. Different versions of Word may have slightly different initial file contents. Use TrID to determine the version of Word. Are you trying to access older Word or PowerPoint documents in Office 2003 and can't? The original Office 2003 will not open older documents that have Office 2000 Clip Art in them. Check the Microsoft site for a critical update that fixes this. Note: This file type can become infected and should be carefully scanned if someone sends you a file with this extension. The OpenOffice Suite can open/save .DOC files. ..... [**Search for DOC file errors with this scan.**]

MIME Type [What's This?]

application/msword
application/doc
appl/text
application/vnd.msword
application/vnd.ms-word
application/winword
application/word
application/x-msw6
application/x-msword
zz-application/zz-winassoc-doc

File Classification [What's This?]

Associated Links [What's This?]

- Free Microsoft Viewer
- Jarte Word Processor
- OpenOffice.org
- TextMaker Viewer
- Zoho Viewer (Web Viewer)

Identifying Characters [What's This?]

Hex: D0 CF 11 E0 A1 B1 1A E1 00
ASCII: ....

Program ID [What's This?]

Word.Document.6
Word.Document.8
WordDocument
WordPad.Document.1

General Notes [What's This?]

(None)

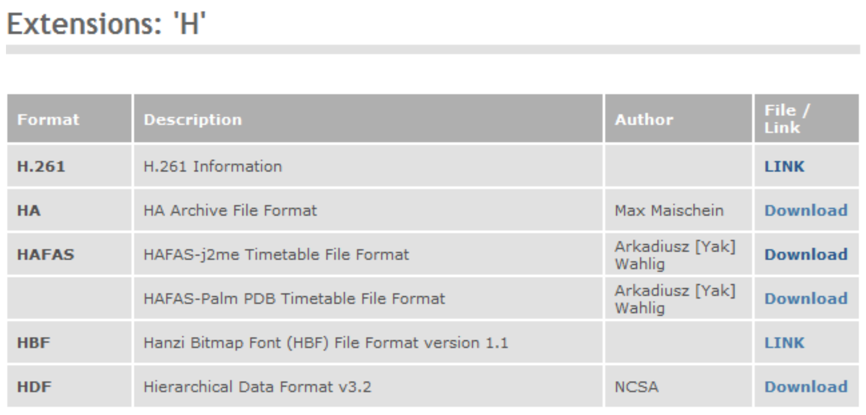
File Identifier TrID

This is record 3405 last modified on 2007-08-21.

Figure 30: Screenshot of FILExt results

A.4 Wotsit

Wotsit (http://www.filext.org) is another website dedicated to file extensions, similar to FILExt. Apart from being an alternative resource which may have some extensions listed that FILExt does not, it also serves a slightly different purpose. Whereas FILExt documents lots of information and background of the formats associated with a particular extension, Wotsit instead provides links to downloadable format specifications, that describe the underlying structure of a format from a programmer's perspective. Wotsit's use, therefore, is more related to working with a format to manipulate it or extract information, as opposed to simply describing where it can be found.



Extensions: 'H'

Format	Description	Author	File / Link
H.261	H.261 Information		**LINK**
HA	HA Archive File Format	Max Maischein	**Download**
HAFAS	HAFAS-j2me Timetable File Format	Arkadiusz [Yak] Wahlig	**Download**
	HAFAS-Palm PDB Timetable File Format	Arkadiusz [Yak] Wahlig	**Download**
HBF	Hanzi Bitmap Font (HBF) File Format version 1.1		**LINK**
HDF	Hierarchical Data Format v3.2	NCSA	**Download**

Figure 31: Screenshot of Wotsit alphabetical results listings

B Digital Image Preservation Formats

Choosing which format to use to preserve digital images is an important decision that shouldn't be taken lightly. What follows is an evaluation of image formats suitable for long term preservation of digital images. The evaluation is primarily aimed at raster image formats, although some formats are general purpose.

B.1 TIFF

TIFF is a tag-based file format designed for storing raster images. The current version is Revision 6.0, published in 1992[15]. The format in an uncompressed state has been labelled "High Confidence Level" for preserving raster images by the Florida Digital Archive (FDA)[1], as well as labelled a "Preferred Deposit Format" by the Arts and Humanities Data Service (AHDS)[4]. The AHDS also go on to state it is the *de facto standard* for image preservation, although it further states an uncompressed TIFF of at least v6 is the only suitable TIFF for preservation purposes.

The TIFF was originally created jointly by Aldus[90] and Microsoft. Today, the specification is controlled by Adobe Systems Incorporated. TIFF does not require a license for use, and its proliferation since conception has resulted it in its wide adoption owing to its suitability as a long term image preservation format.

B.2 PNG

The Portable Network Graphics (PNG) format was originally designed to replace the GIF89a standard for use on the internet[14] in 1995, by being a free an open standard. The related GIF patents are now expired, but the PNG format still holds value as a general purpose raster image format.

PNG has been given a "High Confidence Level" by the FDA and is listed as a "Preferred Deposit Format" by the AHDS, matching the TIFF format. It's open nature and well documented specification[91] as an ISO standard (15948:2003) as well as good platform support (supported by all major browsers and by the latest versions of the three main desktop operating systems (Windows, Linux, and Mac OSX), makes it suitable as a long term image preservation format.

[90]http://en.wikipedia.org/wiki/Aldus
[91]See http://www.w3.org/TR/PNG/

B.3 JPEG2000

The JPEG2000 format is a revision of the JPEG format which includes among other things lossless compression. It has been given a "Medium Confidence Level" by the FDA, and marked as a "Problematic Deposit Format" by the AHDS, citing slow uptake and a lack of industry support[1, 4].

However, a later study (2008) concludes that JPEG2000 is "a good current solution for our digital repositories", and ranks it above TIFF in terms of robustness[6].

B.4 Comparisons

All of the above formats are described as lossless, which means they retain image quality even when compressed. This means if a better suited format is released in the future, images in any of these formats can be converted to it with no loss of quality from the original.

This said, JPEG2000 has better compression than TIFF[6], and is emerging as a good general purpose replacement. I believe at the time of writing, PNG is the best choice for digital image preservation as the format is viewable and provides good compression, like JPEG2000, and despite still not having the market share of GIF, is well supported by applications and operating systems.

C Common Preservation Strategies

The following is a brief outline of the three prevailing methods of digital preservation.

C.1 Emulation

Emulation allows digital objects to run on a different platform from that which they were designed to run on, by emulating the original platform[7].

Whereas not nearly as popular as migration, emulation has benefits in the preservation of such things as software, where the dynamic and complex nature of the digital object is such that it relies on a platform too heavily to be migrated[11]. It can also be more cost effective over time, despite a higher upfront cost, due to less labour hours and continual work as present with migration[8].

C.2 Encapsulation

Encapsulation involves grouping together digital objects with the materials needed to provide access to those objects. It can function on a hardware and software level. Usually it would involve formal specifications of the technologies in use, as well as much metadata as possible to describe the digital objects[19].

C.3 Migration

Migration is the predominant preservation strategy, which involves the conversion of older formats to newer ones, or ones suitable for Long Term Preservation (LTP).

As migration is about fundamentally converting formats, data loss can be an issue with some aspects of the original not being able to be converted for a variety of reasons. This said, in a lot of cases enough information can be converted to make any small lapses ignorable.

As the main preservation strategy, there are many variants of migration, although all invariably involve format conversion. The main advantage of migration, apart from it often being the only viable method, is that it can make old formats usable in different ways on newer platforms, by its very nature.

C.4 Physical Preservation

All of the above digital preservation strategies carry an underlying assumption that the physical medium in which the data resides will remain intact. In reality, this assumption is naïve at best and negligent at worst. The physical medium in which digital data is stores is always susceptible to degradation and eventual corruption leading to data loss.[17, 10]

To combat this, a migration strategy called refreshment should be invoked at regular intervals. Refreshment involves combatting degradation by moving data from one physical medium to another on a bit level. This way degradation will not become an issue, and there will be no added problems with the mediums themselves becoming obsolete.[5]

Refreshment itself works in conjunction with the above strategies, and should be scene purely as a method of ensuring the integrity of the underlying physical medium.

D Code Listings

Below are the descriptions and code listings for all the custom tools I developed to aid preservation of file types in the scenarios. I am an ardent supporter of free and open software, and so would always advocate custom programming over proprietary system if a free and open source tool did not exist, and it was possible to create something of a high quality.

Custom tools also have the advance of being far more customisable than proprietary solutions, as well as giving the ability to mix and match them to fully automate a preservation strategy. In some cases, it is possible to mix the two to give the flexibility of a custom system with the stability and maturity of well established proprietary software.

D.1 Scenario 1

D.1.1 Image Converter

The image converter converts images between image formats via the aid of ImageMagick in the form of the Image::Magick Perl module. The program accepts a list of files and optionally a directory (via the -d flag) to be converted to a particular format, which is specified via the -t flag (if no -t flag is specified, it defaults to PNG).

Example: `perl image.pl -t tiff "logo Pur.bmp"`

Listing 1: ”image.pl”

```
#!/usr/bin/perl

use warnings;
use strict;

use Getopt::Long;
use Image::Magick;

my ($help, $dir);
my $type = 'png';
GetOptions('h' => \$help, 'd=s' => \$dir, 't=s' => \$type);

if ($help) {
  print "Usage: $0 [-t type] [-d directory] <files>\n";
  print "If -d is used, all files in the given directory will be converted\n";
  print "Files can be relative or absolute paths\n";
  print "Type can be any common type (png, tiff etc)\n";
  print "Example: $0 \"logo.bmp\" \"..\\test.jpg\"";
  exit;
}

die "Arguments expected, use the -h flag for help\n"
  unless $dir || @ARGV;
```

```
if ($dir) {
  # format directory
  $dir =~ s!\\!/!g;
  $dir =~ s!(\s)!\\$1!g;

  print convert_file($_) for (<$dir/*>);
}

print convert_file($_) for (@ARGV);

sub convert_file {
  my $file = shift;
  my ($out, $error);

  my $image = Image::Magick->new;

  # remove extension if one
  $out = ($file =~ /(.*)\..*/ && $1) || $file;

  $error = $image->Read($file);
  return $error if $error;

  $error = $image->Write(filename => "$out.$type");
  return $error if $error;

  return "Successfully converted '$file'\n";
}
```

D.1.2 Word Document Converter

The Word document converter converts files between any file readable by Microsoft Word to any output file type supported by Microsoft Word. The converter must be run on the Windows operating system, with Microsoft Word installed, as it interfaces with Microsoft Word via OLE.

The program accepts a list of files to be converted and optionally a directory (via the -d flag) to a particular format, which is specified via the -t flag (if no -t flag is specified, the format will be Microsoft Word's default save format).

The -t flag accepts numerical formats, which are listed at http://msdn.microsoft.com/en-us/library/bb238158.aspx, as well as a 'pdfa'[92] type, if using Word 2007 with the PDF add-in. If additional exporter formats are installed, these can also be used by their numerical values (a list of available exporter types and their correct numerical values can be shown by using 'list' as the value of the -t flag).

The program was tested with Microsoft Word 2007 and 2000, but should be compatible with Microsoft Word 97 and above. What input/output formats are available will depend on which version of Microsoft Word is installed.

Output files are saved in the same directory as the input file, with '-new' appended to

[92]http://en.wikipedia.org/wiki/PDF/A

the file name. The file extension of the output file will depend on the value of the -t flag.

The -v flag can be used to make the Word instance visible, which may be helpful for debugging purposes.

Example: `perl docs.pl -t 0 "report.doc"`

Listing 2: ”docs.pl”

```perl
#!/usr/bin/perl

use warnings;
use strict;

use Getopt::Long;

use Win32::OLE;
use Win32::OLE::Const 'Microsoft.Word';

my ($help, $dir, $visible);
my $type = 16;
GetOptions('h' => \$help, 'd=s' => \$dir, 't=s' => \$type, 'v' => \$visible);

if ($help) {
  print "Usage: $0 [-t type] [-v] [-d directory] <files>\n";
  print "If -d is used, all files in the given directory will be converted\n";
  print "If -v is used, the Word instance will be visible (good for debugging)\ ↩
       n";
  print "Files can be relative or absolute paths\n";
  print "Numerical type values listed at:s"
        . " http://msdn.microsoft.com/en-us/library/bb238158.aspx\n";
  print "Along with the numerical values, 'pdfa' is also supported"
        . " (Word 2007 and above), as well as any exporter types (use type list ↩
            to"
        . " see a list of these)\n";
  print "Note: converted files will have '-new' appended to the filename\n";
  print "Example: $0 \"report.doc\"";
  exit;
}

die "Arguments expected, use the -h flag for help\n"
  unless $dir || $type eq 'list' || @ARGV;

die "Invalid type, use the -h flag for more information"
  if ($type ne 'pdfa' && $type ne 'list' && $type !~ /-?\d+$/);

my $word = Win32::OLE->new('Word.Application')
  or die "Could not interface with Microsoft Word\n";

if ($type eq 'list') {
  my @converters = Win32::OLE::in($word->FileConverters());

  for my $converter (@converters) {
    my $num = defined $converter->{SaveFormat} ? $converter->{SaveFormat} : -1;

    print "Name: `" . $converter->{ClassName} . "', Save Format Number: $num\n"
      if ($num >= 0);
  }

  $word->Quit();
  exit;
}

if ($type eq 'pdfa' && $word->Application->Version() < 12) {
  print "Word 2007 or higher required for type pdfa\n";
  $word->Quit();
  exit;
}

$word->{'Visible'}        = $visible;
$word->{'DisplayAlerts'} = 0;

if ($dir) {
  # format directory
  $dir =~ s!\\!/!g;
  $dir =~ s!(\s)!\\$1!g;

  convert_file($_) for (<$dir/*>);
}

convert_file($_) for (@ARGV);

$word->Quit();

sub convert_file {
  s!/!\\!g; # change delimeter back to \

  my $file = $word->Documents->Open($_)
    or return print "Could not open file '$_'\n";

  my $name = $file->Name();
  my $new  = ($file->FullName() =~ /(?:(.*)\.)|(.*)/ && ($1 || $2) . '-new');

  if ($type eq 'pdfa') {
    $file->ExportAsFixedFormat({ OutputFileName => $new,
                                 ExportFormat   => 17,
                                 UseISO19005_1  => 1 });
    $file->close() && return print "Unable to convert file '$name'\n"
      if Win32::OLE->LastError();
  }
  else {
    $type += 0; # ensure integer (weird bug)

    $file->SaveAs({ FileName => $new, FileFormat => $type});
    $file->close() && return print "Unable to convert file '$name'\n"
      if Win32::OLE->LastError();
  }

  print "Successfully converted '$name'\n";
  $file->close();
}
```

D.1.3 Excel Document Converter

In a similar vein to the Word Document Converter, the Excel Document converter converts files between any file readable by Microsoft Excel, to any output format supported by Microsoft Excel. The converter must run on Windows, with Microsoft Excel installed.

The program accepts a list of files to be converted and optionally a directory (via the -d flag) to a particular format, which is specified via the -t flag (if no -t flag is specified, the format will be Microsoft Excel's default save format).

As with the document converter, the -t flags accepts a range of formats, listed at http://msdn.microsoft.com/en-us/library/bb241279.aspx, as well as ‘pdf’, if using Excel 2007 with the PDF add-in. If additional exporter formats are installed, they will also be available via their numerical ID, but unfortunately the program does not support listings supported IDs at this point.

The program was tested with Microsoft Excel 2007 and 2000, but should be compatible with Microsoft Word 97 and above. What input/output formats are available will depend on which version of Microsoft Excel is installed.

Output files are saved in the same directory as the input file, with ’-new’ appended to the file name. The file extension of the output file will depend on the value of the -t flag.

The -u flag when specified in conjunction with the -d flag, attempts to resolve all links in the converter directory. For example if ‘file1.wk1‘ links information ‘file2.wk1’, this link would be converted to point to ‘file2.xls’, or whatever file extension the converted file has.

The -v flag can be used to make the Excel instance visible, which may be helpful for debugging purposes.

Example: `perl sheets.pl -d "C:\excel\" -u`

Listing 3: ”sheet.pl”

```perl
#!/usr/bin/perl

use warnings;
use strict;

use Getopt::Long;

use Win32::OLE;
use Win32::OLE::Const 'Microsoft.Excel';

my ($help, $dir, $visible, $update_links);
my $type = 16;

GetOptions('h' => \$help, 'd=s' => \$dir, 't=s' => \$type,
           'v' => \$visible, 'u' => \$update_links);

if ($help) {
  print "Usage: $0 [-t type] [-v] [-u] [-d directory] <files>\n";
  print "If -d is used, all files in the given directory will be converted\n";
  print "If -v is used, the Excel instance will be visible"
        . " (good for debugging)\n";
  print "If -u is used, links will be updated if possible (requires -d)\n";
  print "Files can be relative or absolute paths\n";
  print "Numerical type values listed at:"
        . " http://msdn.microsoft.com/en-us/library/bb241279.aspx\n";
  print "Along with the numerical values, 'pdf' is also supported"
        . " (Excel 2007 and above), as well as any exporter types\n";
  print "Note: converted files will have '-new' appended to the filename\n";
  print "Example: $0 \"book.wk1\"";
  exit;
}
```

```
die "Arguments expected, use the -h flag for help\n"
  unless $dir || $type eq 'pdf' || @ARGV;

die "Invalid type, use the -h flag for more information"
  if ($type ne 'pdf' && $type ne 'list' && $type !~ /^-?\d+$/);

my $excel = Win32::OLE->new('Excel.Application')
  or die "Could not interface with Microsoft Excel\n";

$excel->{'Visible'}       = $visible;
$excel->{'DisplayAlerts'} = 0;

my $file_suffix = '-new';

if ($dir) {
  # format directory
  $dir =~ s!\\!/!g;
  $dir =~ s!(\s)!\\$1!g;

  convert_file($_) for (<$dir/*>);
}

convert_file($_) for (@ARGV);

$excel->Quit();

sub convert_file {
  s!/!\\!g; # change delimeter back to \

  my $file = $excel->Workbooks->Open({ FileName    => $_, ReadOnly => 1,
                                      UpdateLinks => $update_links ? 3 : 2 })
    or return print "Could not open file '$_'\n";

  my $name = $file->Name();
  my $new  = ($file->FullName() =~ /(?:(.*)\.)|(.*)/
              && ($1 || $2) . $file_suffix);

  my @links = Win32::OLE::in($file->LinkSources());

  # update any links (source files need converting first)
  if ($file and $update_links
      and my $filedir = ($file->FullName() =~ /(.*\\)/ && $1)) {
    for my $link (@links) {
      next unless $link;

      my $name = ($link =~ /.*\\([^\\]+)$/ && $1) || next;
      next unless (-e $filedir . $name && $name =~ /(?:(.*)\.)|(.*)/);

      my $prefix = ($1 || $2) . $file_suffix; # file without extension

      # format directory for glob
      my $source = $filedir . $prefix . '.*';
      $source =~ s!\\!/!g;
      $source =~ s!(\s)!\\$1!g;

      # link source file already converted
      if (my @source = glob $source) {
        my $ext = ($source[0] =~ /.*\.([^.]+)/ && $1);

        $file->ChangeLink($link, $filedir . $prefix . '.' . $ext);
        next;
      }

      # put us at end of queue so link files can be converted first
      push @ARGV, $_;
      $file->close();
      return;
    }
  }

  if ($type eq 'pdf') {
    $file->ExportAsFixedFormat({ FileName => $new, Type => 0 });
    $file->close() && return print "Unable to convert file '$name'\n"
      if Win32::OLE->LastError();
  }
  else {
    $type += 0; # ensure integer (weird bug)

    $file->SaveAs({ FileName => $new, FileFormat => $type});
    $file->close() && return print "Unable to convert file '$name'\n"
      if Win32::OLE->LastError();
  }

  print "Successfully converted '$name'\n";
  $file->close();
}
```

D.2 Scenario 3

D.2.1 CDS/ISIS Exporter

The ISIS Exporter loads a CDS/ISIS database via the Biblio::ISIS Perl module, and attempts to export the database to either XML or USMARC formats (USMARC requires the Marc::Record Perl module).

The XML exporter is very simplistic, and tried to ascertain the fields and elements format by running through the records and collecting as much information as possible. As a result, in sample exports, the XML database acquires more elements than the WinISIS export function for actual records, but omits elements that are seemingly defined but never used in any of the records. Furthermore, field names are not exported, and fields are listed via their tag numbers, as is the default with the WinISIS Export to XML facility.

The integrity of the XML exporter has not been thoroughly tested due to time constraints, although UNESCO have a utility called XML2Isis[93] which appears to be able to read the structure of the exported XML reasonably well.

The program accepts a path to the CDS/ISIS directory, and a type which is specified via the -t flag (if no -t flag is specified, it defaults to XML). The allowed types are: XML, MARC.

Example: `perl isis.pl -t XML "C:\isis\db1"`

Listing 4: "isis.pl"

[93]http://www.unesco.org/isis/files/winisis/windows/utilities/

```perl
#!/usr/bin/perl

use warnings;
use strict;

use Getopt::Long;
use Biblio::Isis;

my $help;
my $type = 'xml';
GetOptions('h' => \$help, 't=s' => \$type);

if ($help) {
  print "Usage: [-t type] $0 <ISIS DB directory>\n";
  print "Type is either xml or marc\n";
  exit;
}

die "Arguments expected, use the -h flag for help\n"
  unless @ARGV;

die "Invalid type, valid types are: xml, marc\n"
  if (lc $type ne 'xml' && lc $type ne 'marc');

# format directory
my $directory = $ARGV[0] . '/';
$directory =~ tr!\\!/!;
$directory =~ s!(\s)!\\$1!g;

# get database name from directory
my $dbname = ($directory =~ m!.*/([^/]+)/$! && uc $1) || 'DATABASE';

my $isis = new Biblio::Isis(isisdb => $directory, read_fdt => 1)
  or die "Could not load the Database/n";

# change format to work with direct open
$directory =~ s/\\(\s+)/$1/g;

if (lc $type eq 'marc') {
  use MARC::Record;

  open OUTPUT, '>', $directory . "$dbname.dat"
    or die "Could not create file '$dbname.dat': $!\n";

  for my $mfn (1..$isis->count) {
    my $record = $isis->to_hash($mfn) || next;
    my $marc   = MARC::Record->new();

    for my $key (sort { $a <=> $b } keys %$record) {
      next if $key eq '000';
      my $tag = (length $key < 3 ? '0' . $key : $key);

      my $field = MARC::Field->new($tag, '', '', '' => '');

      for my $value (@{ $record->{$key} }) {
        unless (ref $value) {
          $field->add_subfields('0', $value);
          next;
        }

        for my $subkey (keys %$value) {
          $field->add_subfields($subkey, $value->{$subkey});
        }
      }

      $marc->append_fields($field);
    }

    print OUTPUT $marc->as_usmarc();
  }
}

elsif (lc $type eq 'xml') {
  open OUTPUT, '+>', $directory . "$dbname.xml"
    or die "Could not create file '$dbname.xml': $!\n";

  my %tags;
  my @tags;

  print OUTPUT "<DATABASE_$dbname>\n";

  for my $mfn (1..$isis->count) {
    my $record = $isis->to_hash($mfn) || next;
    next unless (%$record);

    print OUTPUT "<RECORD>\n";

    for my $key (sort { $a <=> $b } keys %$record) {
      next if $key eq '000';
      my $field = $record->{$key};

      $tags{$key} ||= {};

      for my $value (@$field) {
        print OUTPUT "<Tag_$key>";

        if (ref $value eq 'HASH') {
          for my $subkey (keys %$value) {
            print OUTPUT "<Tag_${key}_$subkey>" . $value->{$subkey}
                         . "</Tag_${key}_$subkey>";
            $tags{$key}->{$subkey} = 1;
          }
        }
        elsif ($value) {
          print OUTPUT $value;
        }

        print OUTPUT "</Tag_$key>\n";
      }
    }

    print OUTPUT "</RECORD>\n";
  }

  print OUTPUT "</DATABASE_$dbname>\n";
  close OUTPUT;

  {
    # inplace editing of file
    local ($^I, @ARGV) = ('.tmp', ($directory . "$dbname.xml"));

    while (<>) {
      print && next if ($. > 1); # write original file back

      print "<?xml version=\"1.0\" encoding=\"ISO-8859-1\"?>\n";
      print "<!DOCTYPE DATABASE_${dbname}[\n";
      print "<!ELEMENT DATABASE_$dbname (RECORD*)>\n";

      # list of record fields
      # TODO: find out why ? is sometimes * and change appropiately
      push @tags, "Tag_$_?" for (sort { $a <=> $b } keys %tags);
      print "<!ELEMENT RECORD (" . join(', ', @tags) . ")>\n";

      # element list
      for my $key (sort { $a <=> $b } keys %tags) {
       if (!%{ $tags{$key} }) {
        print "<!ELEMENT Tag_$key (#PCDATA)*>\n";
       }
       else {
         print "<!ELEMENT Tag_$key (#PCDATA | ";
         print join(' | ', map { "Tag_${key}_$_" } keys %{ $tags{$key} });
         print ")*>\n";

         print "<!ELEMENT Tag_${key}_$_ (#PCDATA)>\n"
          for (keys %{ $tags{$key} });
       }
      }

      print "]>\n";
      print; # put back original first line
    }

    unlink $directory . "$dbname.xml.tmp"
      if (-e $directory . "$dbname.xml.tmp");
  }

  print "Database '$dbname' successfully exported to XML format";
}

close OUTPUT;
```

References

[1] Florida Digital Archive. Recommended data formats for preservation purposes. URL: http://www.fcla.edu/digitalArchive/pdfs/recFormats.pdf.

[2] Ian Barnes. Preservation of word processing documents. July 2006. URL: http://www.apsr.edu.au/publications/word_processing_preservation.pdf.

[3] Geoff Brown. Adding archival finding aids to the library catalogue: Simple crosswalk or data traffic jam? *Canadian Journal of Library and Information Practice and Research*, 2, 2007. URL: http://gir.uoguelph.ca/index.php/perj/article/viewFile/298/558.

[4] Michael Eadie. Preservation handbook: Bitmap (raster) images, Jul 2005. URL: http://ahds.ac.uk/preservation/Bitmap-preservation-handbook_d6.pdf.

[5] Consultative Committee for Space Data Systems (CCSDS). Recommendation for space data systems standards, Jan 2002. URL: http://public.ccsds.org/publications/archive/650x0b1.pdf.

[6] Paolo Buonora Franco Liberati. A study on jpeg 2000 file robustness. *D-Lib Magazine*, 14(7/8), Aug 2008. URL: http://www.dlib.org/dlib/july08/buonora/07buonora.html.

[7] Stewart Granger. Emulation as a digital preservation strategy. *D-Lib Magazine*, 6(10), Oct 2000. URL: http://www.dlib.org/dlib/october00/granger/10granger.html.

[8] Stewart Granger. Digital preservation & emulation: from theory to practice, Sep 2001. URL: http://www.leeds.ac.uk/cedars/pubconf/papers/ichim01SG.html.

[9] Adobe Systems Incorporated. Pdf/a - archiving standard faq, 2003. URL: http://www.adobe.com/government/pdfs/faq_archivingstandard_pdf_a.pdf.

[10] Julian Jackson. Digital longevity: the lifespan of digital files. URL: http://www.dpconline.org/graphics/events/digitallongevity.html.

[11] Koninklijke Bibliotheek National library of the Netherlands. What is emulation? URL: http://www.kb.nl/hrd/dd/dd_projecten/projecten_emulatiewatis-en.html.

[12] Microsoft. Open specification promise, 2008. URL: http://www.microsoft.com/interop/osp/default.mspx.

[13] Library of Congress (LOC). Ead application guidelines: Appendix b, 1999. URL: http://www.loc.gov/ead/ag/agappb.html#sec4.

[14] Library of Congress (LOC). Sustainability of digital formats: Png, Oct 2006. URL: http://www.digitalpreservation.gov/formats/fdd/fdd000153.shtml.

[15] Library of Congress (LOC). Sustainability of digital formats: Tiff, Oct 2006. URL: http://www.digitalpreservation.gov/formats/fdd/fdd000022.shtml.

[16] Library of Congress (LOC). Sustainability of digital formats: Pdf/a-1, Feb 2007. URL: http://www.digitalpreservation.gov/formats/fdd/fdd000125.shtml.

[17] Optical Storage Technological Association (OSTA). Understanding cd-r and cd-rw disc longevity. URL: http://www.osta.org/technology/cdqa13.htm.

[18] Roger Reeves. Pdf/a - a new standard for long-term archiving, 2008. URL: http://www.pdfa.org/doku.php?id=pdfa:en:pdfa_whitepaper.

[19] Preserving Access to Digital Information (PADI). Encapsulation. URL: http://www.nla.gov.au/padi/topics/20.html.