

Penn Libraries

Scholarship at Penn Libraries

University of Pennsylvania

Year 2001

Archiving and Preserving PDF Files

John Mark Ockerbloom

University of Pennsylvania, ockerblo@pobox.upenn.edu

Published in *RLG DigiNews*, Volume 5, Issue 1, February 15, 2001.
Publisher URL: <http://digitalarchive.oclc.org/>

This paper is posted at ScholarlyCommons.
http://repository.upenn.edu/library_papers/49

Archiving and Preserving PDF files

John Mark Ockerbloom <ockerblo@pobox.upenn.edu>

Digital Library Architect and Planner, University of Pennsylvania,

Since its release in mid 1993, Adobe Portable Document Format (PDF) has become a widely used standard for electronic document distribution worldwide in many institutional settings. Much of its popularity comes from its ability to faithfully encode both the text and the visual appearance of source documents, preserving their fonts, formatting, colors, and graphics. PDF files can be viewed, navigated, and printed with a free Adobe Acrobat Reader, available on all major computing platforms. PDF has many applications and is commonly used to publish government, public, and academic documents. Many of the electronic journals and other digital resources acquired by libraries are published in PDF format.

As libraries grow more dependent on electronic resources, they need to consider how they can preserve these resources for the long term. Many libraries retain back runs of print journals that are over 100 years old, and which are still consulted by researchers. No digital technology has lasted nearly that long, and many data formats have already become obsolete and not easily readable in a much shorter time period. This document discusses ways that libraries can plan for the preservation of electronic journals and other digital resources in PDF format. After a brief discussion of the file specifications and the future plans for PDF, the article focuses on issues related to preservation of PDF files.

Description of PDF

PDF is a platform-independent document format developed by Adobe as a follow-up to its Postscript language (which is now used almost universally in graphics-capable printers). While Postscript was designed specifically for printing, PDF is meant to support on-line, secure, interactive use, as well as printing.

PDF combines attributes of structured text formats and traditional image formats, so that a document can be used both in terms of its text and in terms of its "look" on-screen or on the page. This allows PDF readers not only to display and print documents, but also to carry out text-oriented operations such as searching for text strings, or cutting-and-pasting. (However, as we will see below, it's also possible for a PDF document to consist only of uninterpreted page images—that is, pictures of the text but no machine-readable encoding of the text itself.)

Like Postscript, PDF is, at its heart, a page description language. However, unlike Postscript, it is more of a declarative than a procedural language (which is to say, the data isn't wrapped up in program code as it is in Postscript). This makes PDF easier to manipulate and analyze than Postscript.

As with Postscript, the specification of PDF is openly and freely published (1), but Adobe controls its development. Three revisions of PDF have been published to date, with PDF 1.3 the latest version for which specifications are published. (Adobe Illustrator 9, released in 2000, outputs files in PDF 1.4, a version whose specification has not yet been published.) So far, all of the revisions for which specifications have been published are backward-compatible (that is, if you can understand PDF 1.3, you can also understand PDF 1.1 and 1.2.) Because the basic standard is public, Adobe does not have a monopoly on PDF tools, and third parties have developed tools that also work with PDF. However, because the standard is complex, and changes from time to time, much of the support for the format, as well as the tools that are considered to set the standard for PDF use, comes from Adobe.

A typical PDF document consists of a sequence of pages, each of which includes data that indicates what should be "drawn" on a particular page. This data includes text, font specifications, margins, layout, graphical elements, and background and text colors. PDF includes metadata on the document, specifying things like the size of pages, the numbering used for pages, and a structured table of contents. Extra add-ons, such as hyperlinks, forms, and even pieces of scripting code in Java or Postscript, that control dynamic displays or the user interface, can also be included.

Not all PDF documents encode the text of a document directly. It is possible to embed images of a page (or of anything else) in PDF, so that a picture of text is encoded but not the text itself. Some scanners, for instance, can output this "uninterpreted" form of PDF. Such uninterpreted PDF files tend to be much larger than PDF files that encode text directly as characters, and text-oriented operations such as searching and textual copying and pasting are not supported by these files. Still, some libraries have produced these large, uninterpreted PDF files to their users as a way of quickly and inexpensively distributing scans of print materials. For applications like electronic reserves, where scanned copies

are typically not kept beyond a semester or two, this form of PDF can be acceptable. Some programs also exist that will attempt to recognize whatever text they can in a scanned PDF document, and replace the images of this text with a briefer encoding of the characters and fonts. Such programs tend to produce "hybrid" PDF files where encoded text strings in appropriate fonts are positioned inside page images, and any unrecognized characters are kept on the page in the form of images. While such PDF files can be smaller than those that consist only of uninterpreted page images, they also often have only partial, and choppy, encoding of their underlying text, because recognition programs typically cannot recognize 100% of the text in a typical page scan. Moreover, misrecognized characters may even introduce errors into the "hybrid" document.

PDF was originally designed for 8-bit character sets (which can support at most 256 characters). Some limited support for Unicode also exists, but the format has not yet taken full advantage of Unicode's capabilities. One consequence of this design is that some multilingual documents may be represented by switching between different, sometimes customized, 256-character sets (each with its own font). Alternatively, some "exotic" characters may be represented as small images, rather than as text.

Unlike XML, PDF was not originally designed for arbitrary structural annotations, although the last revision of PDF includes some facilities for user-defined structural markup. PDF is a less efficient format for text analysis than purely text-based formats like XML or HTML, and readers have noted that searching for words within a PDF viewer can be noticeably slower than searching for words in an HTML viewer. The central role that appearance plays in PDF is both a strength and a weakness; it allows publishers to specify exactly how they want a document to look, but it also makes it difficult for users to reformat or repaginate the document if that would be better for them.

PDF includes facilities for encrypting content, or directing that a document not be printable. It is possible to use encryption so that a document cannot be decrypted without a password. However, once the document **is** decrypted, it is possible, in theory, to make an unlocked copy that can be freely copied, read, or printed. Even if it is technically possible to do so, though, legal and practical restrictions to doing so may still exist. (2)

The future of PDF

As data formats go, PDF is particularly likely to be supported for a long time, and to spawn migration paths. This has been the case with Adobe's Postscript, which is now an industry standard, and for which PDF itself can be seen as the next step in migration. Adobe has openly published the PDF standard, and freely distributes readers for all major operating systems. Adobe is highly invested in the success of PDF, but even if Adobe fails or abruptly changes course, a community of third-party tools for handling PDF has started to emerge. PDF is used widely by many well-funded bodies (including the US government, which is now using PDF as its standard way of distributing government publications) so there should be widespread support for using and migrating PDF, should Adobe fail to provide adequate support for the format.

Even so, it is likely that PDF will one day be superseded by another format. It may be a successor format (as PDF is to Postscript), or it may be a completely different format that users prefer over PDF. Hence, it is necessary to have migration strategies planned for PDF.

Migrating PDF

Currently, there is no other format that can encompass all of the features of PDF in a form that standard tools can interpret. However, it is possible to easily migrate two important aspects of PDF documents: the page images and (for many types of PDF files) the text. These aspects may be sufficient for the types of preservation that libraries require.

Any program that displays or prints PDF files in effect converts PDF pages into static page images. Most page image formats are pixel-oriented, where pictures are encoded using a grid with each grid cell (known as a pixel) associated with a particular color (or with no added color). Some image formats, including PDF and Postscript, are stroke-oriented, encoded with instructions about lines to plot, regions to color, and text to place. (They can also include instructions to embed pixel-oriented images at specified locations.) Pixel-oriented formats have a well-understood mathematical representation, making them easy to migrate. Stroke-oriented image formats are not so standardized. They can encode information about the structure of a picture, such as the lines that are drawn in it, that can only be approximated to a certain resolution in purely pixel-oriented formats. On the other hand, since pixel-oriented formats can typically accommodate arbitrarily fine resolutions, an image converted from a stroke-oriented format to a pixel-oriented format at fine enough resolution can be visually indistinguishable from its original.

Third-party software exists to turn PDF both into Postscript, and into pixel-oriented image formats. Converting from PDF to most image formats loses direct encoding of the text (making searching, or cutting-and-pasting, infeasible unless the text is OCR'd.) Structural information and other PDF extras would also be lost in such a conversion.

PDF can also be converted into text, at least when the text (rather than pictures of the text) has been directly encoded in PDF files. Formatting and layout information would be lost in a straight text conversion, and the text conversion of certain pages with complex layout may not always come out in "logical" reading order, without human intervention. For example, if a PDF encoding of a two-column page first renders the left column and then the right, a simple text extraction from the PDF will yield a readable text transcription. On the other hand, if the PDF encoding goes back and forth between the two columns (as is possible in PDF, where drawing operations on a page can be defined in any order), or renders the right column before the left column, programs may have a much more difficult time reconstructing the text in the order it was meant to be read. Different PDF-generating programs may be more or less accommodating about rendering PDF text in a logical reading order. A straight text conversion would also lose structural information and other PDF extras, as well non-textual items such as pictures, diagrams, and possibly equations. However, it is possible to write conversions to HTML or XML that would include a "table of contents" header that incorporated the structure of the PDF document, and that would preserve certain types of PDF hyperlinks. Text in non-European languages may also be difficult to migrate correctly, because of the previously-described limitations of PDF's support for Unicode, unless a standard set of well-defined fonts, with widely available specifications, is used to represent non-European characters.

Although it is possible to migrate the appearance of PDF files, or the text, migrating *both* of them into a single document would be more difficult. The only other formats commonly used to specify both the text and the appearance of documents are word-processor formats, and those tend to be proprietary, system-dependent, and more ephemeral than PDF itself is likely to be. Converting to these formats may also lead to subtly different appearances, depending on the implementation of the word processing program. RTF may be the best target for word-processor-oriented conversions at this time, though it is far from a perfect choice.

Even if it is infeasible to migrate both the text and the look of PDF into a single document, it may still be possible to migrate PDF documents into a presentation that allows one to switch between text-oriented and image-oriented views of the same document. (Some on-line book projects now support this type of view for their titles. For example, some documents from the Library of Congress' American Memory project include both page images and transcriptions, with hyperlinks placed between corresponding pages of each version.)

In theory, it is possible to convert all of the information in a PDF file into an XML format with appropriate tags and attributes. However, at present there is no standard XML format used for representing PDF-like documents. Unless such a format were standardized, and then supported with software, converting to XML might be largely an academic exercise, since no software currently exists to interpret such a format, and software developed especially to interpret the new XML format could probably just as easily interpret the information in its original PDF form. On the other hand, an XML-based form of PDF might be easier for existing XML-based search and analysis software to process.

Interactive features of PDF would be harder to migrate, and Postscript, JavaScript, or other scripting snippets may be impractical to migrate on a large scale, because it is much harder to migrate program code than data. Libraries may want to avoid committing to preserving these "extra" features of PDF files, except in carefully limited cases.

Archiving PDF-Format Works

Certain preservation techniques, such as integrity checks and backups, are necessary to preserve any type of digital information. Beyond those basic requirements, institutions need three things to reliably preserve PDF-format documents so that they can be reused in perpetuity:

- First, a specification of exactly what an institution is committing to preserve.
 - Easiest to commit to:

Static images of the PDF document pages (at worst, with some variation in the fonts that would not affect their sizing or layout)

- A bit harder, but still important:

Static images with exact replication of original fonts
The text encoded in the PDF document
The table-of-contents structure encoded in the PDF document

- Much more difficult:

Dynamic, script-dependent, or other auxiliary attributes of a PDF document. Specific attributes, such as normal Web hyperlinks, may be preservable if a commitment is made up-front to preserve them.

- Second, quality control over the PDF documents acquired, to ensure that the format specifications (e.g., type of encoding, use of embedded scripting and encryption, etc.) match the archiving institution's guidelines.

Institutions that create PDF files meant for archiving, or who receive them from publishers, need to employ policies that ensure that the PDF they acquire is easy to migrate. In particular:

- For PDF files where the text encoding should be preserved, all text should be encoded as characters (not pictures, and not as "hybrid" OCR'd files where recognized character codes are interspersed with uninterpreted character glyphs), using standard character sets. Unicode is preferred for multilingual documents or documents not in European languages. The pages and text encodings should appear in the document in logical reading order.
- Embedded scripting should be avoided, where possible, and especially when it affects the static appearance of the page.
- Documents should be received in unencrypted form, or with the encryption undone immediately on receipt.
- Documents should use a version of PDF that is publicly documented, and preferably which has an ample set of third-party tools available.

In addition to establishing standards for acquired PDF files, institutions may also find it useful to have tools to automatically check incoming PDF files and warn of any unexpected PDF constructs appearing in them. Such automated checks could be used to automatically flag malformed PDF files, or files that failed to meet institutional encoding standards, while avoiding expensive manual checks on all incoming files. However, writing such tools, or buying suitable tools developed by third parties, may also be nontrivial. Institutions with a common interest in preserving PDF may wish to collaborate on the development of such tools.

- Third, tools and procedures for migrating PDF into other formats.

Some software tools exist for extracting text and images from PDF files, and can be used for migration. Especially useful are tools that can be invoked automatically without human interaction. Examples in the free software world include Ghostscript for image migration, and Pstotext or Prescript for text migration. Some commercial conversion tools exist as well. Further research and development may be necessary to ensure that these conversion tools will correctly migrate all of the PDF files of interest to institutions. At present, image-based conversion appears not difficult to complete and verify, but automatic extraction of text, due to the nature of text layout, may not always be perfect. One possible technique for increasing the reliability of automatically extracted text would be similar to the "double-keying" principle for data entry: run two independently developed text extractors against the same PDF file, and flag points where the extractors yield different text in a different order. It remains to be seen how useful this technique would be in practice.

It may be useful to run migration routines on new PDF files as soon as they are acquired, even if migrated files are not yet required. After the migration is attempted, checks could be run to see if any problems were detected in the migrated files. Institutions do not necessarily have to keep migrated copies of the files as long as they have verified that migration works, and they preserve the original PDF files and the programs that will perform the migration (along with any necessary operating environment for these programs).

If one were being especially careful, though, it may be advisable to save alternative forms immediately, not just in digital form, but also in print form. While such practices would further decrease the probability of total loss of PDF documents, they also may substantially increase the per-volume cost of preserving these documents. It is probably sufficient to have trusted methods of backup, integrity checking, mirroring, and migrating.

In summary, it is reasonable, given careful techniques such as those described above, for institutions to collect documents in PDF format with the expectation that they can be archived and preserved indefinitely, even as computer technology and standards advance. Some further experimentation and development is necessary for establishing reliable techniques for migration. We plan to engage in such experimentation and development in connection with a journal archiving project at the University of Pennsylvania Library. We would be very interested in sharing experiences and expertise with partner institutions.

Notes:

1. The most recently published PDF specification at this writing, for PDF 1.3, can be found at <http://www.pdfzone.com/resources/technicalinfo.html>
2. It is technically possible to undo any purely software-based copy-protection scheme, given sufficient resources to reverse-engineer and write tools for unlocking the copy-protected format. Undoing copy-protection will not in itself make an encrypted file interpretable, but if a file can be decrypted in software for display, it can also be decrypted to bypass its original access restrictions. Such decryption is unlikely to be supported by Adobe's standard tools, and the legal restrictions imposed by the Digital Millennium Copyright Act, or the proposed Uniform Computer Information Transactions Act (UCITA), may legally proscribe such actions. At the same time, doing so may sometimes be the only way of keeping a PDF file readable if changes in the computing environment make it infeasible to read in its original form. Librarians concerned about this issue may need to work to ensure that the law, and their contracts with electronic data providers, does not prohibit them from preserving their electronic data in this way.