Experience with the use of Acrobat in the CAJUN publishing project[†]

David F. Brailsford

Electronic Publishing Research Group
Department of Computer Science
University of Nottingham
NOTTINGHAM NG7 2RD
E-mail dfb@cs.nott.ac.uk

ABSTRACT

Adobe's Acrobat software, released in June 1993, is based around a new Portable Document Format (PDF) which offers the possibility of being able to view and exchange electronic documents, independent of the originating software, across a wide variety of supported hardware platforms (PC, Macintosh, Sun UNIX etc.).

The fact that Acrobat's imageable objects are rendered with full use of Level 2 PostScript means that the most demanding requirements can be met in terms of high-quality typography and device-independent colour. These qualities will be very desirable components in future multimedia and hypermedia systems. The current capabilities of Acrobat and PDF are described; in particular the presence of hypertext links, bookmarks, and 'yellow sticker' annotations (in release 1.0) together with article threads and multi-media 'plug-ins' in version 2.0,

This article also describes the CAJUN project (**C**D-ROM **A**crobat **J**ournals **U**sing **N**etworks) which has been investigating the automated placement of PDF hypertextual features from various front-end text processing systems. CAJUN has also been experimenting with the dissemination of PDF over e-mail, via World Wide Web and on CD-ROM.

1. Introduction

There is one question that is always certain to provoke furious debate in the field of electronic documents: "Does page fidelity matter?". Many research workers in hypermedia are keen to investigate electronic documents, in all their generality, free from the constraints of a page-based format that is seen to be of relevance only for print on paper. Indeed, those systems with few, if any, connections to a 'page metaphor', have been precisely the ones able to concentrate on the *structure* of a document as the key to its electronic dissemination and to the placement of hypertext links. An example of this sort of approach can be found in the Dynamic Book Company's *Dynatext* system [1], which uses SGML tags as structural markup for its electronic documents. A recent, and very successful, system which has also adopted this approach is the Internet-based World Wide Web (WWW) [2] which employs an SGML-like markup, called HTML, to allow documents to specify hypertext links and thus to 'point' at one another over the Internet. Viewers for all of these systems tend to concentrate more heavily on the link structure than on any notion of page fidelity with respect to some printed version — a WWW document will quite happily re-flow its text to different window shapes on different viewing hardware.

And yet, despite all of this, there are still powerful pressures from the electronic publishing industry for a page-based approach to electronic documents. Two factors seem to be important here: firstly there is a

† FINAL DRAFT of paper which appeared in the Proceedings of the ACM European Conference on Hypertext, Edinburgh, 1994 pages 228−232

feeling that nervous first-time users will welcome familiar-looking pages on their screens; secondly any archive material that has been acquired by scanning, or by reprocessing laser-printer output coding (e.g. PostScript), is very likely to be in page form already.

So what is the situation for page-based electronic documents, if we stipulate at the outset that we want to have a more abstract representation (and better imaging quality) than simple bitmap images of scanned pages? Until recently, the market-leader in this area has been Interleaf's *WorldView* system which uses Printerleaf, a proprietary internal format, but which can accept material in a range of graphic formats including CGM (Computer Graphics Metafile). *WorldView* has hypertext features and cheap viewer software, but the conversion software to enable current documents to be converted for *WorldView* is considerably more expensive. There are many other more recent competitors: the WordPerfect Corporation has recently announced a format called *Envoy*, the No Hands Corporation has released *Common Ground* — which uses Macintosh PICT files — while the *Replica* system from Farallon Computing uses GDI and Quickdraw primitives, from the PC and Macintosh environments respectively. These last two approaches have the advantage that the use of system primitives from the Macintosh and the PC makes it easy to capitalise on the windowing display software already present in the operating system. On the other hand it also means that they are tied to specific hardware platforms in the first instance. All of these systems offer hypertextual features, approximate page fidelity and varying qualities of screen-font rendering.

Given the established status of PostScript Level 2 as a *de facto* standard for high-quality imaging, it is clear that Adobe Systems Inc are in a powerful position to develop their technology further and to create a page-based electronic document format; PostScript's Type 1 fonts are truly resolution-independent and Level 2 PostScript also provides a device-independent colour model. The fact that hardware manufacturers such as NeXt and SUN have already licensed Display PostScript as an imaging model for use on their high-end workstations, sets the scene for the use of this technology in 'electronic documents'. However, Display PostScript itself is not enough. A significant further development is needed, with lower file-size overheads than ever before, and with 'added value' in terms of hypertext facilities, text search and so on. It would also need to be robust with respect to transmission over networks and capable of being viewed with software available on *all* of the popular computer platforms (IBM PC, Macintosh, UNIX systems etc.).

2. What is Acrobat?

In July 1993 Adobe Systems released a product called Acrobat, which is based on a new Portable Document Format (PDF), developed by Adobe [3]. PDF remains close to Level 2 PostScript but has a range of file-compression options available. It also has a set of facilities for hypertext links, 'thumbnail' icons of pages, chapter outlines and page annotations. The links were initially intra-document only in release 1.0 but the recent arrival of release 2.0 has brought with it inter-document links and an Application Programmer's Interface (API) which allows Acrobat's imaging technology to be interfaced into other multi-media and networked systems.

There is an option in PDF for LZW compression on text, which gives a compression factor of about 2:1, but bigger gains are obtainable on images, particularly those in colour, where JPEG compression can achieve some spectacular reductions in the region of 10:1. An optional feature of the screen display in the Acrobat viewer software is to display 'thumbnails' of the document pages. These are miniaturised JPEG-compressed bitmaps of the document's pages — each one is unique and there is enough detail to be able to recognise a page from the layout of its thumbnail. The thumbnails can be optionally displayed down the left-hand side of the screen and they greatly facilitate fast browsing and random access: one can jump from the page displayed on the screen to any distant page by clicking on the appropriate thumbnail for the destination page.

In addition to links — which are of the classic hypertextual 'goto' variety — PDF also has a 'chapter outlines' (or 'bookmark') feature which enables one to create a hypertextual Table of Contents for a book or an article. All sections and subsections can be entered into this hierarchical outline, each entry of which is a link to some predefined view of a fixed page within the document.

Two other hypertextual features of PDF are page annotations — an electronic version of the "yellow stickers" that are commonly attached to paper-based memoranda — and article threads. Article threads, which have been introduced in release 2.0, are simply linked lists of various views of a document's pages. They enable 'guided tours' to be set up, which may omit some of the material in the full document or may ensure that the reading of a multi- column document flows smoothly from the bottom of one column to the top of the next.

PDF has a set of markers, called pdfmarks, for all of these hyperfacilities. They can either be added to the PostScript after it has been produced or can be passed down from ‘front-end’ text-processing packages into the final PostScript. The pdfmarks are interpreted during the distillation process (see section 2.1) and converted into the corresponding PDF hypertext objects; a revised prologue in the PostScript file ensures that the pdfmarks are ignored if the file is sent to a PostScript laser printer.

2.1. Acrobat viewers and the Distiller

Acrobat is currently supported on IBM-compatible PC (MS-DOS and MS-Windows) on the Macintosh and on SUN UNIX platforms. Two versions of Acrobat viewer software are available. The *Reader* provides facilities for browsing existing PDF documents and for printing them out. If the document has already been enhanced with hypertext links these can be followed but they cannot be altered in any way. By contrast the *Exchange* version of the viewer permits a degree of editing with respect to the the various ‘hyperfeatures’ and it also allows complete pages from other PDF documents to be interleaved with those already present. In what follows we shall use the general phrase ‘viewer’ to mean either of the Reader or Exchange versions. Note that PDF does not, at present, allow the underlying formatted text to be altered in any way — it is a fixed-page format.

A program called the Distiller converts PostScript into PDF, carefully transforming any pdfmarks into the corresponding PDF hyperfeatures as distillation proceeds. However, if the text-processing software in use cannot produce pdfmarks directly then they can be inserted into the PostScript output file afterwards or added ‘by hand’ using the Exchange viewer. Needless to say, the first of these possibilities is the most desirable by far; it ensures that the pdfmarks are placed at the earliest possible stage, as the document is being prepared and while its innate structure is still known.

Figure 1 shows the process of producing PDF by distilling PostScript and it also shows the stages in the process where pdfmarks can be inserted. Another route to PDF, for simple documents, is via a ‘PDFwriter’ module which is, in essence, another printer interface. This enables PDF to be produced *directly* from any software (e.g. MS-Word) which uses the underlying graphic primitives of the two popular platforms (i.e. Quickdraw on Macintoshes and GDDI on MS-Windows PC hardware).

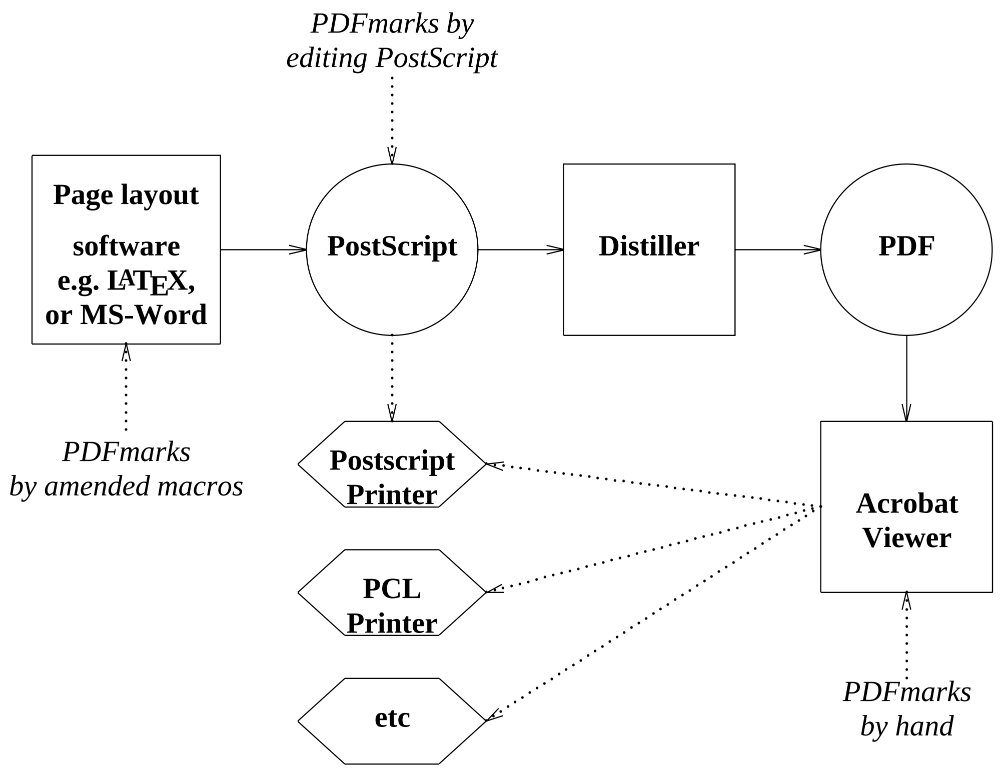
PDFmarks by editing PostScript
Page layout software e.g. LATEX, or MS-Word
PostScript
Distiller
PDF
PDFmarks by amended macros
Postscript Printer
PCL Printer
etc
Acrobat Viewer
PDFmarks by hand

Figure 1: From source code to PDF via PostScript

3. The CAJUN project

The John Wiley Ltd. journal *Electronic Publishing — Origination, Dissemination and Design* (*EP-odd*) has been published 4 times a year since 1988. Its title alone put great pressure on editors and publishers alike to produce an 'electronic' form as early as possible and yet the very range of topics covered[1] made it difficult to find a suitable format. All scanned-page, bitmap formats were rejected out of hand and systems involving hypertextual structured markup — but with no page fidelity — also had to be rejected. One of the key considerations was that the journal publishes material on topics such as type design and grey-scale fonts, which require high-quality screen rendering from any 'electronic' format that is adopted. Adobe Acrobat, when it was announced, seemed capable of meeting these needs. Device-independent colour, and resolution independence, are an integral part of Acrobat's PostScript Level 2 foundations; moreover its Super ATM font-rendering mechanisms made it the *only* electronic document technology capable of rendering type design well enough for details of stems and serifs to be clearly discernible on the workstation screen. The page-based nature of Acrobat was not seen as a drawback — at least initially — for a journal which had appeared in conventional paginated form for several years, but it was slightly disappointing to find that Acrobat's hypertextual features, though just about adequate to our needs in terms of appearance and functionality, lacked any kind of abstract qualities; their concrete nature requires that link anchors be place absolutely in terms of bounding boxes of source and destination areas on fixed pages. The automated placement of these links has called for multi-pass delayed-binding link-edit techniques (see section 3.1)

Having decided to fix on Acrobat for our experiments, funding was obtained from John Wiley Ltd and Chapman & Hall for a two-year project to be carried out in the Electronic Publishing Group at Nottingham. This project is called CAJUN (**C**D-ROM **A**CROBAT™ **J**ournals **U**sing **N**etworks). In addition to the Wiley *EP-odd* journal, Chapman & Hall have contributed material from their journal *Optical and Quantum Electronics* (OQE) and have committed themselves to producing a new journal, *Collaborative Computing* (CC), using PostScript and PDF formats.

3.1. Automating the PDF hyperlinks

The principal source of experimental material for the CAJUN project has been the archive of source text, and final PostScript, for everything published in *EP-odd* since its launch in 1988. This gave us the option of using the Distiller on the PostScript archive or of regenerating the PostScript for distillation (complete with inserted pdfmarks for the links) from the marked-up source text, using new, enhanced, macro sets.

The latter strategy has been adopted for volume 5 onwards of *EP-odd*. The CAJUN project has adapted the existing *troff* and LaTeX macros for *EP-odd* so that, for example, a call-out of a reference can automatically pass down a pdfmark for a PDF 'hot-link', into the final PostScript. After a second-pass the citation can be automatically linked to the References page. In a similar way the LaTeX `\label` and `\ref` mechanisms could be used to ensure that phrases such as " … see Figure 1" could be linked, either forwards or backwards, to zoom in on Figure 1, on the appropriate page, once the button for the link has been pressed. The innate structure of a PDF file is a hierarchically-structured set of objects attached to a page-based tree. Each object is identified by a unique number and a cross-reference table keeps track of where the objects are in byte-offset terms. Link objects are owned (i.e. 'pointed at') by page objects and the indirect referencing via object number means that the links can be gathered together anywhere in the PDF file. Unfortunately this flexibility for locating the links anywhere in the PDF file was not passed on to users in version 1.0 of the pdfmark for links. The format of this procedure call is:

```
[/Rect [llx lly urx ury] /Border [bx by c] /Page pagenum
 /View [x y zoom] /LNK pdfmark
```

Notice that the `pagenum` field refers to the destination page of the link. It was thought quite adequate that the source page should be denoted implicitly, by requiring that the link pdfmark be emitted on

[1] "Everything from serifs to SGML; everything from hyphenation to hypertext"

that page. This restriction, if related to classical compilation of programs for virtual memory architectures, is rather like insisting that certain machine code instructions cannot be flexibly relocated — they must reside at fixed addresses inside fixed page numbers. Filling in the missing arguments on absolutely-positioned instructions of this kind is extremely tricky, though reference [4] shows how this can be accomplished, without having to remember the position of the pdfmark instruction itself. A hybrid scheme was adopted which uses PostScript position coordinates at the source of a forward link and LaTeX or *troff* coordiantes at the destination. Fortunately, a new version of the link pdfmark has now been released, with an extra `/SrcPg` key. This enables a more elegant processing technique to be employed. During initial text-processing by LaTeX, *troff*, or any other suitably configured package, all sources and destinations for links are made to post down commands, into the PostScript output stream, which cause the current page position to be stored in variables in PostScript's global dictionary space. A suitable naming scheme allows sources and destinations to be matched, bounding boxes for buttons to be calculated, and all necessary values to be prepared on the stack for insertion into the link pdfmarks, which are now all gathered together at the end of the PostScript file. As far as the front-end software is concerned no extra passes are needed. The processing of the pdfmarks, by the PostScript interpreter at the heart of the Distiller, invokes a standard set of procedures in a revised PostScript prologue. In effect, the extra processing pass is now carried out by the Distiller itself, using delayed binding of coordinates into the pdfmark calls. An additional advantage of this approach is that it becomes independent of the particular front-end text processor in use. Any package can be used so long as it can be persuaded to emit the appropriate link-processing procedure calls, at the right time, into the PostScript output stream.

In April 1994 the CAJUN project released a CD-ROM containing the first 6 volumes of the EP-odd journal in electronic form, together with Acrobat viewers (release 1.0 version) for the Macintosh, MS-DOS and MS-Windows operating systems. The majority of the papers in this CD-ROM had their links and bookmarks created by the techniques just described.

Turning to network dissemination, we have recently set up an experimental service on a journal server at Nottingham which is available over the Internet. Seven papers are available in PDF form — one of them is [4] which describes the automated hyperlink techniques developed in the CAJUN project so far. The details for accessing this online material are as follows:

```
Gopher: quill.cs.nott.ac.uk
        (128.243.23.12)

FTP: ftp.cs.nott.ac.uk
       Files are in /ep/pub/pdf

WWW: http://quill.cs.nott.ac.uk
```

We have also conducted several experiments in transmitting PDF files over e-mail. The fact that PDF will accept either CR, LF, or CR/LF as denoting 'end of line', coupled with the ability to map all PDF data onto an 85-character subset of ASCII, means that it is a reasonably robust format for e-mail — most material gets through unscathed. Nevertheless, various problems remain, which are by no means unique to PDF: some mailer software can be extremely puritanical about truncating long messages and any mail travelling via Bitnet, with its preponderance of IBM mainframes, is liable to suffer loss of vital information in ASCII / EBCDIC / ASCII code conversions. In the long term, the adoption of MIME protocols (which include an ASCII-64 mapping option) by all mailer software, will ensure that any data, including video and sound clips, should be able to survive the transition even through the most hostile of gateways and networks.

4. Conclusions

Acrobat seems set to become a *de facto* standard in the world of page-based electronic documents. Its very closeness to PostScript, and the modular design of the API in release 2.0, means that aspects of Acrobat, such as font rendering and device-independent colour, can be used as components in a variety of electronic document viewers. The ability to create 'plug in' modules for the API means that platform-dependent processes for video and sound can be initiated as link actions.

Acrobat marks a turning point for everyone working in the field of electronic documents and particularly for projects where page fidelity is important with respect to some past, or future, printed version. This is not to say that paper documents will go away — it's doubtful if this will ever occur. But increasingly a paper document will come to be seen as a useful two-dimensional snapshot, at a given point in time, of a much richer electronic document. Adobe's Acrobat, is unique in being able to make the transition to an electronic format without losing any of the attributes of the high-quality printed version.

It is likely that future versions of PDF will introduce hypertextual features in a more abstract form, which links them with other objects in the PDF file rather than anchoring them to fixed page positions. In the meantime it is pleasing to note that standard Computer Science techniques of delayed binding will cope quite happily with the 'absolute' positioning requirements of the present generation of PDF hyperlinks.

5. Acknowledgements

Thanks are due to John Wiley and Sons Ltd, Chapman & Hall Ltd, Adobe Systems Inc and Adobe (UK) Ltd. for sponsorship and software support for the CAJUN project. I am also deeply indebted to my colleagues David Evans, Leon Harrison, Steve Probets, Phil Smith and Peter Sutton for patiently educating me about many aspects of LaTeX, *troff*, PostScript and PDF which had hitherto passed me by. Technical assistance from Adobe Systems — and particularly Ken Anderson, Gary Cosimini and Steve Rogers — has been invaluable in getting to grips quickly with the features of Acrobat release 2.0

References

1. *Dynatext — Electronic Book Publishing and Delivery System*. Electronic Book Technologies Inc. Providence R.I.
2. *World Wide Web*. Further information can be obtained by retrieving a document with URL **`http://info.cern.ch/hypertext/WWW/TheProject`** into a WWW browser (URL = 'Universal Resource Locator').
3. Adobe Systems Incorporated, *Portable Document Format Reference Manual*, ISBN 0-201-62628-4, Addison-Wesley, Reading, Massachusetts, June 1993.
4. Philip N. Smith, David F. Brailsford, David R. Evans, Leon Harrison, Steve G. Probets, and Peter Sutton, ''Journal publishing with Acrobat: the CAJUN project,'' *Electronic Publishing — Origination, Dissemination and Design*, vol. 6, no. 4, pp. 482–493, December 1993.