Wusteman, Judith and Brown, Heather (1994) *Electronic journals using Acrobat, Mosaic and Guide.* Technical report. University of Kent, Computing Laboratory, University of Kent, Canterbury, UK

Electronic Journals using Acrobat, Mosaic and Guide

Judith Wusteman and Heather Brown
Computing Laboratory
University of Kent at Canterbury
England

Abstract

This paper considers some of the issues affecting the use and readability of electronic journals. These issues are considered in relation to prototype electronic journals implemented using Adobe Acrobat[14], the Mosaic browser for the World Wide Web[3, 4] and the Guide hypertext system[6]. The potential of each system as a vehicle for electronic journal provision is discussed.

Keywords

Electronic journals, hypertext, indexing, browsing

1 Introduction

According to publishers at the 1993 Book Fair, electronic journals have now arrived [17]. The concept of the online journal has been in existence for some time; indeed, it is claimed that there were, by 1993, already one hundred and thirty "strictly electronic journals" in circulation [17]. Despite this proliferation, the future form and organisation of such publications is still wide open for debate.

This paper considers some of the issues affecting the use and readability of electronic journals. These issues are considered in relation to Adobe Acrobat[14], the Mosaic browser for the World Wide Web[3, 4] and the Guide hypertext system[6].

The paper first discusses the concept of the electronic journal and introduces the three systems. The techniques that people use to read and skim paper-based journals are considered and an attempt is made to identify which of these could be constructively replicated in electronic form. The availability, or the potential for development, of appropriate online tools to aid journal users is discussed in relation to Acrobat, Mosaic and Guide. This discussion is illustrated using prototype electronic journals based on each of the three systems. Following from this, the potential of each as a vehicle for electronic journal provision is assessed.

2 Electronic Journals

Opinion differs as to what constitutes, or should constitute, an electronic journal. A journal may be defined as "any collection of learned articles which has been accepted via the peer review process for publication as part of a series" [23].

An electronic journal could then be defined as a journal in which the end product is available electronically, whether this is over the network or via storage devices such as the CD-ROM. However, this discounts many of the products currently describing themselves as electronic journals, not least because the majority do not employ the peer review process.

The constituents of the “collection of learned articles” may also differ for an electronic journal. It could be argued that electronic dissemination does away with the need to package articles into journal issues. Articles can be published as soon they are accepted by the journal’s editorial board, thus minimising publication delays. In this environment, an issue based on the dates of article submissions may lose credibility. However, the concept of special issues on particular subjects may still be valid and personalised journal issues, containing articles of particular interest to an individual subscriber, become a possibility.

As to whether the journal itself is an appropriate metaphor to carry over to electronic form, Pullinger [19] points out that a journal is more than a set of articles. Not only does it contain non-article-based information, such as indexes, it also provides a focus for the readership community and a forum for discussion. Indeed, some publishers have introduced bulletin boards for particular journal titles in order to encourage informal discussion among their readership. Readers need to be provided with some method of knowing where and how to look for the latest details of work in a particular field. If papers are not to be presented as predetermined collections, this “awareness function” needs to be provided in some other way.

A continuing problem for electronic journals is that of credibility. One of the main functions of an academic journal is the recognition it provides for an author’s work. Paper journals have an established reputation and therefore carry authority. The peer refereeing of papers enhances this authority. Electronic journals that formally referee submissions are, as yet, in the minority. This problem of credibility does not look as if it will disappear in the near future. It appears that the technical issues involved in electronic journal proliferation may not be as problematic as political and social issues.

3 Acrobat, Mosaic and Guide

This section introduces the main features of Acrobat, Guide and Mosaic. In each case it gives a brief introduction to the underlying representation used by the system and shows how this influences the facilities provided. Examples of prototype electronic journals implemented using each of the three systems are discussed.

Sections 4 to 6 discuss the systems in more detail with particular reference to electronic journal presentation.

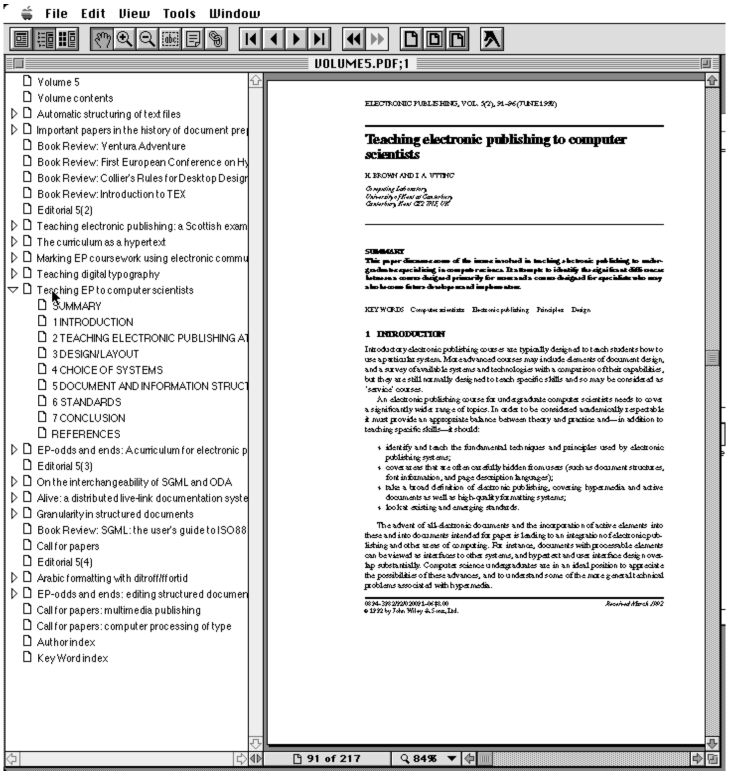
3.1 Acrobat

The Acrobat suite of software uses the Portable Document Format (PDF) [15] as its underlying representation. PDF may be regarded as a version of PostScript [13] with additional features for online presentation and browsing of documents. Thus, Acrobat provides a high quality page-based representation of a document (similar to that provided by a PostScript previewer) together with a limited range of features for browsing and hypertext linking.

PDF supports all the device and resolution-independent page descriptions familiar to PostScript users. The imaging model is the same, but PDF also provides the following features for online viewing

- simple hypertext links
- annotations and bookmarks
- thumbnail sketches of pages

PDF notation is essentially the same as PostScript. For example, a typical link in Version 1 of Acrobat might contain the following information



Volume 5
Volume contents
Automatic structuring of text files
Important papers in the history of document pre
Book Review: Ventura Adventure
Book Review: First European Conference on Hy
Book Review: Collier's Rules for Desktop Design
Book Review: Introduction to TEX
Editorial 5(2)
Teaching electronic publishing: a Scottish exam
The curriculum as a hypertext
Marking EP coursework using electronic commu
Teaching digital typography
Teaching EP to computer scientists
SUMMARY
1 INTRODUCTION
2 TEACHING ELECTRONIC PUBLISHING AT
3 DESIGN/LAYOUT
4 CHOICE OF SYSTEMS
5 DOCUMENT AND INFORMATION STRUCT
6 STANDARDS
7 CONCLUSION
REFERENCES
EP-odds and ends: A curriculum for electronic p
Editorial 5(3)
On the interchangeability of SGML and ODA
Alive: a distributed live-link documentation syste
Granularity in structured documents
Book Review: SGML: the user's guide to ISO88
Call for papers
Editorial 5(4)
Arabic formatting with ditroff/ffortid
EP-odds and ends: editing structured documen
Call for papers: multimedia publishing
Call for papers: computer processing of type
Author index
Key Word index

ELECTRONIC PUBLISHING, VOL. 5(2), 91–96 (JUNE 1992)

Teaching electronic publishing to computer scientists

H. BROWN AND I. A. UTTING

Computing Laboratory
University of Kent at Canterbury
Canterbury, Kent CT2 7NF, UK

SUMMARY
This paper discusses some of the issues involved in teaching electronic publishing to undergraduates specialising in computer science. It attempts to identify the significant differences between a course designed primarily for users and a course designed for specialists who may also become future developers and implementors.

KEY WORDS Computer scientists Electronic publishing Principles Design

1 INTRODUCTION

Introductory electronic publishing courses are typically designed to teach students how to use a particular system. More advanced courses *may* include elements of document design, and a survey of available systems and technologies with a comparison of their capabilities, but they are still normally designed to teach specific skills and so may be considered as 'service' courses.

An electronic publishing course for undergraduate computer scientists needs to cover a significantly wider range of topics. In order to be considered academically respectable it *must* provide an appropriate balance between theory and practice and—in addition to teaching specific skills—it should:

- identify and teach the fundamental techniques and principles used by electronic publishing systems;
- cover areas that are often carefully hidden from users (such as document structures, font information, and page description languages);
- take a broad definition of electronic publishing, covering hypermedia and active documents as well as high-quality formatting systems;
- look at existing and emerging standards.

The advent of all-electronic documents and the incorporation of active elements into these and into documents intended for paper is leading to an integration of electronic publishing and other areas of computing. For instance, documents with processable elements can be viewed as interfaces to other systems, and hypertext and user interface design overlap substantially. Computer science undergraduates are in an ideal position to appreciate the possibilities of these advances, and to understand some of the more general technical problems associated with hypermedia.

0894–3982/92/020091–06$8.00 *Received March 1992*
© 1992 by John Wiley & Sons, Ltd.

Figure 1: An article page from *EP-odd* viewed using Exchange

```
/Type  /Annot
/Subtype  /Link
/Rect [ 140  805  250  825 ]
/Border [ 12  12  1 ]
/Dest [ 3  0  R/FitR  3  270  600  748 ]
```

This specifies the source of the link (the location of a rectangular area on the current page, and border information) and its destination (the page number, location of rectangular area, and information on how it should be displayed). The use of the hypertext features is discussed further in Sections 5 and 6.

Acrobat itself comprises four software tools: Distiller, PDFWriter, Exchange, and Reader. PDF files may be created either by converting PostScript files to PDF using the Distiller or by creating them directly from applications via the pseudo printer driver PDFWriter [22]. Hypertext features may be included at this stage via a new PostScript operator, **pdfmark**, introduced for this purpose. Online presentation and browsing of PDF documents is via the Exchange software. This allows links, annotations, and bookmarks to be created and used. The Reader is a limited version of the Exchange software.

Acrobat was first announced in late 1992. In Version 1, only intra-document links are possible. Version 2, announced in October 1994, introduces further facilities including iter-document links.

In summary, Acrobat aims to exploit the enormous international investment in PostScript and provides hypertext facilities on top of the PostScript imaging model.

3.1.1 The CAJUN Project

Acrobat is being used for electronic journal dissemination by the CAJUN (CD-ROM Acrobat Journals Using Networks) project[1]. The archives of several journals, including Wiley's *Electronic Publishing—Origination, Dissemination and Design (EP-odd)*, have been converted to PDF. A sample volume from *EP-odd* is used throughout this paper to illustrate the ideas discussed. Figure 1 shows a page from the CAJUN version of an *EP-odd* article. A new release of the *EP-odd* archive using Acrobat Version 2 is expected in the last quarter of 1994 [10].

The CAJUN project is on-going and further work is intended in network dissemination. Smith *et al* [22] describe the two categories of network dissemination as 'push' and 'pull'. In the former, information is transferred to subscribers by the publisher. The latter refers to the more flexible approach whereby subscribers access information of their choice.

Initially, CAJUN concentrated on the 'push' approach. Sample papers were sent to chosen test sites, both via CD-ROM and using file transfer methods. More recently, sample PDF papers have been made available for the 'pull' approach via network information tools such as Gopher, World Wide Web and anonymous file transfer protocol[2].

3.2 Mosaic

NCSA Mosaic is a distributed hypermedia browser for information retrieval over the Internet. It is based on the World Wide Web (WWW) technology and uses WWW's HTML (Hypertext Markup Language) [5] as its underlying representation.

HTML is a simple application of the SGML standard [12]. It is a straightforward markup system using SGML tags and attributes to identify items such as titles, paragraphs, and links. A simple example is shown below

```
<H1>Introduction to Computing Service Courses</H1>
<UL>
<LI> <A href="general.html">General Information</A>
<LI> <A href="reg.html">Registration</A>
<LI> <A href="cancel.html">Cancellation</A>
</UL>
```

The `<H1>` and `</H1>` tags mark the beginning and end of a main heading and the `<UL>` and `<LI>` tags indicate the start of a list and items within the list respectively. Links or 'anchors' are indicated by `<A>` tags. The link name is given by the text between the `<A>` and `</A>` tags, and the link target (i.e. the document that is displayed when the link is selected) is identified by giving its Universal Resource Locator or URL[2][3] as the value of an `href` attribute in the `<A>` tag. In the example above, the URLs are simple local file names, but they may specify the protocol to be used to access the target, and its location as well. For example

```
href="http://www.ukc.ac.uk/comp_lab/service.html".
```

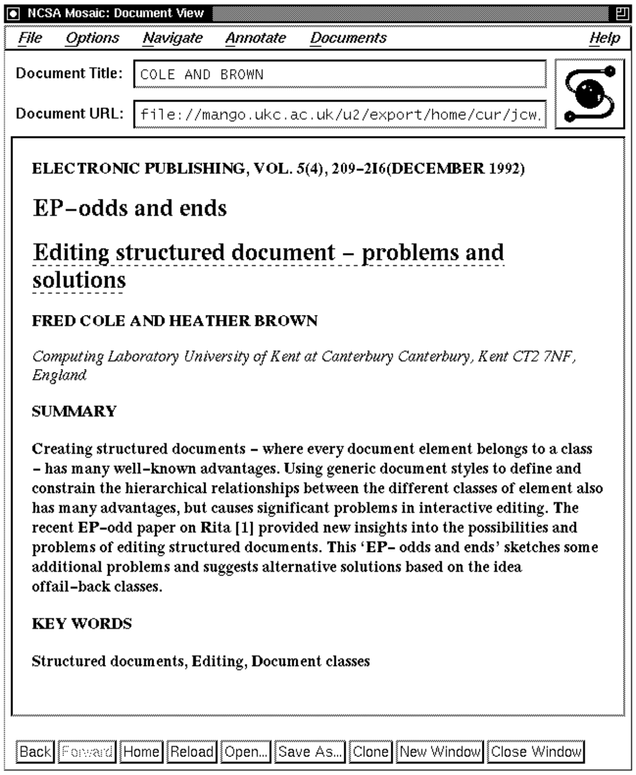
Unlike Acrobat, Mosaic documents are not page-based; their contents are presented as continuous scrolls, formatted dynamically to fit the window size.

Mosaic aims to provide 'a unified interface to the various protocols, data formats and information archives used on the Internet'[1]. It is based on a client/server architecture in which servers at specific Internet sites respond to queries sent by clients from anywhere on the Internet. A unit of information sent from a server to a client is referred to as a document. This document may be almost any form of information, from a stand-alone text-file to the result of a database

[1] Based at the University of Nottingham and jointly funded by John Wiley & Sons Ltd. and Chapman and Hall Ltd.

[2] Further information concerning access of sample CAJUN documents via Gopher, World Wide Web and anonymous ftp may be obtained from circus@cs.nott.ac.uk.

[3] Further information about URLs may be found at the following WWW address: http://info.cern.ch/hypertext/WWW/Addressing/Addressing.html



Document Title: COLE AND BROWN

Document URL: file://mango.ukc.ac.uk/u2/export/home/cur/jcw.

ELECTRONIC PUBLISHING, VOL. 5(4), 209–2I6(DECEMBER 1992)

EP–odds and ends

Editing structured document – problems and solutions

FRED COLE AND HEATHER BROWN

Computing Laboratory University of Kent at Canterbury Canterbury, Kent CT2 7NF, England

SUMMARY

Creating structured documents – where every document element belongs to a class – has many well–known advantages. Using generic document styles to define and constrain the hierarchical relationships between the different classes of element also has many advantages, but causes significant problems in interactive editing. The recent EP–odd paper on Rita [1] provided new insights into the possibilities and problems of editing structured documents. This 'EP– odds and ends' sketches some additional problems and suggests alternative solutions based on the idea offail–back classes.

KEY WORDS

Structured documents, Editing, Document classes

Figure 2: An *EP-odd* article viewed using Mosaic

query. Thus, documents don't have to exist as files; they can be 'virtual' documents generated by a server in response to a query or document name[4]. In fact, a document can be anything 'that is available somewhere on the network through some network-based access mechanism'[2].

Mosaic makes no attempt to hide the distributed nature of its documents. Typically, when the user clicks on a link, Mosaic displays a new document and the user is made aware of this via presentation of the URL at the bottom of the window.

In summary, Mosaic is specially designed to exploit the variety of information available in WWW and to aid browsing and searching in this very large information space. The use of HTML as its underlying representation helps it to capture the logical structure of documents.

3.2.1 Journals in Mosaic

Project MUSE enables WWW access to scholarly journals published by the John Hopkins University Press[18][4]. The initial prototype provides access to current issues of three journals—Configurations, Modern Language Notes, and English Literary History—via a Mosaic server.

Like CAJUN, MUSE aims to provide journal access with facilities that cannot be provided in print. MUSE also aims to develop a costing model for network access and, in the longer term, to make works of scholarship available at reasonable prices to University Libraries and the academic community. If funds can be raised, the Press hopes to make about forty of its journals available in this way in advance of the printed version.

In addition to access via a variety of indexes, MUSE offers hypertext links to notes and illustrations, searches, and voice and textual annotations. Users can exploit Mosaic's facilities to download text for printing and build up 'hot lists' of frequently accessed documents.

[4] Further information may be accessed on the MUSE project at the following WWW address: http://muse.mse.jhu.edu.

To facilitate comparison of the three systems, sections of a volume of *EP-odd* have been converted into HTML; an example article abstract displayed using Mosaic is shown in Figure 2.

3.3 Guide

Guide is a hypertext system running on UNIX workstations[6]. It is a true hypertext system designed to free documents from the constraints of paper. As with Mosaic, a Guide document is not page-based; it is best thought of as a single continuous scroll with sections folded behind buttons. A document is often presented initially as a summary or an index that allows the reader to select buttons to reveal details of areas of interest. Guide may also be used like a card-based hypertext system where information is presented as a series of separate nodes with links leading from one node to another.

Guide documents may contain a mixture of text and images, but the imaging model is relatively simple and the text formatting is designed for online presentation. Again, as in Mosaic, text formatting changes dynamically if the window size changes. Unlike many hypertext systems, however, Guide supports 'contexts' which allow the logical document structure to be captured. This helps with intelligent searching and systematic formatting (all 'titles' to be presented in a particular font, for example) [7].

Guide has its own underlying representation[9]. This is based on the *roff* command format with document content interspersed with *roff*-like mark-up. An example of a simple button and its replacement text is

```
.Bu l 1 n
Button
.bU
.Re
Text to be inserted inline when 'Button' is selected.
This may contain further nested buttons.
.rE
```

where `.Bu` and `.bU` delimit the button name and `.Re` and `.rE` delimit the replacement information shown when the button is selected. Button definitions may be nested to provide a hierarchical structure. In addition to this extremely simple syntax, 'extensions' to the button name may specify complex actions that allow the replacement for the button to be found elsewhere or generated dynamically by running external programs. Two very simple examples are

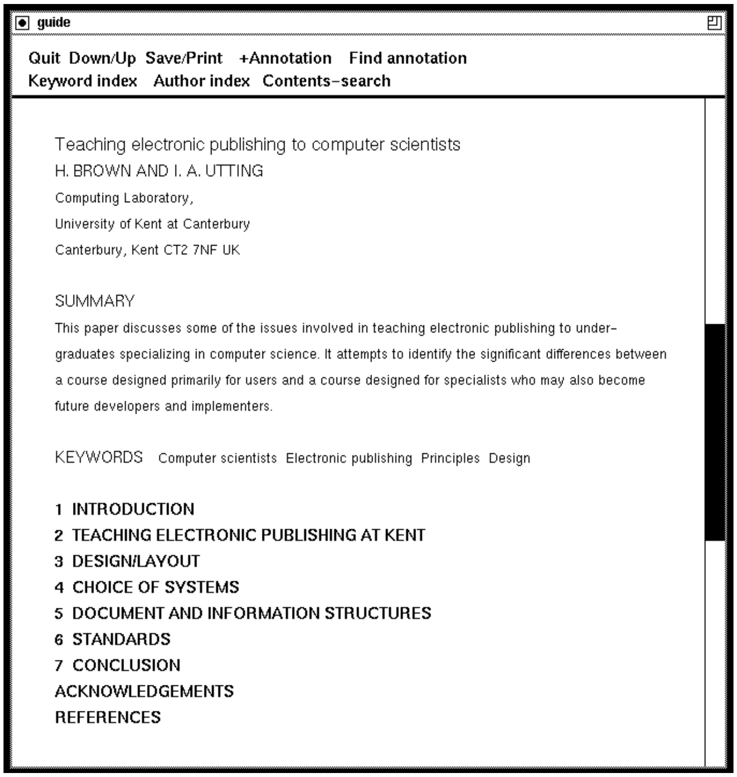
```
.Bu u 1 n
Jekyll    alias Hyde     in RLStevenson
.bU

.Bu u 1 n
Current date and time     run date
.bU
```

The first example causes the button `Jekyll` to obtain its replacement from a button called `Hyde` in a separate file called `RLStevenson`, the second causes the button `Current date and time` to obtain its replacement dynamically by running the UNIX command `date`.

The button name extensions constitute a simple language that is interpreted by Guide when the button is selected, thus providing the system with a great deal of flexibility and programmability. The examples above show that the same information may appear at different places within a document by using `alias`, and that Guide can be used as a front-end to other systems, such as information retrieval software, by using `run`.

A recent addition to Guide [8] allows information to be accessed remotely across the Internet and inserted into the document as a replacement for a button. This is done using a mechanism similar to the URL concept used in Mosaic.



Teaching electronic publishing to computer scientists

H. BROWN AND I. A. UTTING

Computing Laboratory,
University of Kent at Canterbury
Canterbury, Kent CT2 7NF UK

SUMMARY

This paper discusses some of the issues involved in teaching electronic publishing to undergraduates specializing in computer science. It attempts to identify the significant differences between a course designed primarily for users and a course designed for specialists who may also become future developers and implementers.

KEYWORDS Computer scientists Electronic publishing Principles Design

1 INTRODUCTION
2 TEACHING ELECTRONIC PUBLISHING AT KENT
3 DESIGN/LAYOUT
4 CHOICE OF SYSTEMS
5 DOCUMENT AND INFORMATION STRUCTURES
6 STANDARDS
7 CONCLUSION
ACKNOWLEDGEMENTS
REFERENCES

Figure 3: An *EP-odd* article viewed using the Guide prototype application

Guide provides an authoring mode which allows users to create buttons dynamically. Authors do not need to be aware of the underlying representation, but they do need to understand the facilities provided by the button name extensions in order to specify complex actions.

In summary, Guide provides a flexible means of representing electronic documents based on a continuous scroll metaphor. It emphasises hypertext features and flexibility rather than high quality formatting, and has simple networking facilities.

3.3.1 Documents in Guide

Many large paper documents have been converted into Guide [20, 16] but there has been no significant Guide implementation of an online journal. A prototype Guide version of a Volume of *EP-odd* was therefore developed to demonstrate the capabilities of Guide in this context. Figure 3 shows a typical Guide screen from this prototype.

4 Browsing and Reading Journals Online

In order to determine the most appropriate form of presentation for electronic journals, we must work backwards from a consideration of how we read hard-copy journals to a determination of why we use these particular methods. Only then can we distinguish the underlying principles of readability from the limitations imposed by a paper-based technology. We can then ensure that we translate into electronic form only those hard-copy features that aid readability.

The differing information needs of users lead to differing modes of document perusal. For example, users often skim journals for general information concerning their structure and the content of individual articles. In addition, they may study particular papers in more depth.

Sections 5 and 6 discuss some of the techniques that readers have evolved for reading and skimming paper-based journals, and the journal features that reinforce or encourage these techniques. An attempt is made to identify those features which could be constructively replicated in electronic form and to consider the potential of each of the three systems to provide these features.

Section 5 considers these issues in relation to browsing journals and their constituent articles. Section 6 discusses the more detailed study of articles. The CAJUN and MUSE projects and the Guide and Mosaic *EP-odd* journal prototypes are used to illustrate the issues raised.

As already mentioned, the form in which electronic journal articles will be packaged in the future is open for debate. Users may access, or be provided with, collections of articles ordered, for example, by keyword, date, author, citation, volume or issue. Alternatively, they may access individual articles from a remote database chosen using the same criteria. In the case of Guide and Mosaic, individual articles, and other features such as reviews and notices, may be stored at different locations across a network. Due to the lack of inter-document links in Acrobat Version 1, it is assumed, for the purposes of this paper, that a collection of articles and features is held in one PDF file. This enables cross references between, for example, indexes and articles.

5 Browsing Journals

A well-designed hard-copy journal has a clearly defined structure at both the journal and individual article level. Although this structure may differ from one title to another, it should be easily recognisable to the user, thus simplifying the tasks of orientation, navigation and information retrieval.

Paper-based journals have developed conventions for presenting information concerning their content and structure. Some general conventions or standards will need to be developed for the electronic presentation of such information so that readers can instinctively access the various features of an electronic journal without any previous knowledge of that journal's setup. Such features might include the contents page, any indexes, and general information concerning the journal, such as notes for authors and subscription details. In addition, each article and its subcomponents should be instinctively accessible. For paper-based journals, this implies knowing where the various components are. For electronic journals, the user may not need to know the actual location of a component; the emphasis may rather be on knowing how to access them.

It is necessary to consider what might constitute appropriate default access, browsing and navigation tools for electronic journals and how these tools should function. For example, how does a user choose a hypertext link and what result does this choice have? The answer to such questions may differ for electronic journals and other documents.

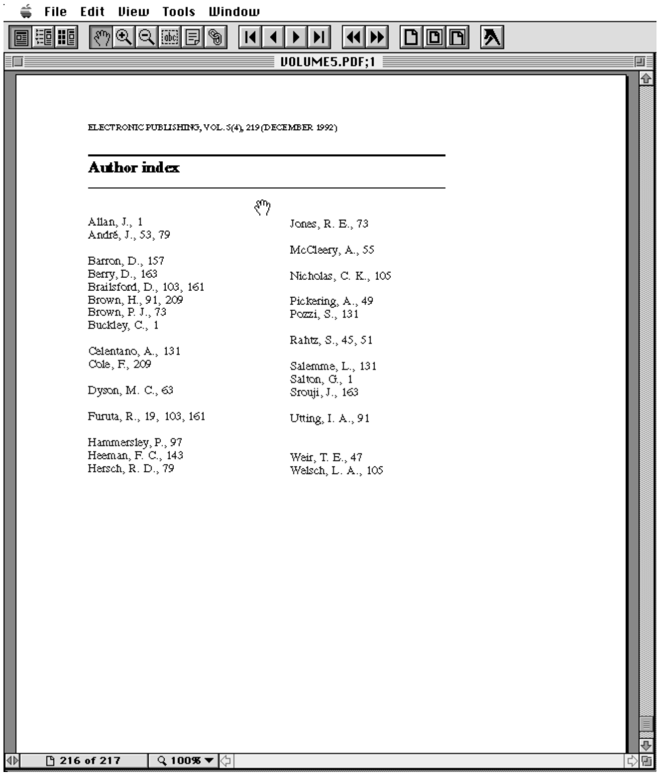
5.1 Indexes

However articles are packaged or accessed, some form of indexing will be required, whether locally or at the remote source. For all three systems, "active indexes", based on the hypertext facilities, have the potential to add extra value to that provided by paper-based journals.

5.1.1 Acrobat

As indicated in Section 4, cross-article links may be simulated with Version 1 Acrobat software by storing several concatenated articles in a single file. As a result, the initial version of the CAJUN *EP-odd* archive on CD-ROM is organised into volumes. Each PDF file represents one volume. Thus, links may be made from volume contents pages, keyword and author indexes to the appropriate papers. This allows for immediate access to articles or features via a simple mouse click on the relevant index item.

A sample author index page is illustrated in Figure 4. The page numbers of the original hard-copy version have been retained to allow for situations in which one author name refers to more than one article. In this case, clicking on the author name or first page number results in the display of the first relevant article. Clicking on a different page number will display the article starting on that page. This solution to the problem of multiple references is only meaningful in page-based viewing systems.



ELECTRONIC PUBLISHING, VOL. 5(4), 219 (DECEMBER 1992)

Author index

Allan, J., 1
André, J., 53, 79

Barron, D., 157
Berry, D., 163
Brailsford, D., 103, 161
Brown, H., 91, 209
Brown, P. J., 73
Buckley, C., 1

Celentano, A., 131
Cole, F., 209

Dyson, M. C., 63

Furuta, R., 19, 103, 161

Hammersley, P., 97
Heeman, F. C., 143
Hersch, R. D., 79

Jones, R. E., 73

McCleery, A., 55

Nicholas, C. K., 105

Pickering, A., 49
Pozzi, S., 131

Rahtz, S., 45, 51

Salemme, L., 131
Salton, G., 1
Srouji, J., 163

Utting, I. A., 91

Weir, T. E., 47
Welsch, L. A., 105

Figure 4: An Author Index displayed using Exchange

Following an Acrobat hypertext link, such as an index item, results in the display switching to the page on which the link destination appears. Thus, the reader loses sight of the index, although they may return to this view using the navigation feature described in Section 5.4. This temporary loss of context parallels the situation in following an index item in a paper-based journal.

The **bookmark** is a built-in hypertext feature which may be used in an indexing role. Bookmarks can also provide an outline of an article, as illustrated in Figure 1.

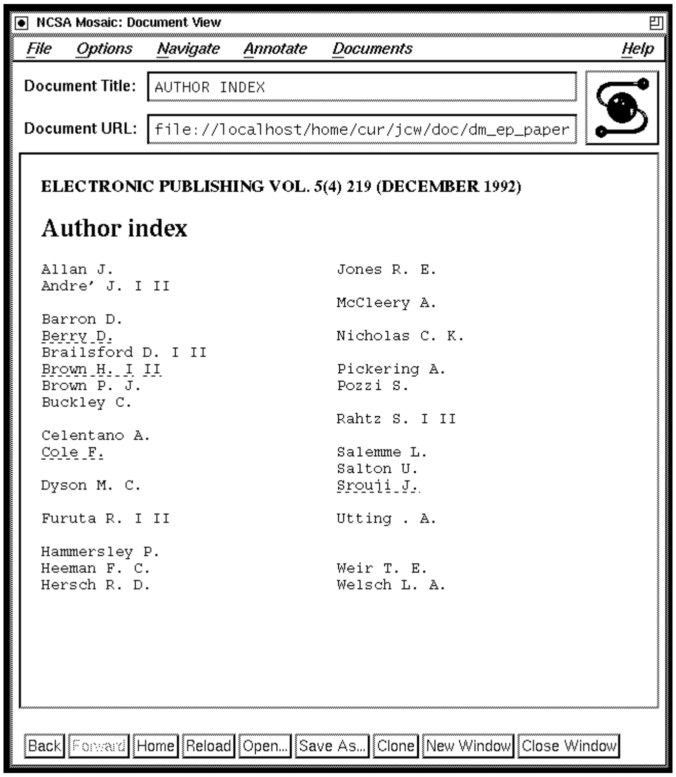
In this example, the bookmarks are nested, main headings being represented by top level bookmarks, subheadings by the next level down and so on. Papers can optionally be set up so that, on opening them, only those bookmarks above a specified level are initially visible. In this case, further levels of bookmarks can be opened out by clicking on the triangle associated with the visible bookmark entry. In Figure 1, the top level bookmarks and one set of lower level bookmarks are visible.

In the papers distributed by the CAJUN project, the bookmark facility is set up during development of the PDF file to provide a contents summary based on section headings and subheadings. Bookmarks can also be added using Exchange by choosing the appropriate option from the Edit menu. User-defined bookmarks may be added to any provided by the PDF document developer.

5.1.2 Mosaic

As access to information via Mosaic is based on the traversal of a network of links, the development of contents pages, author or keyword indexes is straightforward. However, as Mosaic does not incorporate the concept of a page, it is not possible to distinguish between multiple occurrences of the same author or key word using page numbers. One solution to this problem is illustrated in Figure 5 in which each of the Roman numerals after the author name Brown, for example, indicates a link to a different article.

As with Acrobat and Guide, Mosaic hypertext links are activated by pointing and clicking



Document Title: AUTHOR INDEX

Document URL: file://localhost/home/cur/jcw/doc/dm_ep_paper

ELECTRONIC PUBLISHING VOL. 5(4) 219 (DECEMBER 1992)

Author index

Allan J.
Andre' J. I II

Barron D.
Berry D.
Brailsford D. I II
Brown H. I II
Brown P. J.
Buckley C.

Celentano A.
Cole F.

Dyson M. C.

Furuta R. I II

Hammersley P.
Heeman F. C.
Hersch R. D.

Jones R. E.

McCleery A.

Nicholas C. K.

Pickering A.
Pozzi S.

Rahtz S. I II

Salemme L.
Salton U.
Srouji J.

Utting . A.

Weir T. E.
Welsch L. A.

Figure 5: An Author Index in Mosaic

using a mouse. Via a single click of the left mouse button, the Mosaic document associated with a link is displayed in the current window. A click of the middle mouse button results in the document appearing in a new window. This may then be positioned and sized by the user to enable the link source and destination to be viewed simultaneously.

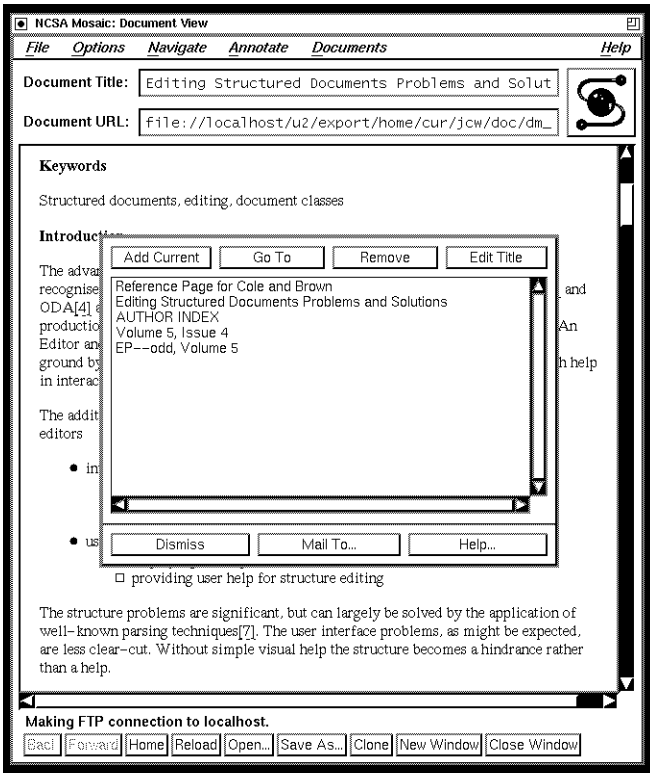
Although MUSE provides hypertext links within articles and to articles via the table of contents, it relies on search facilities to a greater extent than does CAJUN, as described in Section 5.3.

Whereas Acrobat's bookmark feature provides access to points within a single document, Mosaic's **hotlist** feature also enables access to different documents. It allows users to store important or often-accessed hyperlinks such as the reference pages of an article. As illustrated in Figure 6, the hotlist window displays the titles of stored documents in the order they were added. Users can visit any listed document by choosing the title. A function exists to allow a hotlist to be sent, in HTML form, to another user via e-mail. The **window history** provides a similar function, although in this case, a title list of document views accessed in each window is automatically saved. As with hotlists, any view listed may be accessed by double clicking on its title and the window history may be mailed to other users.

5.1.3 Guide

Although Guide has no built-in feature corresponding to either bookmarks or hotlists, its flexibility allows equivalent features to be set up easily.

In the prototype, the initial view corresponds to the normal contents summary. Clicking on an item provides progressively more information about the relevant paper, as illustrated in Figure 7. In this way, users may see just the Summary, a list of subtitles, or the entire text. This is similar in many respects to the Acrobat bookmark feature, but clicking on a title reveals the associated text at that point in the scroll rather than causing a jump to another page in a separate view.


NCSA Mosaic: Document View
File Options Navigate Annotate Documents Help
Add Current Go To Remove Edit Title
Dismiss Mail To... Help...
Back Forward Home Reload Open... Save As... Clone New Window Close Window

Document Title: Editing Structured Documents Problems and Solut

Document URL: file://localhost/u2/export/home/cur/jcw/doc/dm_

Keywords

Structured documents, editing, document classes

Introduc

The adva
recognise
ODA[4] a
productio
Editor an
ground by
in interac

and
An
h help

The addit
editors

- in

- us

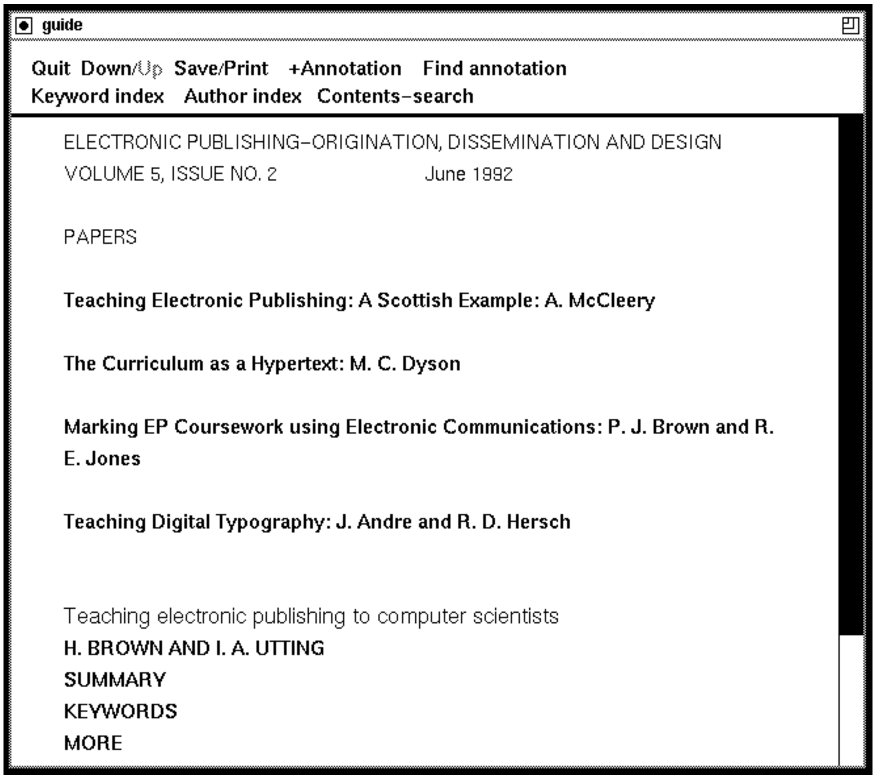
□ providing user help for structure editing

Reference Page for Cole and Brown
Editing Structured Documents Problems and Solutions
AUTHOR INDEX
Volume 5, Issue 4
EP--odd, Volume 5

The structure problems are significant, but can largely be solved by the application of well-known parsing techniques[7]. The user interface problems, as might be expected, are less clear-cut. Without simple visual help the structure becomes a hindrance rather than a help.

Making FTP connection to localhost.

Figure 6: The Mosaic Hotlist facility


guide
Quit Down/Up Save/Print +Annotation Find annotation
Keyword index Author index Contents-search

ELECTRONIC PUBLISHING-ORIGINATION, DISSEMINATION AND DESIGN
VOLUME 5, ISSUE NO. 2 June 1992

PAPERS

Teaching Electronic Publishing: A Scottish Example: A. McCleery

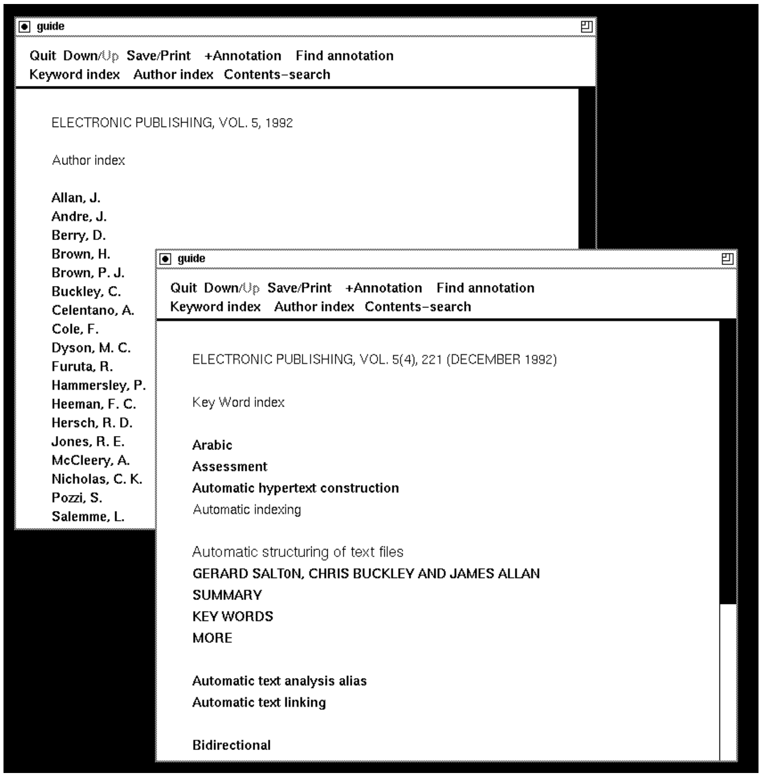
The Curriculum as a Hypertext: M. C. Dyson

Marking EP Coursework using Electronic Communications: P. J. Brown and R. E. Jones

Teaching Digital Typography: J. Andre and R. D. Hersch

Teaching electronic publishing to computer scientists
H. BROWN AND I. A. UTTING
SUMMARY
KEYWORDS
MORE

Figure 7: Article Details in Guide


guide
Quit Down/Up Save/Print +Annotation Find annotation
Keyword index Author index Contents–search

ELECTRONIC PUBLISHING, VOL. 5, 1992

Author index

Allan, J.
Andre, J.
Berry, D.
Brown, H.
Brown, P. J.
Buckley, C.
Celentano, A.
Cole, F.
Dyson, M. C.
Furuta, R.
Hammersley, P.
Heeman, F. C.
Hersch, R. D.
Jones, R. E.
McCleery, A.
Nicholas, C. K.
Pozzi, S.
Salemme, L.

guide
Quit Down/Up Save/Print +Annotation Find annotation
Keyword index Author index Contents–search

ELECTRONIC PUBLISHING, VOL. 5(4), 221 (DECEMBER 1992)

Key Word index

Arabic
Assessment
Automatic hypertext construction
Automatic indexing

Automatic structuring of text files
GERARD SALTON, CHRIS BUCKLEY AND JAMES ALLAN
SUMMARY
KEY WORDS
MORE

Automatic text analysis alias
Automatic text linking

Bidirectional

Figure 8: The result of selecting keyword and author indexes and then clicking on the keyword `Automatic indexing`

In addition to access via the contents information, the prototype also allows access via the author and keyword indexes of *EP-odd*. Clicking on the menu item `Keyword index` or `Author index` brings up the appropriate index in a separate view. Subsequent clicks on an author's name or a keyword leads to details of all the relevant papers in exactly the same form as via the main contents view. This approach maintains the notion of the Guide document as a continuous scroll with sections hidden behind buttons. However, if required, it would be simple to replicate the Acrobat or Mosaic approach to following index items in which the destination appears as a distinct document.

To demonstrate the flexibility of Guide, the keyword index appears in a separate window and the author index covers the original view. Figure 8 shows the result of selecting both indexes and then clicking on the keyword `Automatic indexing`.

It would be fairly simple to implement some form of hotlist or window history via a new menu item and some simple shell scripts which maintain a list of certain types of buttons accessed.

5.1.4 Link Automation

For the time-being at least, authors will continue to write articles geared, in the main, for submission to conventional paper-based journals. Thus, the journal articles involved may have been developed before the question of adding hypertext links becomes relevant. In this case, some of the central questions are

- Should defaults exist as to which features of an article should be linked?
- If so, what should these defaults be?
- Who will add links to the text?

- At what stage of article development will the links be added?

The default chosen in Version 1 of the CAJUN project is to add links to most features in which there is a cross-reference to other parts of the same document. This includes links to figures, tables, equations and referenced sections, as well as to cited references and from author and keyword indexes. The same defaults appear appropriate for the Mosaic and Guide representations of *EP-odd* although, here, inter-document links are also possible. As well as linking citations to bibliography pages, MUSE also links bibliographic references back to the citations, thus providing additional flexibility of navigation.

However, the addition of hypertext links may be a time-consuming and tedious process. As a result of this, some form of link automation is foreseen in the CAJUN project [11]. HTML editors exist to ease the authoring process for Mosaic documents, though currently available editors are described as "inadequate"[21]. As regards Guide, the automatic construction of, for example, components of indexes is relatively simple due to Guide's underlying format and its easy interaction with other tools.

The time constraints involved in journal production may be even more stringent than in the development of a one-off hypertext system and consistency of journal structure will be of paramount importance. These constraints may be limiting factors for the form of tools and features it is practicable to provide for electronic journals.

5.2 Accessing Other Information

In the CAJUN and MUSE projects and in the Guide prototype, hypertext facilities provide access to information such as the notes for authors, details of the editorial board and so on. In the case of CAJUN Version 1, the link's destination is always local, whereas the destination of links in the MUSE project, and potentially in the Guide prototype, may be anywhere on the Internet. The advantage of the latter is that such information is always available without it being repeated in every journal issue. It has been suggested [8] that users prefer to retain local copies of information in case the remote link disappears or the information changes. However, this form of information would be required only infrequently so that access to a remote link might be acceptable in this case. Remote browsing of articles may be less acceptable, as discussed below.

5.3 Search and retrieval facilities

All three systems provide basic tools for searching for occurrences of words or phrases within a document.

If a Mosaic server has a suitable search engine, index documents may be made available; the reader may use a keyword search on such a document. The MUSE project allows full text search over all journal articles. Words may be combined using the logical connectives **and**, **or** and **not**. Results of searches are ordered so that the articles with the largest number of terms requested are presented to the user first. In addition, readers are able to search the tables of contents of all the journals involved. This feature is made more useful by the fact that subject headings come from the Library of Congress subject headings list and are assigned by experienced librarians. Extensive use of search facilities, in addition to hypertext links, may be particularly appropriate in an application such as electronic journal provision, in which a large quantity of material must be processed within strict time constraints. It reduces the time-consuming process of manually adding hypertext links. This use of sophisticated search facilities is a good example of how the introduction of online journals may provide an opportunity to improve on the methods currently employed with their paper equivalents.

Guide includes a simple built-in string search facility that covers all views and may cover external links as well (at the choice of the author). The search may be limited to certain specified contexts or to button names.

In the prototype, this built-in facility has been tailored to

- Cover the whole of Volume 5 of *EP-odd* but not to extend to links outside Volume 5.

- Use Guide's 'context' facility to allow full-text search or to limit it to the front matter of papers (title, author information, summary and keywords).

The `Contents-search` menu item, as seen in Figure 3, launches a short dialogue with the user to specify the string to search for, whether to search front matter or full text and whether to start searching at the beginning of the view or from the current position.

Version 2 of the Acrobat software introduces a full search and retrieval engine and indexing capability.

5.4 Browsing Articles

One of the main considerations in the provision of browsing tools is that of speed. The rapidity and ease with which a user may pick up a paper-based journal and manually flick through it is difficult to replicate online, particularly when the document is being accessed remotely.

Acrobat provides various features to facilitate movement within documents. These include the **browse** and **navigation** buttons. The browse buttons are the group of four arrow-headed buttons on the tool bar in Figure 4. They allow movement to the first and last pages of a document and to the next and previous page in numerical order. The double-headed navigation buttons, to the right of the browse buttons, allow users to retrace their steps through a document, moving to each view in the order visited.

Although not as fast as the manual browsing process, response time to browse and navigation buttons is relatively short, partly due to the fact that the documents are local.

As Guide and Mosaic do not incorporate the concept of the page, the principles of navigation are slightly different. Both provide scrolling facilities. As mentioned in Section 5.1.2, Mosaic also allows the user to view a history of the documents, or document views, displayed in the current window. Users may navigate forward and backward through these documents or move to any one of them at random.

The replacement of any Guide button may be *undone* with a single mouse click, returning the user to the original button text or picture. Thus the user may navigation around the document by choosing buttons and then undoing their choices.

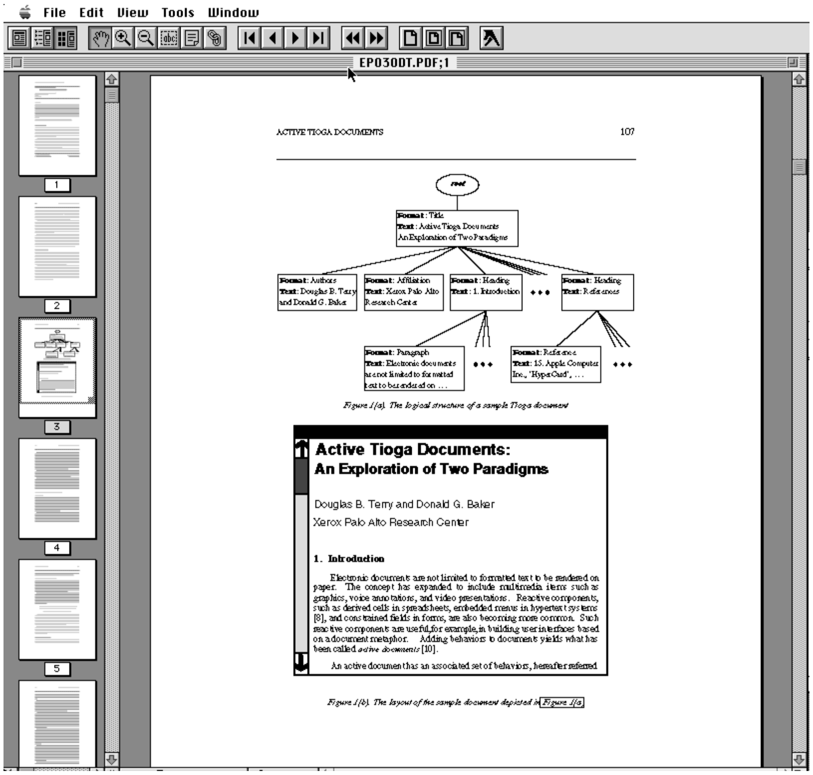
Another browse tool available in Acrobat is the **thumbnail** feature, as illustrated in Figure 9. This provides a gallery of miniature images of the pages of a document. As can be seen, figure outlines are clearly distinguishable. When presented online, the text of each page is represented as a grey area on the thumbnail sketches. The thumbnail feature is of limited use in electronic journals as there is little information to be gleaned from the thumbnail sketches, except in situations in which there are a large number of diagrams spread evenly through the paper. For this reason, and because the concept relies on a page-based document model, the translation of the thumbnail feature into equivalent features in Mosaic or Guide is not considered.

6 Studying a Paper

As well as browsing articles or collections of articles, users may also wish to study individual papers in more depth. A first guess at the type of tool which may be useful in this process may be made by considering the procedures that readers follow when studying a paper copy of an article.

When readers are interested in studying one particular journal paper, they often start by looking up its location in the journal's contents page. Having found its page number, they may find the first page and flick through the article to get a feel for its size and structure, keeping track of the start of the article by keeping a finger at the first page. They may then read the summary and the headings and sub-headings, trying to get a feel for the length and structure of text and diagrams under each heading. If, after this process, the article appears particularly interesting or relevant, they may photocopy it and then personalise that copy. Personalising an article may consist of some or all of the following actions.

- Highlighting headings and subheadings,


File
Edit
View
Tools
Window
EP030DT.PDF;1
1
2
3
4
5
root
Format : Title
Text : Active Tioga Documents
An Exploration of Two Paradigms
Format : Authors
Text : Douglas B. Terry and Donald G. Baker
Format : Affiliation
Text : Xerox Palo Alto Research Center
Format : Heading
Text : 1. Introduction
Format : Heading
Text : References
Format : Paragraph
Text : Electronic documents are not limited to formatted text to be rendered on . . .
Format : Reference
Text : 15. Apple Computer Inc., 'HyperCard', . . .

ACTIVE TIOGA DOCUMENTS 107

Figure 1(a). The logical structure of a sample Tioga document

Active Tioga Documents:
An Exploration of Two Paradigms

Douglas B. Terry and Donald G. Baker

Xerox Palo Alto Research Center

1. Introduction

Electronic documents are not limited to formatted text to be rendered on paper. The concept has expanded to include multimedia items such as graphics, voice annotations, and video presentations. Reactive components, such as derived cells in spreadsheets, embedded menus in hypertext systems [8], and constrained fields in forms, are also becoming more common. Such reactive components are useful,for example,in building user interfaces based on a document metaphor. Adding behaviors to documents yields what has been called *active documents* [10].

An active document has an associated set of behaviors, hereafter referred

Figure 1(b). The layout of the sample document depicted in Figure 1(a).

Figure 9: The thumbnail feature in Exchange

- Highlighting important or useful ideas and main concepts (perhaps using a colour code),
- Adding notes in margins,
- Circling important words,
- Underlining important sentences.

At the same time as performing the above actions, they may read the article. They may not manage to read the entire paper in one sitting. When they decide to stop, they may leave the photocopy open at the page they have reached or make some mark on the copy to indicate where they stopped reading. In addition to marking the photocopy, some readers make separate notes, possibly grouped under the same headings as used in the article.

When users look back at the paper, their personalisations give it added value for them. They probably don't give the paper added value for anyone else and, if colleagues ask to look at the paper, users may wish they had remembered to take two photocopies and left one pristine.

These procedures have their equivalent in the study of online articles, as described below.

6.1 Saving and Printing Documents

As indicated above, if readers are interested in studying a particular journal article in some detail, they will probably require their own local copy which they can read and annotate online. They may also want to produce a hard copy.

Saving hypertext documents involves determining what information to save and, if information is dynamic, what version to save. Similar decisions need to be made when producing hard copies.

Mosaic allows the user to save the current document to a local file in one of several formats; plain text, pretty-formatted text, raw HTML, or PostScript. There is also an option to send the document to a local printer in one of these same formats.

Guide provides the user with various options for saving documents. In the prototype system, the Save and Print menu items shown in Figure 8, have been tailored so that choice of the Save item results in the Guide source file being saved along with any edits or annotations added by the user. The Print option dumps the current view of the file, with the current settings of buttons, to an ASCII file. Guide does not place great emphasis on WYSIWYG paper output so this is a fairly crude version. It is, however, fairly simple to translate a Guide source file into a format such as LaTeX if a higher quality formatted printout is required.

Acrobat places more emphasis on high quality output than do Guide or Mosaic. The PDF file may be sent to a PostScript printer, although the hard-copy output will not, of course, include any hypertext features that were in the original PDF file.

6.2 Editing Articles

Both Guide and Mosaic allow the user to save and edit their own versions of documents. Acrobat differs from Guide and Mosaic in that, although users may alter the hyper-textual features, they may not alter the underlying formatted text.

The method of authoring the initial document also varies between the systems. As already described, the original Acrobat document may be developed in one of a variety of formats which is then converted into PDF using the PDFWriter or Distiller. As mentioned in Section 3.1, if a PostScript version is available, hypertext features may be added before conversion to PDF. Alternatively, such features may be included via the WYSIWYG Exchange interface.

In contrast, Guide incorporates an authoring tool which provides one simple interface for both editing text and adding hypertext structure. This is available to both readers and original authors.

Similarly with Mosaic, the author can directly edit the HTML file although, as already noted, HTML authoring tools require some improvement [21].

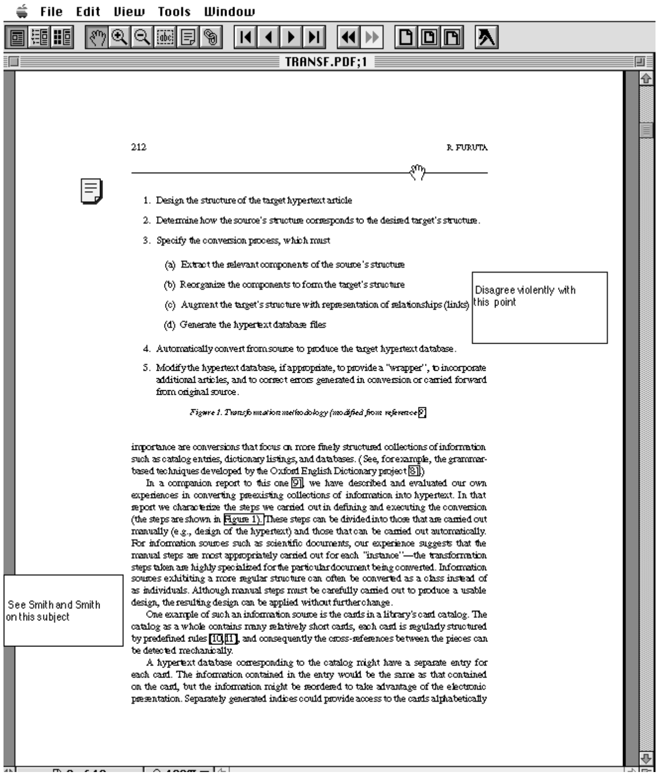
6.3 Annotations

The ability to annotate journal articles may be useful in various situations. For example, referees or editors may wish to include comments for document authors, and individual readers may wish to annotate documents as they read them.

This facility is provided in Acrobat by the **note** tool. This is positioned to the left of the links tool in the tool bar. Once the tool has been activated, the position of a note on a document page is chosen by a click of the mouse button. This opens a resizable note window as illustrated in Figure 10. Text can then be entered in the window which, on closing, appears as a small "note" icon.

In a similar fashion, Mosaic's Annotate window allows users to create annotations which are inlined as hypertext links at the end of the document. Annotations can be edited and/or deleted at any time. Currently, all annotations are personal; they are stored in the user's local file-system and are not accessible to other users. However, there are plans to support work-group and public annotations. The Annotate feature provides some flexibility. The author and title fields of the annotations may be customised, other files may be included and the annotation may be moved from its default position at the bottom of the document. If used on a suitable platform, audio annotations are possible.

Again there is no equivalent built-in feature of Guide, but annotations may be implemented in a number of ways. As illustrated in Figure 11, the prototype allows the user to insert easily recognisable annotation buttons and to step through the annotations in a document. The +Annotation menu item inserts an annotation button, and Find annotation moves to the first or next annotation in the document. Guide allows the user to edit the text of the document freely and to save the updated version, but the annotation mechanism in the prototype encourages a disciplined way of doing this. It is possible to save items in different contexts in different 'layers', so annotations could be saved separately to the main view.



212 R. FURUTA

1. Design the structure of the target hypertext article
2. Determine how the source's structure corresponds to the desired target's structure.
3. Specify the conversion process, which must
   (a) Extract the relevant components of the source's structure
   (b) Reorganize the components to form the target's structure
   (c) Augment the target's structure with representation of relationships (links)
   (d) Generate the hypertext database files
4. Automatically convert from source to produce the target hypertext database.
5. Modify the hypertext database, if appropriate, to provide a "wrapper", to incorporate additional articles, and to correct errors generated in conversion or carried forward from original source.

Figure 1. Transformation methodology (modified from reference 9)

Disagree violently with this point

importance are conversions that focus on more finely structured collections of information such as catalog entries, dictionary listings, and databases. (See, for example, the grammar-based techniques developed by the Oxford English Dictionary project [8].)

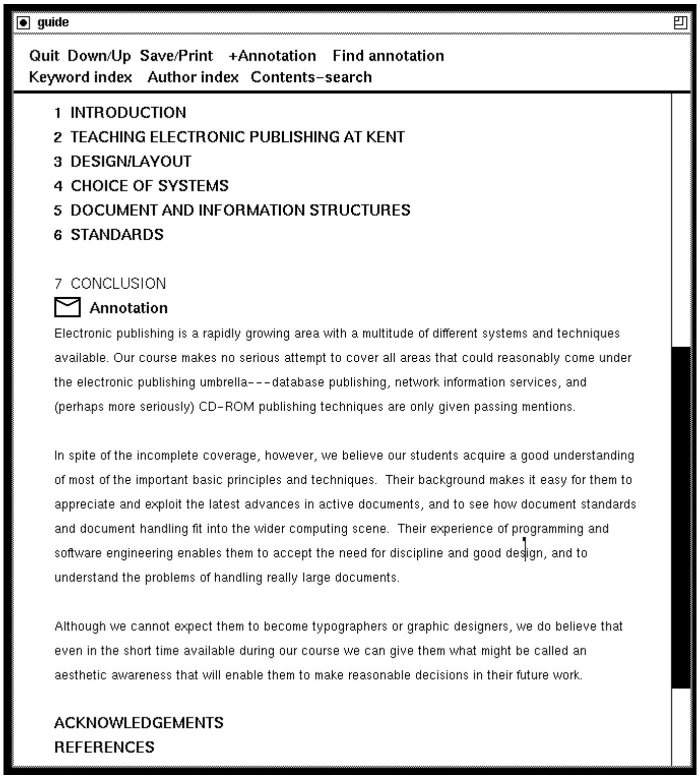
In a companion report to this one [9], we have described and evaluated our own experiences in converting preexisting collections of information into hypertext. In that report we characterize the steps we carried out in defining and executing the conversion (the steps are shown in Figure 1). These steps can be divided into those that are carried out manually (e.g., design of the hypertext) and those that can be carried out automatically. For information sources such as scientific documents, our experience suggests that the manual steps are most appropriately carried out for each "instance"—the transformation steps taken are highly specialized for the particular document being converted. Information sources exhibiting a more regular structure can often be converted as a class instead of as individuals. Although manual steps must be carefully carried out to produce a usable design, the resulting design can be applied without further change.

See Smith and Smith on this subject

One example of such an information source is the cards in a library's card catalog. The catalog as a whole contains many relatively short cards, each card is regularly structured by predefined rules [10,11], and consequently the cross-references between the pieces can be detected mechanically.

A hypertext database corresponding to the catalog might have a separate entry for each card. The information contained in the entry would be the same as that contained on the card, but the information might be reordered to take advantage of the electronic presentation. Separately generated indices could provide access to the cards alphabetically

Figure 10: The Exchange Note Tool



1 INTRODUCTION
2 TEACHING ELECTRONIC PUBLISHING AT KENT
3 DESIGN/LAYOUT
4 CHOICE OF SYSTEMS
5 DOCUMENT AND INFORMATION STRUCTURES
6 STANDARDS

7 CONCLUSION

✉ **Annotation**

Electronic publishing is a rapidly growing area with a multitude of different systems and techniques available. Our course makes no serious attempt to cover all areas that could reasonably come under the electronic publishing umbrella–––database publishing, network information services, and (perhaps more seriously) CD–ROM publishing techniques are only given passing mentions.

In spite of the incomplete coverage, however, we believe our students acquire a good understanding of most of the important basic principles and techniques. Their background makes it easy for them to appreciate and exploit the latest advances in active documents, and to see how document standards and document handling fit into the wider computing scene. Their experience of programming and software engineering enables them to accept the need for discipline and good design, and to understand the problems of handling really large documents.

Although we cannot expect them to become typographers or graphic designers, we do believe that even in the short time available during our course we can give them what might be called an aesthetic awareness that will enable them to make reasonable decisions in their future work.

ACKNOWLEDGEMENTS
REFERENCES

Figure 11: A Guide document with annotations

7 Conclusions

Paper-based journals offer several advantages over current electronic journals; their conventions are familiar, they make good use of human reading skills, they are easily portable, and they can be read in the bath. Thus, if electronic journals are to be accepted by users, they will need to add value to the journal concept beyond that currently offered by the paper-based journal.

As demonstrated by the CAJUN and MUSE projects and by the Guide prototype, all three systems have the potential to act as a vehicle for electronic journals. Each has particular strengths and weaknesses.

Of the three systems, the facilities and display provided by Acrobat most directly relate to the paper equivalent. Acrobat retains the concept of a page and, apart from any superimposed hypertext features, its layout is identical to the paper printout. These similarities are emphasised in the CAJUN project; components such as editorial information, indexes and articles are in the same relative position as they appear in hard-copy. These conventions help readers make the transition from paper to electronic format.

However, the introduction of electronic journals provides the opportunity to introduce new conventions. The MUSE project moves further away from the conventions of the paper copy. For example, there is no concept of a page and no explicit keyword index. Instead of the latter, search facilities are used to provide a more flexible method of indexing. This reduces the potential work required in preparing a journal issue.

The workload involved in preparing articles for distribution is a key factor in determining what facilities can practically be provided. Another example is the use of Acrobat bookmarks. Pre-setting these may take a considerable time. Features which are developed by the system automatically, such as Mosaic window histories, may be more practical.

Exploration of Guide as a vehicle for electronic journal provision is at an earlier stage than is the study of Acrobat and Mosaic software. However, it is already clear that, for this application, Guide is more flexible in several respects. In particular, its hypertext features are more sophisticated. As a result, it is possible to customise Guide applications so that they are appropriate for use in electronic journal provision. This provides an opportunity to improve on the methods currently employed to browse and study paper-based journals.

Mosaic is designed to facilitate navigation and searching across documents. In contrast, Acrobat Version 1 concentrates on the provision of features to aid navigation within a single document. Guide is sufficiently flexible to be of use in inter or intra-document navigation and study. For efficient provision of electronic journals, this flexibility may be vital. At different points in the process of accessing, browsing and studying electronic journal articles, the same system features may be required to operate on local and remote files. This is illustrated in the difficulty of providing an adequate browsing capability that can replicate the manual operation in speed and flexibility.

An area which requires further consideration is the saving and printing of journal articles. This is of particular relevance as journal articles are likely to be accessed or sent from a distance, possibly at frequent intervals and the user may well require a paper copy. Of the three systems, Acrobat provides the highest quality output. However, it is more limited in that it is not possible to edit the text of Acrobat files. It could be argued that this is an advantage from the point of view of the original authors and that the facility to annotation papers is all that should be provided to a reader.

One of the main drawbacks to all three systems is that, as yet, documents do not contain an embedded viewer. The reader needs access to Acrobat Exchange or Reader software or requires the appropriate Mosaic or Guide setup on site.

Features in different systems need not be identical. After all, journals are not homogeneous in layout or function; different journals may require different access, browse and study facilities. Whichever systems are used, however, general conventions need to be developed for the electronic presentation of journals and their constituent articles. This paper has illustrated how some of these conventions may be translation into relevant features in Acrobat, Mosaic and Guide.

Acknowledgments

Thanks are due to colleagues at the University of Nottingham, particularly David Brailsford, Steve Probets and Phil Smith, for their continuing help and advice. Thanks also to Peter Brown of the University of Kent for his advice on the Guide system prototype.

References

[1] M. Andreessen. Getting Started with NCSA Mosaic. Technical report, National Center for Supercomputing Applications, Champaign IL, USA, May 1993.

[2] M. Andreessen. Ncsa mosaic technical summary. Technical report, National Center for Supercomputing Applications, Champaign IL, USA, May 1993.

[3] M. Andreessen. NCSA Mosaic Technical Summary 2.1. Technical report, National Center for Supercomputing Applications, 605 E. Springfield, Champaign IL 61820, May 1993.

[4] T. Berners-Lee, R. Cailliau, J. Groff, and B. Pollermann. World-wide web: The information universe. *Electronic Networking: Research, Applications and Policy*, 1(2), Spring 1992.

[5] T. Berners-Lee and D. Connolly. Hypertext markup language: A representation of textual information and metainformation for retrieval interchange. Technical report, Internet Working Draft, July 1993.

[6] P.J. Brown. A hypertext system for UNIX. *Computing Systems*, 2(1):37–53, 1989.

[7] P.J. Brown. Using logical objects to control hypertext appearance. *Electronic Publishing – Origination, Dissemination and Design*, 4(2):109–118, June 1991.

[8] P.J. Brown. Adding networking to hypertext: Can it be done transparently? *ACM European Conference on Hypermedia Technology, Edinburgh*, September 1994.

[9] P.J. Brown. The format of unix guide source files. Technical report, Computing Laboratory, University of Kent, 1994.

[10] University of Nottingham Electronic Publishing Research Group, Dept of Computer Science. Sample journal papers in Acrobat/PDF form over the Internet: Update no 7 from the CAJUN project, 12 October 1994.

[11] D. Evans, S. Probets, P. Sutton, S. Benford, D. Brailsford, and L. Harrison. The CAJUN Project First Year Report. Technical report, Dept of Computer Science, University of Nottingham, 1993.

[12] C.F. Goldfarb. *The SGML handbook.* Clarendon Press, Oxford, 1990.

[13] Adobe Systems Incorporated. *PostScript Language Reference Manual.* Addison Wesley, Reading, Massachusetts, 2nd edition, December 1990.

[14] Adobe Systems Incorporated. Adobe acrobat products and technology : An overview, November 1992.

[15] Adobe Systems Incorporated. *Portable Document Format Reference Manual.* Addison Wesley, Reading, Massachusetts, June 1993.

[16] K. Lewis. Designing the HCI for a graphical knowledge tree editor: a case study in user-centred design. *ICL Technical Journal*, 7(3):554–564, 1991.

[17] J. Maddox. Electronic journals are already here. *Nature*, 365, 21 October 1993.

[18] S. Pathak. Electronic Publication at Johns Hopkins: Project MUSE. *EJournal*, 4(2), June 1994.

[19] D. Pullinger, February 1994. Informal comments at SuperJournal Project Presentation, Imperial College, London, UK.

[20] G.W. Rouse. Locator - an application of knowledge engineering to ICL's customer service. *ICL Technical Journal*, 7(3):546–553, 1991.

[21] B. Rousseau and M. Ruggier. Writing documents for paper and WWW. *Advance Proceedings of the First International WWW Conference, Geneva*, pages 169–178, 1994.

[22] P.N. Smith, D. Brailsford, D. Evans, L. Harrison, S. Probets, and P. Sutton. Journal publishing with Acrobat: the CAJUN project. *Electronic Publishing – Origination, Dissemination and Design*, 6(4), 1994.

[23] P. Waddell. The Potential for Electronic Journals in UK Academia. In *Libraries and IT: Working Papers of the Information technology Sub-Committee of the HEFCs' Libraries Review.* UKOLN: The Office for Library and Information Networking, University of Bath, UK, 1993.