Electronic Journals Using Guide

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

Abstract

This paper considers the use of the Guide hypertext system for electronic journal provision. The feasibility of providing appropriate tools for journal users with Guide is illustrated by discussing features developed for indexing, searching, annotating, saving and printing journal components. The use of Guide's logical objects or *contexts* is central to the development of a useful interface for browsing and reading journals. A prototype Guide electronic journal incorporating these tools is described.

Keywords

Electronic journals, hypertext, contexts, customised output

1 Introduction

This paper discusses the potential of the Guide hypertext system [5] as an appropriate vehicle for electronic journal provision.

Section 2 briefly describes some of the systems that have been or are currently being used as vehicles for electronic journals, including a previous electronic journal project using Guide. This is followed in Sections 3 and 4 by introductions to the Guide system and to the Guide prototype electronic journal developed by the authors. Sections 5 and 6 describe two features of Guide, buttons and contexts, which are particularly relevant to journal interface development. Sections 7 to 10 describe tools to aid journal browsing and reading that have been implemented or customised for the Guide journal application. These include tools for searching, annotating, saving and printing journal components.

2 Electronic Journal Formats and Vehicles

Until recently, most electronic journals and articles had to be restricted to the lowest common denominator formats of ASCII or bitmaps. Vehicles for more sophisticated formats were not widely available to users. Both ASCII and bitmaps remain popular formats for widely distributed material. ASCII is often used for list-server-based journals and articles, in which documents are delivered to subscribers via email. Both the ADONIS [15] and TULIP [17] projects have employed scanned bit-mapped images for document delivery.

LaTeX and PostScript are becoming the formats of choice for many electronic journals in the science disciplines. Their use enables the high quality output necessary for the presentation of, for example, mathematical equations. The peer-reviewed Chicago Journal of Theoretical Computer Science, for instance, is available via the Internet in both LaTeX and PostScript form [1]. However, whereas PostScript-based laser printers are widely available, the use of LaTeX is generally confined to the scientific community, particularly the mathematical sciences and computer science communities.

An early experiment in electronic journal provision was the development of HyperBIT, a Macintosh Guide version of the journal Behaviour and Information Technology (BIT) [4]. This development at Loughborough University was part of Project QUARTET, which aimed 'to investigate the implications of information technology for the scholarly communication process' [8]. HyperBIT incorporated a HyperCard-based front-end for browsing journal contents. The journal articles were stored as individual Guide documents. HyperBIT was 'the world's first hypertext electronic journal' and illustrated Guide's potential as a vehicle for such publications [13].

With the widening availability of World-Wide Web browsers such as Mosaic and Netscape, HTML-based[1] [14] electronic journals are proliferating. An example of one of the increasing number of peer-reviewed electronic journals to use HTML is the quarterly Journal of Computer-Mediated Communication [2], made available over the web as an HTML hypermedia document, but also accessible as plain text through gopher.

Several major electronic journal projects employ SGML[2] [9]. Among these, the OCLC's Electronic Journals Online (EJO) programme incorporates Guidon, a graphical user interface based on SGML [3].

The Acrobat suite [11], with its Portable Document Format (PDF) representation [12], is being used for electronic journal dissemination by the CAJUN (CD-ROM Acrobat Journals Using Networks) project[3] [16]. PDF may be regarded as a version of the PostScript page description language [10] with additional hypertext features for online presentation and browsing of documents.

A plethora of platforms and accompanying document formats is currently emerging. It remains to be seen which of these will be adopted, in the long term, as standards for electronic journal dissemination and presentation.

3 The Guide Hypertext System

Versions of the Guide hypertext system [5] have been developed to run on UNIX workstations, PCs and Macintosh computers[4]. Unless otherwise stated, references to Guide in this paper refer to UNIX Guide.

Guide is a true hypertext system designed to free documents from the constraints of paper. 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. The same material may appear at different points in the same document. This is of particular relevance to electronic journal browsing and is further explained in Section 6. 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. Text formatting changes dynamically if the window size changes.

The Guide system is flexible and readily programmable. It has been designed to exploit the UNIX environment and can easily interact with existing UNIX utilities. This interaction is exhibited in many of the features described in this paper.

Of particular importance to the electronic journal prototype is the *contexts* feature described in Section 5. Contexts, or logical objects, enable Guide documents to be described in terms of their logical structure rather than their appearance only [6].

---

[1] HyperText Markup Language

[2] Standard Generalised Markup Language

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

[4] The PC version of Guide is marketed by OWL Computers, as was the Macintosh version; the latter is no longer supported.

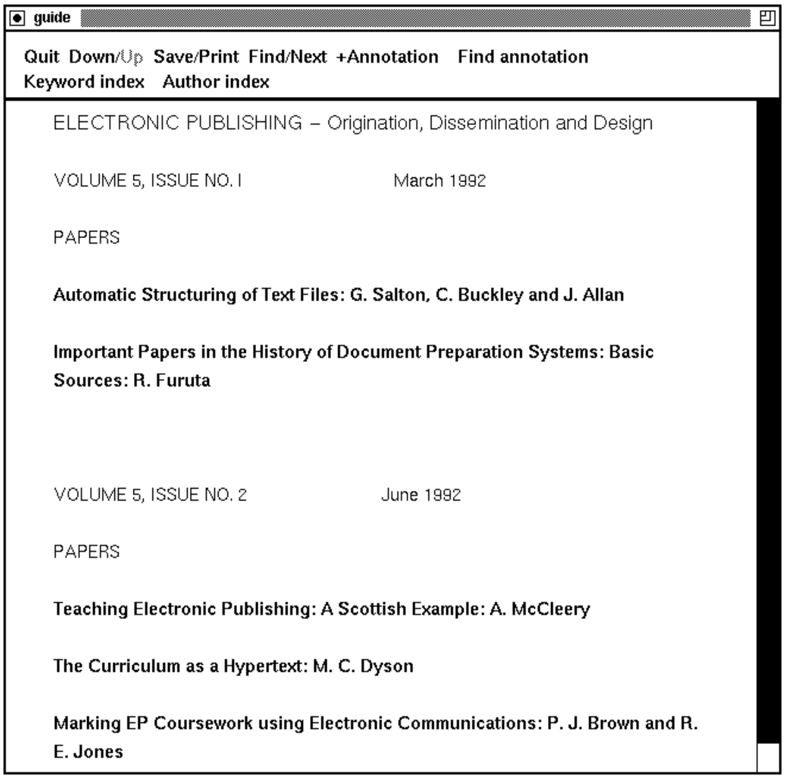

Figure 1: *EP-odd* contents viewed using the Guide prototype application

4 The Guide Electronic Journal Prototype

Recent additions to Guide's functionality, such as facilities for networking and the contexts concept mentioned above, have increased the potential of Guide as a suitable electronic journal vehicle. These facilities have been used in the development of a prototype Guide version of a Volume of the Wiley journal *Electronic Publishing—Origination, Dissemination and Design (EP-odd)*. This prototype is used throughout the paper to illustrate the ideas discussed. Figure 1 shows the initial view which corresponds to the normal contents summary for a volume of the journal. Clicking on an item provides progressively more information about the relevant article, as shown in Figure 2. In this way, users may see just the abstract, a list of subtitles, or the entire text.

The prototype application provides menu features for

- Saving customised versions of documents
- Context-based searching
- Adding and finding annotations
- Accessing keyword and author indexes.

Because the journal prototype is for use by journal readers, the authoring features available in Guide are not made accessible. A section of the expanded volume contents shown in Figure 2 is displayed again in Figure 3, this time viewed using the Author mode available under the full Guide system. In this mode, developers of the Guide prototype journal may add structure, such as contexts and buttons, to the journal content. The use of contexts and buttons in the prototype is described in more detail in the following sections.

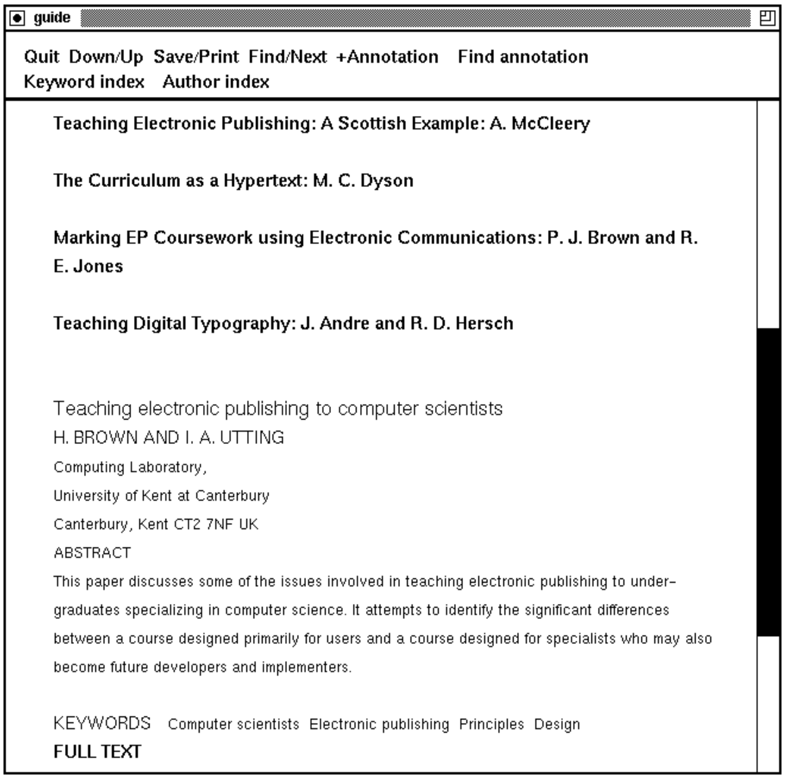

Figure 2: Partly expanded *EP-odd* contents viewed using the Guide prototype application

5 Contexts in Guide

Unlike many hypertext systems, Guide supports contexts or logical objects, which allow the logical document structure to be captured [6]. This may be compared to the logical structuring enabled by markup languages such as SGML [9].

An example of the use of contexts within Guide is to define the generic properties of appearance attributes such as text sizes and fonts. Instead of having to define the appearance of document components such as section headers whenever they appear, a section header context may be defined once and then used throughout the relevant documents. As Brown [6] points out, not only does this make systematic changes to document structure and appearance easier, it also allows the meaning of the document to be captured. In the prototype, this feature is used to implement intelligent searching and printing of the journal contents.

Although contexts may affect appearance on the screen, they are not explicitly visible to journal readers. In author mode, however, they are explicitly labelled, and Figure 3 illustrates how they are used in the journal prototype. Each occurrence of a context is delimited by curly brackets and identified by its name which appears just inside the left-hand bracket. The contexts in Figure 3 include `Front-matter`, `Title`, `Author-details` and `Author`.

Guide stores the names and properties of contexts in a *context table* which may be accessed and edited while in Author mode. Lines 4 and 5 of the menu at the top of Figure 3 list the contexts available in the prototype. Contexts can be nested to an arbitrary depth. For example, the contexts `Author` and `Institution` are nested inside the `Author-details` context.

If a property is not specified for a context, that property is inherited, for each instance of the context, from the containing context. Properties can also be set relative to properties of the containing context. For example, the `Title` context is given the property of being displayed in the same font as the surrounding text but several points larger. In addition to affecting appearance, properties may also affect the treatment of items during searching and printing as described in

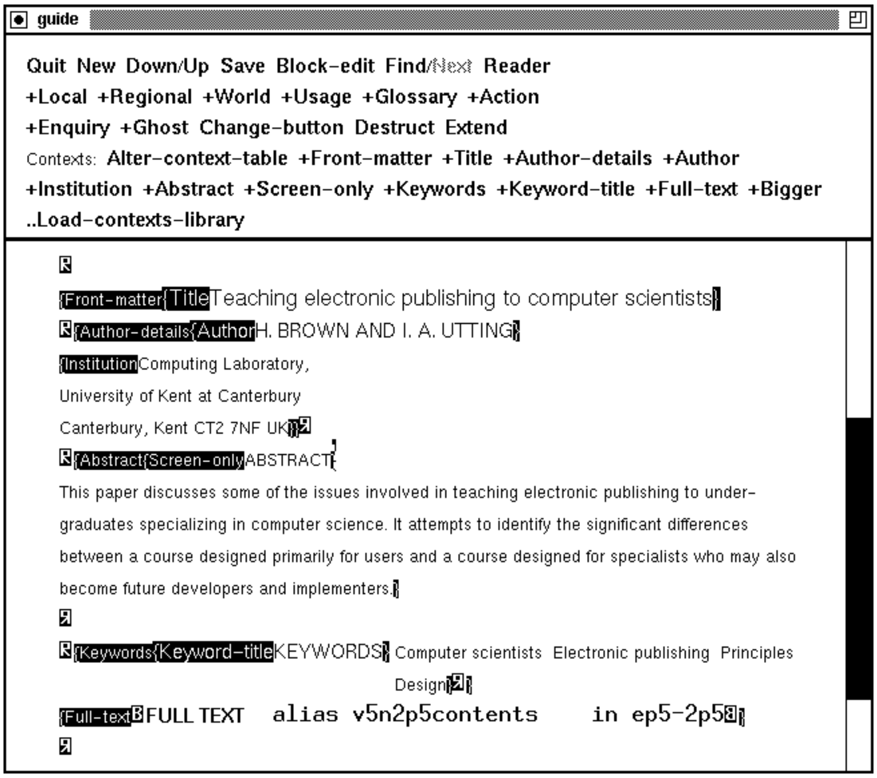

Figure 3: Partly expanded *EP-odd* contents viewed in Guide Author mode

Sections 8 and 10.

6 Buttons in Guide

Guide documents may incorporate various forms of button. These can be created, edited and deleted using the Guide Author mode; lines two and three of the menu in Figure 3 list the relevant operations.

There are three basic types of button: glossary, action and replace. When a glossary button is selected, its replacement appears in a separate view. Glossary buttons could be used in an electronic journal application to contain citation details or footnotes.

Action buttons exploit Guide's programmability. The choice of an action button may involve commands to the Unix shell, for example to start up a new application, or commands to Guide itself, for example to create a new button or load a file. The Guide menu is, in fact, a collection of action buttons.

Replace buttons are the commonly used ones. Selecting a replace button typically 'unfolds' the document scroll to reveal a piece of information or a picture in place of the button name. The replacement may be stored in the same file as the button name or in a different file. Alternatively, the replacement may be created dynamically, for example, by running UNIX programs directly from Guide. This means that Guide can be used as a front-end to other systems, such as information retrieval software.

Replace buttons are normally distinguished from the rest of the text by being presented in bold font or in a different colour. The words **FULL TEXT**, for example, at the bottom of Figure 2, represent a replace button. The same button may be seen in Author mode in Figure 3. The button and reference to its replacement are delimited by and . As may been seen, the button is enclosed within the context `Full-text`.

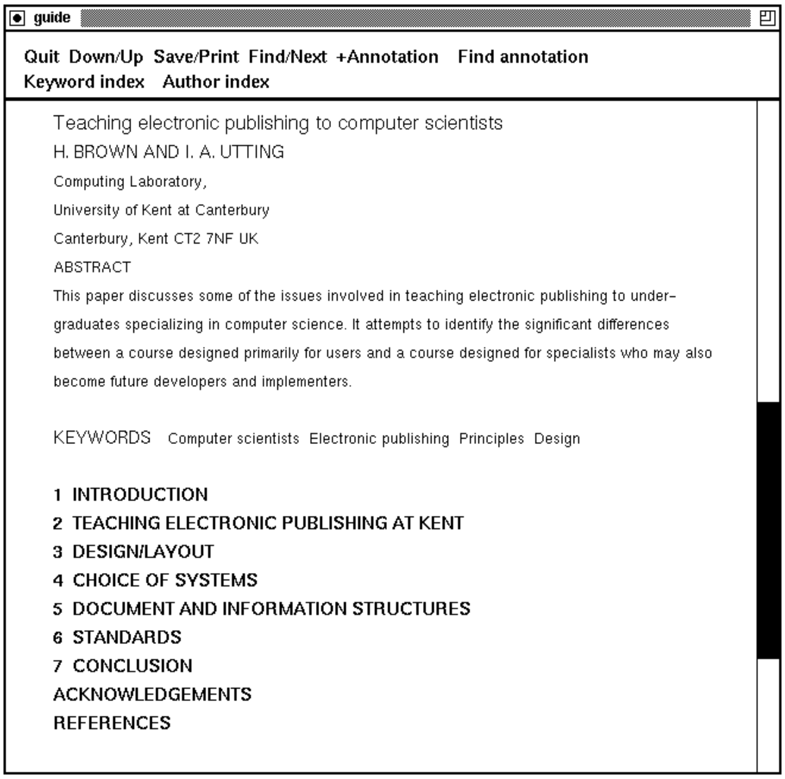

Figure 4: Result of clicking on words FULL TEXT

For a simple replace button, the normal visible button name (only) is enclosed between the button delimiters and the replacement is stored immediately after the button in the source file. However, *button name extensions* may be used to cater for more complex cases. These extensions are only visible in author mode. Two commonly used extensions are

- `alias <button name>` — This causes the replacement to be taken from another button of the same name. The use of `alias` allows a single replacement to 'appear' at different places in the document scroll.
- `in <name>` — This causes the search for the replacement to be made in the named file. It is normally used in conjunction with `alias`. A recent extension to Guide allows remote files to be searched as well as local ones [7].

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. In the case of the **FULL TEXT** button in Figure 3, the replacement is held within a button called `v5n2p5contents` in a separate file called `ep5-2p5`. The use of the `alias` extension enables the replacement, in this case the full text of the article, to be accessed via different buttons.

Clicking on the words **FULL TEXT** in the journal prototype results in the display of the article section headers as shown in Figure 4. Like Guide contexts, button definitions may be nested to provide a hierarchical structure. The section headers are also buttons, and clicking on one of these will display the relevant section of the article.

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. Navigation is also aided by the display of the current hierarchy of open buttons at any point. This is displayed by holding down the left-hand mouse button over the scroll bar on the right-hand side of the Guide window, as illustrated in

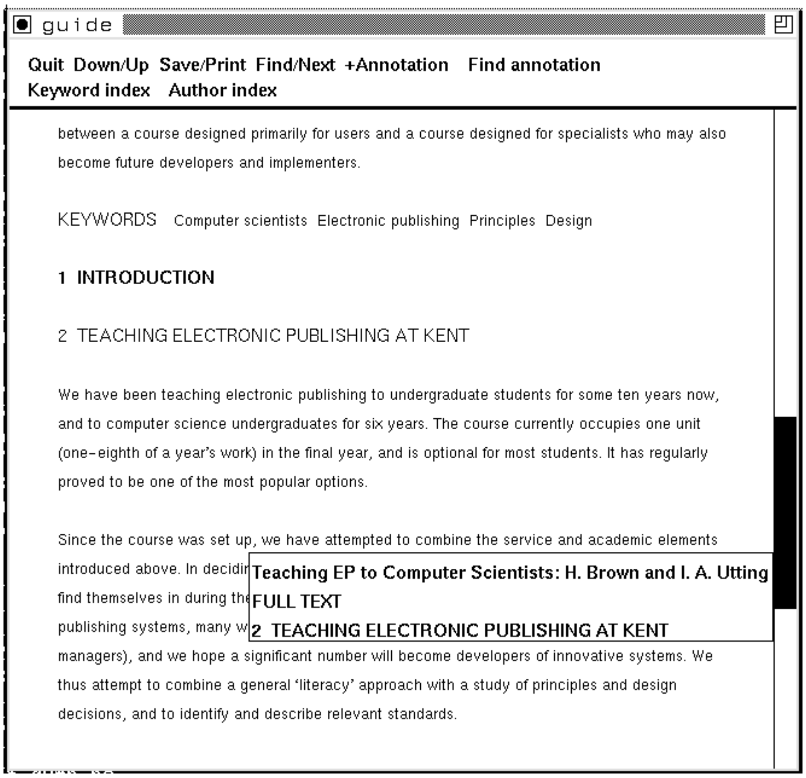

Figure 5: Scroll bar navigation aid

Figure 5. In this case, the contents of the navigation window indicate that the text currently being viewed is the result of unfolding buttons in the following order:

- Teaching Electronic Publishing to Computer Scientists: H. Brown and I. A. Utting
- FULL TEXT
- 2 TEACHING ELECTRONIC PUBLISHING AT KENT

7 Indexes

General conventions need to be developed for the electronic presentation of journal 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 sub-components 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; rather, the emphasis may be on knowing how to access them.

Although Guide has no built-in indexing features, its flexibility allows such features to be set up easily. In the prototype, we have chosen to replicate the original access methods from the paper version. Thus, in addition to access via the contents information, as shown in Figures 1 and 2, 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 lead to details of all the relevant articles in exactly the same form as via the main contents view. This approach maintains the notion of the

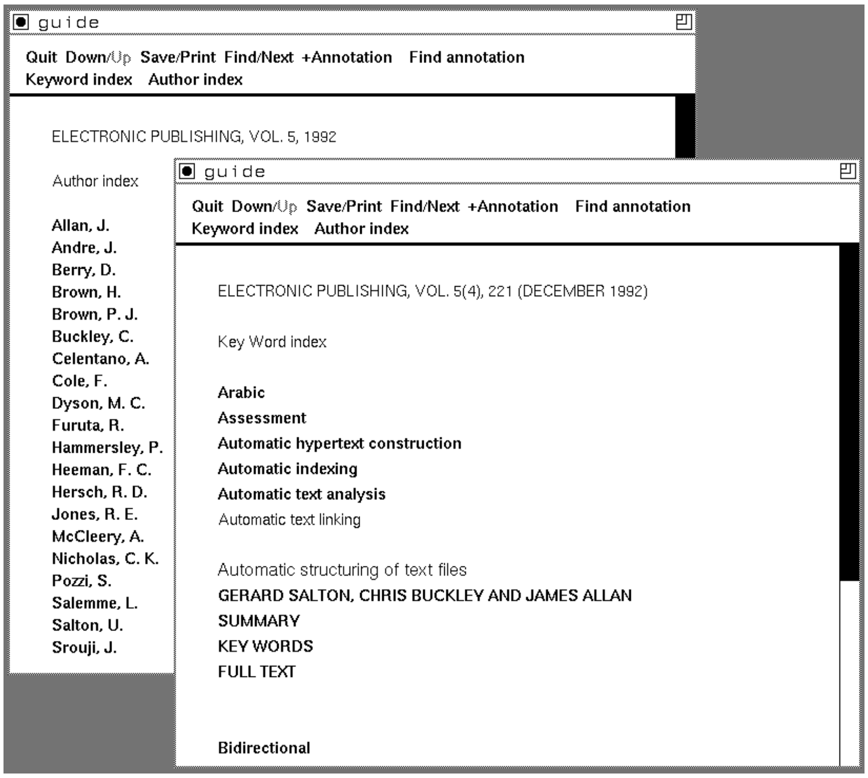

Figure 6: The result of selecting keyword and author indexes and then clicking on the keyword `Automatic text linking`

Guide document as a continuous scroll with sections hidden behind buttons. To demonstrate the flexibility of Guide, the keyword index appears in a separate window and the author index covers the original view. Figure 6 shows the result of selecting both indexes and then clicking on the keyword `Automatic text linking`.

Journal users all have their own particular approach to browsing and studying journals. As well as enabling access via author and keyword indexes, we intend to develop a feature to allow users to create their own indexes, either from scratch or by customising those automatically provided. As an example of the former, a user may wish to develop an index based on institutions, abstracts or subsection titles. Alternatively, the user may want to customise the author index so that it displays authors in chronological order of article publication. Access to these personalised indices will be via appropriate menu items. The use of contexts will be central to the operation of this feature.

8 Search and retrieval facilities

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 articles (title, author information, abstract and keywords).

The `Find` menu item, as seen in Figure 6, 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

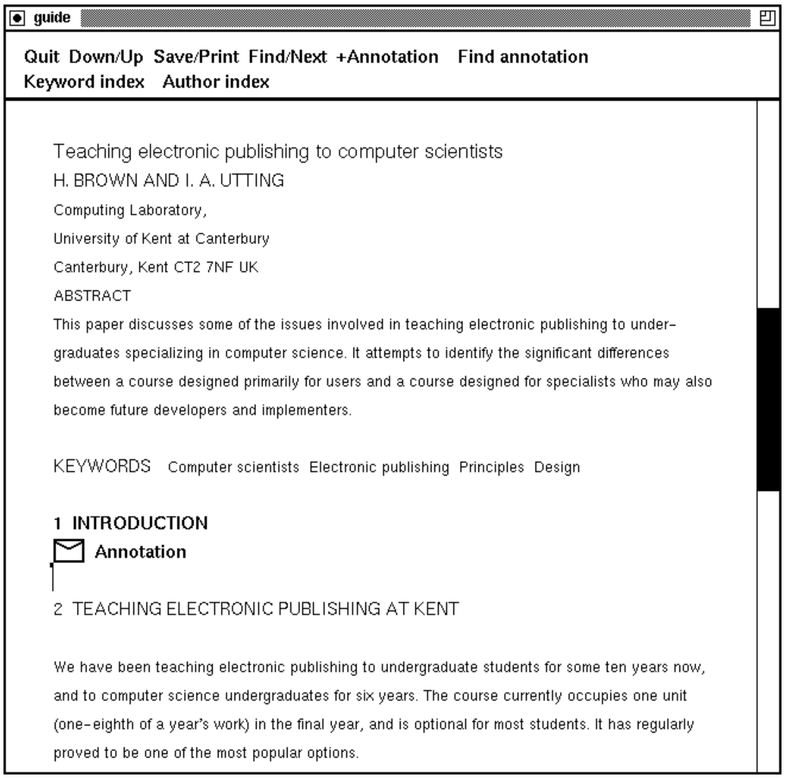

Figure 7: A Guide document with annotations

at the beginning of the scroll or from the current position. Further occurrences of strings may be found using the `Next` menu item. The find feature could be easily extended to allow limited searching within other specific contexts, for example, searching across author names, subtitles, keywords or annotations.

9 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.

A feature has been implemented in the prototype that allows the user to insert easily recognisable annotation buttons and to step through the annotations in a document. This is illustrated in Figure 7. Clicking on the `+Annotation` menu item launches a brief dialogue with the user in which the position of the required annotation is indicated and the text entered. Annotation buttons are indicated by the 'envelope' icon followed by the word 'Annotation' as shown in Figure 7. Clicking on the annotation button reveals the text of the annotation in the normal way.

`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.

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

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

ABSTRACT

This paper discusses some of the issues involved in teaching electronic publishing to under-graduates 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

Annotation

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 8: Portion of Customised output from `Print` option

10 Saving and Printing Documents

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.

Guide provides the user with various options for saving documents. In the prototype system, the `Save` menu item shown in Figure 7, has been tailored so that its choice results in the original Guide source file being saved along with any edits or annotations added by the user.

The context feature has been used in the `Print` option to specify actions to be performed on saving a document. In particular, context properties are set so that the relevant formatting commands (groff -ms macros) are added before and after each occurrence of a particular context. Other formatter markup, such as LaTeX, could be substituted for groff. The resulting marked-up file may then be passed through a groff formatter and the Postscript output from this process sent to a printer. A section of a sample resulting document is illustrated in Figure 8. An entire document can be seen in the Appendix.

As Figure 8 and the Appendix illustrate, unopened buttons are indicated by bold text. The content of opened articles is displayed as prescribed by the groff ms macros. Thus, article titles, author names, institutions and abstracts are centred. Article titles are also emboldened and author names are italicised. Annotations also appear; the annotation in Figure 8 appears as a button name as it had not been opened when the document was saved using the `Print` option.

The result of processing the majority of ms macros is solely to position and format the existing text. The abstract macro, `.AB`, is an exception in that its use results in the addition of text, the word ‘ABSTRACT’, to the output file. This means that the word ‘ABSTRACT’ which appears on the screen must be removed before the file is passed through the groff formatter. Here again, contexts are employed. Text within the `Screen-only` context seen in Figure 3 will appear on the screen version but will be filtered out when using the `Print` option.

11 Conclusions

The programmability of Guide makes it an ideal research tool for experimentation in the development of electronic journals. Guide provides a flexible means of representing electronic documents based on a continuous scroll metaphor. The particular strengths of Guide in the field of electronic journals lie in its programmability, its sophisticated hypertext features, provision of structure through contexts and its networking facilities.

One of the main drawbacks to Guide, and to most hypertext systems, is that documents do not contain an embedded viewer; users require the appropriate Guide setup on site. A further drawback is the limited graphics capabilities of Guide. Inline images are only possible in bit-map form. However, viewers may be easily spawned from Guide.

Despite the current proliferation of electronic journals, the future form and organisation of such publications is still wide open for debate. Their introduction provides the opportunity to introduce new conventions compared to those provided by paper-based journals. At this stage of electronic journal development, it is important to explore different paradigms and metaphors, to experiment with different vehicles and to evaluate the strengths and weaknesses of each, rather than to allow the paradigm of a few vehicles to dictate prematurely the future form of the electronic journal.

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 translated into relevant features in Guide.

Acknowledgments

Thanks are due to Peter Brown of the University of Kent for his advice on the Guide system prototype.

Appendix:
Document produced by passing output from `Print` action through a groff formatter

ELECTRONIC PUBLISHING - Origination, Dissemination and Design

VOLUME 5, ISSUE NO. 1 March 1992

PAPERS

Automatic Structuring of Text Files: G. Salton, C. Buckley and J. Allan

Important Papers in the History of Document Preparation Systems: Basic Sources: R. Furuta

VOLUME 5, ISSUE NO. 2 June 1992

PAPERS

Teaching electronic publishing: a Scottish example
A. McCLEERY
ABSTRACT

KEYWORDS Publishing Teaching Electronic publishing Desktop publishing Non-print publishing Simulation

FULL TEXT

The curriculum as a hypertext

MARY C. DYSON

Department of Typography & Graphic Communication
University of Reading
2 Earley Gate, Whiteknights
PO Box 239. Reading RG6 2AU, UK
ABSTRACT

KEY WORDS
FULL TEXT

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

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

ABSTRACT

This paper discusses some of the issues involved in teaching electronic publishing to under-graduates 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

Annotation

EP-ODDS AND ENDS

A Curriculum in Electronic Publishing: P. Hammersley

VOLUME 5, ISSUE NO. 3 September 1992

PAPERS

On the Interchangeability of SGML and ODA: C. K. Nicholas and L. A. Welsch

ALIVE: A Distributed Live-link Documentation System: S. Pozzi, A. Celentano and L. Salemme

Granularity in Structured Documents: F. C. Heeman

VOLUME 5, ISSUE NO. 4 December 1992

PAPERS

Arabic Formatting with ditroff/ffortid: J. Srouji and D. Berry

EPODDS AND ENDS

Editing Structured Documents - Problems and Solutions: F. Cole and H. Brown

References

[1] Chicago Journal of Theoretical Computer Science. For further information, contact the MIT Press at JOURNALS-INFO@MIT.EDU or Journals Advertising/Publicity, The MIT Press, 55 Hayward Street, Cambridge, MA 02142-1399, USA.

[2] The Journal of Computer Mediated Communication. Available online as http://shum.huji.ac.il/jcmc/jcmc.html or http://cwis.usc.edu/dept/annenberg/announce.html.

[3] OCLC Releases Guidon 3.0 for Its Electronic Journals Online Offering. Online libraries and microcomputers, Feb 1995. vol 13, no 2.

[4] Behaviour and Information Technology. Taylor and Francis.

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

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

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

[8] McKnight C. The Electronic Journal. *Library and Information Briefings*, Issue 44, May 1993.

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

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

[11] Adobe Systems Incorporated. Adobe Acrobat Products and Technology : An Overview, November 1992.

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

[13] C. McKnight, A. Dillon, and J. Richardson. *Hypertext in Context*. Cambridge University Press, Cambridge, 1991.

[14] NCSA. A beginner's guide to html. URL: http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html.

[15] S. Pilling. The ADONIS Experience: Some Views. *Journal of the United Kingdom Serials Group*, 7(3):249–252, November 1994.

[16] 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.

[17] K Willis, K Alexander, and B Warner. TULIP–The University Licensing Program: Experiences at the University of Michigan. *Serials review*, 20(3), 1994. See URL: http://www.elsevier.nl/TULIP/Menu.html for further details.