
Increasingly, text is being viewed not in printed form but on a computer screen. This book is therefore a timely one, as it concentrates on the ergonomic and human-factors issues underlying the design of electronic text.

It has ten chapters, a bibliography, a single appendix and a subject index. Following a short introductory scene-setting chapter, the author moves straight into a discussion of usability and the potential utility of ergonomics and human factors within this area. This is followed by a review of the experimental literature on reading from paper and from screens. Two basic strands are considered: outcome measures (speed, accuracy, fatigue, comprehension and preference) and process measures (eye movements, manipulation and navigational issues). The review is organized into three basic parts, covering reported differences between paper and screen, analysis of differences in terms of physical, perceptual and cognitive processes, and issues relating to text and task variables.

In chapter 4, the author examines the value of existing human-factors literature to electronic document designers. The problems of applying this knowledge to the design and evaluation of an electronic-text system are then illustrated by means of a case study (an interactive document retrieval system based on a CD-ROM called ADONIS).

Chapter 5 delves into the problems of classifying texts by means of repertory grid analysis using the FOCUS program, and the implications of the results for electronic text design are then discussed. Chapter 6 explores in more depth readers’ interactions with two particular types of text – academic journals and software manuals. The methodology used in the investigation is based on the WWH (Why, What and How) approach to document usage. The chapter concludes with a discussion of the design implications of the research findings.

The material presented in chapter 7 is concerned with the literature dealing with readers’ impressions of structure and shape in information space (based on schemata theory), and how this might relate to navigational problems within electronic information. Two simple experiments designed to investigate document structure are then briefly described.

In chapter 8, the author proposes a framework which represents the ergonomic factors involved in using a text, and suggests the variables to consider when designing an electronic document. The framework consists of four interactive elements which reflect the issues dominating a reader’s attention at various stages in a reading process. The components of the framework include the task model, the information model, a set of manipulation skills and facilities, and a serial reading processor.

The validity and utility of the proposed framework are explored in chapter 9. This describes two experiments involving readers’ use of an ordinary text (on paper) and electronic texts (both linear text and hypertexts) for information retrieval. Various tools were employed to produce the electronic texts (HyperCard, TIES, a word processor and GUIDE). The experiments were conducted in a usability laboratory, and verbal protocol techniques were used to analyse readers’ behaviour.

In the final chapter of the book the author summarizes and discusses what has previously been said, and then turns his attention to potential applications of the framework – particularly for the development of systems based on the use of electronic text.

I found this book quite useful. It draws together much of the documented scientific knowledge on reading processes, and reviews its worth in the context of comparing paper-based and screen-based visual interaction with text. Although there are many more avenues yet to explore, the ideas contained here provide some useful starting points on which to base future research.

Philip Barker, University of Teesside


It is somewhat difficult to choose the most useful way to summarize a book of selected contributions from a conference. The reader
Reviews

does not know if the contributions selected (in this case, 21) are meant to represent as faithfully as possible the range of interest areas and contributions presented at the conference, or if they are chosen on a basis of attempted balance among institutions or regions represented among the participants, or some other criterion such as a mix of 'old' and 'new' contributors. Perhaps the standard is purely scientific: the editors' opinions of the most valuable entries. But because of this uncertainty, the reader cannot fairly approach the book as an inventory of British activities with computers in education, nor as conference proceedings. Thus, how to proceed? By treating the book as a book, and thus expecting that the title is an organizer for the content? This approach does not help very much either, in that the reader entering the book through interest in its title gets very little sense of how much computer-assisted learning has gone 'into the mainstream' of UK education after reading the entries. This said, perhaps the only way to address the book is as a hard-covered journal, with a collection of 21 articles offered without any overt editorial synthesis.

That rather long-winded introduction is by way of explaining my choice of procedure for this review. I will give a general comment about the topics and settings addressed by the articles, then note some that I have found to be particularly useful.

Thus the overview of contents: 17 of the 21 articles are from UK author teams, one a team of mixed UK and non-UK authors, and three from other countries (Canada, Germany and Finland). As to content and setting, my analysis is that:

- three deal with software development methodology;
- ten could be described as 'specific-case' studies situated at the university level,
- seven of which I would most generally categorize as case studies of particular prototype packages, including descriptions of the theory of their designs and development processes and evaluations based on the author/designers' experiences with the in-house use of the package,
- one of which is a more cognitively-oriented study of an experiment involving an existing package (a concept mapping tool),
- and two of which relate, in different ways, to distributed learning (OU students' experiences with home computer use in a particular OU course, and experiences at Lancaster with CMC);
- four relate to experiences with children as they use a particular software environment;
- one, from Finland, studies a group of lower-secondary students and a LEGOLogo experience;
- three, including one from Canada, look at groups of primary students as they use either modelling software, or a prototype developed by the authors, or a CD-ROM 'talking storybook';
- two describe case-study type observations about groups of students in initial teacher education at two institutions (articles 17 and 19 - these articles were so similar in style and substance that I looked back twice to see if I was not reading the same article a second time); and
- two describe, from large-scale and longitudinal perspectives, implementation experiences about computers in education.

The latter two in the categorization I offer above are, in my opinion, the two articles which most substantially relate to the title of this book. They are the two I solidly recommend to anyone interested in what happens when large-scale initiatives try very hard to stimulate computer use in schools. They are also two of the three articles from the collection I have already cited in my own writings, and will continue to use as valuable references, not only because they are based on broadscale samples of experience, but also because of the value of the insights they offer.

The first of these two, whose title does not do much to suggest its contents, is by Alan Brown ('Processes to support the use of information technology to enhance learning') and is based on an evaluation of the UK Information Technology Teacher Training Development Programme (1988-1992). It presents many rich and useful insights from seven projects focusing on the management of IT use by teachers, and ways teachers use IT to enhance learning in the classroom. The second, by John Gardner et al. ('Learning with portable computers'), is a fascinating study of a broadscale attempt,
involving nine schools and 235 pupils, to give students unlimited access to personal computers (at home and at school) over an entire school year, then to investigate the impact of this access on their performance in core school subjects. The article, whose findings deserve careful study by anyone interested in 'the mainstream', again points out the critical influence of the teacher in what happens with computers and learning.

The third article I found particularly useful is by Mike Aston and Bob Dolden ('Logiciel sans frontières'), again a title which will not help the reader to identify its contents if seeing only a list of titles. This article is of less general interest than the ones by Brown and Gardner et al, but of particular value to those involved in software development from a broad perspective. The article summarizes some of the experiences from a number of European-level projects involving cross-national educational software portability.

As for the rest of the articles, well, I certainly can say they are well-written, well-edited, and a number of them I have noted for myself because of their relation to my various research interests. But the reader will most benefit from them by seeing them in wider contexts. Studies of the generic type: 'What a certain group of our students did with our prototype and what we have learned from the experience', for example, are most valuable when one can place such observations into the context of other studies involving similar cases in order to see where trends and insights transcend the particular case under description. For the reader interested in CMC, I quickly went to the article by Steeples et al on the topic, and am pleased to add it to my CMC Insights and Experiences file.

One accepts a journal as a collection of articles, and the articles included here are quite interesting. Nonetheless, one would like a book, destined to live for years on library shelves, to go one step beyond the collection level to bring some sort of coherence in the articles for the reader who in the future seeks out the book because of its title. One hopes that those readers will find the articles by Brown and by Gardner and his colleagues, if they want a reflective summary of a wide set of experiences from the field, from mainstream practice.

Betty Collis, University of Twente, Netherlands

Note
The content of this book also forms Volume 22 (Issues 1 and 2) of the journal Computers and Education.


This book is aimed at those wishing to develop HyperCard applications and who already have some programming experience (for example, in a conventional third-generation language such as Pascal or BASIC). The authors suggest that their work is suitable for a number of target reader groups: novice programmers, expert programmers, students, advanced students, and HyperCard developers, and individual chapters are identified in the Preface as being of particular relevance to each target reader group.

A floppy disk containing HyperCard stacks accompanies the book, and complements it by providing the HyperTalk code discussed within it. In this way, readers with a Macintosh and HyperCard are able to run the programs and, more importantly, cut and paste both objects and code for use in their own programs (known as stacks in HyperCard).

There are a number of books about HyperCard, but this one is quite novel in its approach. It goes straight into HyperTalk (without lengthy description of the HyperCard interface) and combines this with the pedagogic strategy of teaching by example. Throughout the book, example code is used to introduce new HyperCard and HyperTalk topics to readers. A key benefit of this topic-based approach is that it presents material in problem-oriented ways. Readers are thus able to discover hints and tips while learning about programming in HyperTalk. They are also presented with solutions to some of the practical problems that might be encountered when they come to develop their own HyperCard stacks.

There are fifteen chapters, all oriented towards practical use. The first provides an immediate introduction to programming in HyperTalk. This is followed by two chapters on the basics of working with HyperCard and HyperCard objects. The fourth chapter then describes the basic commands and control structures used in HyperTalk programming.

Chapters 5 to 11 adopt the topic-based