
MICROCOSM: An Open Model for Hypermedia With Dynamic Linking

Andrew M. FOUNTAIN, Wendy HALL, Ian HEATH and Hugh C. DAVIS

*Department of Electronics and Computer Science
University of Southampton
Southampton SO9 5NH*

ABSTRACT : There are currently a number of commercially available hypertext and hypermedia systems, of varying levels of sophistication and usability, but there are still many problems to be resolved in the design of such systems. In this paper, we itemise some of the major problems that we have identified as possibly causing a barrier to the growth and development of hypermedia applications outside the research community. A model of an open hypermedia architecture with dynamic linking features is proposed that moves some way to resolving these problems, and the first implementation of the system, Microcosm, is presented and discussed.

KEY WORDS : hypertext, multimedia, dynamic linking

1 Introduction

The terms hypertext and hypermedia are now so well known that definitions are unnecessary. Advances in computer power and performance have permitted the visionary concepts put forward by Bush [1945], Engelbart [1963] and Nelson [1967] to become a practical reality and we are now in a position to explore the implications, both positive and negative, of using hypertext systems for the management of information. The need for us to be able to control and utilise the phenomenal amounts of information that will be available to us through advances in technology, is the driving force behind the rapid pace of developments in hypertext and hypermedia systems. Much of the textual information that is available in the world today is stored or originally created in electronic format can be revised, edited and distributed easily as text files in a computer system. When we add to this the potential afforded by developments in optical disc technology for the storage of vast amounts of information from different media, we see the dawning of a new age of information processing.

Traditional ways of presenting and distributing information in a linear fashion are no longer sufficient, and conventional database technology does not permit the wide range of functions required to fully exploit the potential of the

new multimedia information systems. Neither of these strategies will become redundant but new systems must be developed that combine support for structured text, the handling of large volumes of data, the ability to mix media, and the freedom for the reader to switch seamlessly between the mixed media documents in the system [Rahtz *et al.* 1989a]. Hypertext and hypermedia techniques seem to offer much potential in this context ([Rahtz *et al.* 1989], [Yankelovich *et al.* 1988], [Shneiderman *et al.* 1989]), but whilst providing solutions to many information processing problems, they naturally give rise to further sets of usability problems that have yet to be resolved ([Halasz 1988], [Nielsen 1990a]).

The use of hypertext and hypermedia systems is still largely confined to the research community. This is partly because of the limitations of commercially available systems and partly because of the tremendous effort required to create and maintain a hypertext system. These issues are compounded by the fact that currently available hypertext packages are basically closed systems, so that if material is created in one system it is very difficult to integrate it with material created in another system. We believe that this is a major barrier to the growth and development of hypertext and hypermedia applications outside the research community. This paper describes the design and implementation of a dynamic model of hypermedia that moves some way towards resolving these problems.

2 Current Hypermedia Systems

Nielsen's new book [Nielsen 1990b] gives a very full account of the nature of hypertext and hypermedia and describes the major systems that are currently available. The two most widely used, certainly in the U.K., are Apple's HyperCard and OWL's Guide. It is a debatable point whether HyperCard can actually be described as a hypertext system. Its strength is in its rapid prototyping capabilities and it can certainly be used to simulate a hypertext environment through the means of its scripting language, HyperTalk. On the other hand, Guide developed from a research base at the University of Kent [Brown 1986], and was designed as a tool for reading documents on a computer. Guide has many well-defined and well-designed hypertext features which make it very easy to use. Version 2 of Guide had only limited extensibility but version 3, which has recently been released, incorporates a full scripting language.

Both Guide and HyperCard have become successful commercially because they are easy to use and are readily available on the most widely used personal computer workstations, namely the IBM pc (or compatibles) and the Apple Macintosh. HyperCard is of course only available for the Macintosh range of computers but Apple's marketing ploy of giving a copy of HyperCard away free

with every Macintosh has served to make HyperCard phenomenally successful as an application development environment. It is also possible to extend both Guide and HyperCard to incorporate multimedia information (such as image, sound and video) so that they can be classified as hypermedia systems. However, neither system provides a means for cataloguing or manipulating information about links independently from the documents in which they are activated. This means it is very difficult to provide summary information about links for maintaining, updating and enhancing the system, or to provide graphical browsing facilities and other sophisticated navigation tools. Guide overcomes navigation problems by controlling the types of buttons that the author can use and by emphasising the use of 'replacement' buttons in a scrolling text. However, HyperCard encourages a 'goto' approach to the creation of button and links and its very flexibility can be its downfall as far as controlling the development of the application is concerned.

Other well-known hypertext systems, such as Intermedia [Yankelovich *et al.* 1988], Notecards [Halasz 1988] and Hyperties [Shneiderman 1987], have also developed out of large research projects and contain many highly sophisticated and well-researched hypertext features and navigation tools. They all embody the concept of a database of nodes and links but with very different models of design. The Intermedia project at Brown University is probably the most well-known and most often referenced hypertext research project. It was designed for educational use at the university level and has been used to teach several courses in subjects such as biology and English literature. Hyperties is well-known as the system in which the hypertext version of the July'88 issue of the CACM was issued, and for the computer-based version of the book *Hypertext Hands On* [Shneiderman & Kearsley 1989]. It is available for the IBM pc and has been used extensively for museum applications [Shneiderman *et al.* 1989]. Intermedia and Notecards both require high-powered Unix workstations and this may limit their use outside of the research environment in the immediate future.

3 Problems with Current Systems

We consider that there are four problems common to many current hypertext systems, which act as possible barriers to the creation, extension and integration of hypertext document systems.

Authoring effort required. The large effort required to insert links into documents is seen as one of the main limitations of hypertext systems. Documents are becoming available in computer readable form far faster than they can be converted to hypertext. The user of a hypertext system may often

feel frustration that they are limited to the documents that have been painstakingly converted to run under their system.

Hypertext systems are closed. Hypertext systems are usually run as stand alone programs and will not communicate bi-directionally with other software packages. Often it is possible to add drivers to extend the system, but these connections are not usually bi-directional in the sense that external programs can access the hypertext linking mechanisms. It is seen as important to be able to integrate a hypertext system into the user's whole environment and this requires an open system. When multimedia documents are used, it is often necessary to add support for extra video/graphics formats. A system that has been designed to be open can provide for this sort of extensibility.

Proprietary document formats. In general it is not possible to take documents produced in one system (e.g. Guide) and use them in another (e.g. Hypercard). Document formats are sometimes closely guarded secrets. This exacerbates the authoring problem above, and makes it difficult to write conversion programs from one format into another.

Problems with read-only media. Many hypertext systems implement links by means of tags and pointers embedded in documents. With CD-ROMs and networked files it is not possible to freely add links into documents. Although hypertext CD-ROMs are available, the links are hard-coded into the documents and cannot be added to or changed on the medium itself. We believe that it is vital for the users of hypertext systems to be able to add their own links and annotations.

Clearly, some hypertext systems partly solve these problems, for example Intermedia keeps link information separate from the documents themselves, but we have attempted to design a new system that solves them all by providing an open model for hypertext/hypermedia with dynamic linking capabilities. The first instantiation of this new system is called Microcosm.

4 Principles

Microcosm has been designed with a set of guiding principles. Each design decision was made by evaluating the alternatives against this set of principles:

No ultimate distinction between author and user: All users should be in a position to add linking information to the system. The task of having to structure knowledge by adding links can be an effective learning experience and does not belong only to the teacher domain.

Loosely coupled system with a low level of interdependency: Microcosm is not just one program, it is a set of communicating tasks, open in structure. It should be possible to couple any other program into the system.

Modular to allow for slot-in experimental replacements. Microcosm is intended to be the workbench for a research environment. Every element, such as the linker, the document viewers, etc. must be replaceable. The system must be configurable and flexible so it may be used for investigating different ideas.

Separation of links from data objects: Information about relationships between documents is separated from the documents themselves, and is abstracted to a higher level. There are two worlds of information: the world of raw information: resource bases of text, graphics, video and sound; and the world of relationships and links between ideas, words, phrases, images and documents. Microcosm aims to keep these separate.

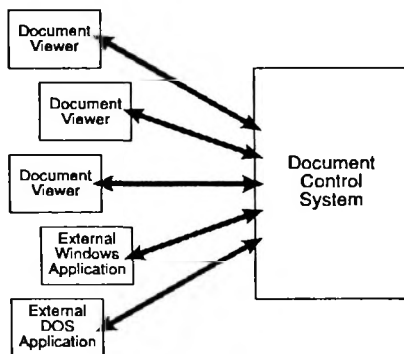


Figure 1: Microcosm Tasks

5 Design

5.1 Microcosm Tasks

In a Microcosm session, at least two tasks are running. The *Document Control System* opens documents routes messages and supports links. Most of the time it is not visible to the user except as an icon at the bottom of the screen. One or more *Document Viewers* are used to display document on the screen. One viewer is running for each document visible. Different viewers are used for different types of document or medium. Currently there are viewers for text (plain ASCII), monochrome and colour bit-mapped graphics, vector graphics, animated graphics, CD sound, and still or moving video from a video disk. There may be a mix of viewers on the screen at the same time and the software imposes no limit in practice to the number running. Document viewers communicate by means of messages, and all messages are routed through the Document Control

System. A third type of task, *External Applications* may also be running. These are applications which are not part of the Microcosm system but can be linked in by the Document Control System. For example, a spreadsheet can be used for graphing figures, or information can be extracted from a database, or from another hypertext system.

5.2 Microcosm Messages

The basic mechanism of Microcosm is the message. It is used for following links, creating links, opening new documents and invoking programs. The message contains three essential parts: a *selection* which will typically be one or more words of text or an area of an image or graphic object selected by the user; an *action* which is to be performed on the selection, such as "follow link" or "look this up in the dictionary"; and *context information* which includes the name of the document containing the selection, its position in that document, the date, and the user's name.

Selection: text of arbitrary length (if from a text viewer)			
Action: follow link/Create link/Complete link/Search dictionary/Language processing etc.			
Document name:	Offset: (bounding box for graphics)	Date:	Author:

Figure 2: Message Format

A message can be created in two ways. The simplest way is by the user highlighting a selection on the screen using a mouse, and then pulling down a menu of actions from a menu bar. Choosing an action will cause the message to be dispatched. The context information is provided by the document viewer. Messages are also created when buttons are pressed. Each document that has had buttons created for it has a button file associated with it. This button file contains a list of the locations of buttons within the document. A selection and an action are also stored for each button. A button then is merely a pre-defined selection bound to an action. It is a special case of the much more general message mechanism.

5.3 Message Processing

When the Document Control System receives a message it processes it according to the action. Each action has a task associated with it, and the task will be started up and send the required message. For example, if the action was *create link* then the message will be sent to the link creator. If it was *search dictionary* then the DOS dictionary application will be invoked with the message

placed on the command line or in a data file as desired. The action *follow link* is the most common and so will be described in more detail.

Links are stored in a link database. This is a simple flat file system with one record per link. Each record associates a link message with a dispatch message. When the linker receives a follow link action, it attempts to match the rest of the message with a link message in the database. The way this match is performed is dependant on information entered at the time the links were created about how specific or general the link is. If a match is found, then the dispatch message is sent back to the Document Control System. It contains the name of the target document and offset into it where the link was directed. This information is used to start up a Document Viewer of the appropriate type, load the target document, and move to the specified location. In this instance, the user is not aware of the operation of the Document Control System at all. As far as they are concerned they have asked to follow a link and the new document has appeared in another window.

Things are slightly more complicated if more than one link is available from a given selection. The user is then presented with a menu of links from which one or more may be chosen. Figures 3–6 illustrate the process of creating and following links

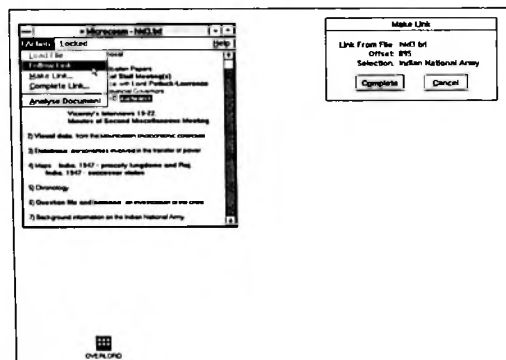


Figure 3: Screen shot of Microcosm showing a file relating to the Mountbatten archive from which the user has chosen to make a link on the phrase "Indian National Army". The "Make Link" dialogue box appears, ready for the user to complete the link at some point in the future. Meanwhile the user chooses to select the word "Auchinleck" and the action "Follow Link".

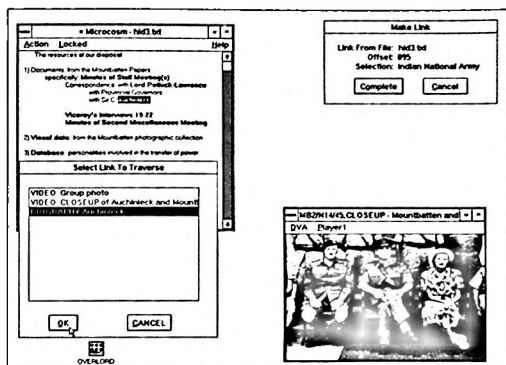


Figure 4: The link database has three links for the word "Auchinleck": two video and one text. The user has chosen first to view the second video link. The video picture appears in a window via the DVA-4000 board. The user now selects the text link to the Auchinleck biography.

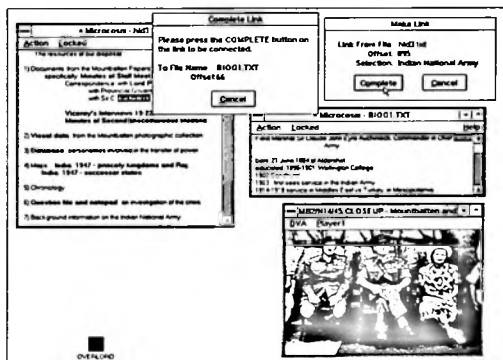


Figure 5: The biography text file appears in another window and the user chooses to complete the link made on the phrase "Indian National Army".

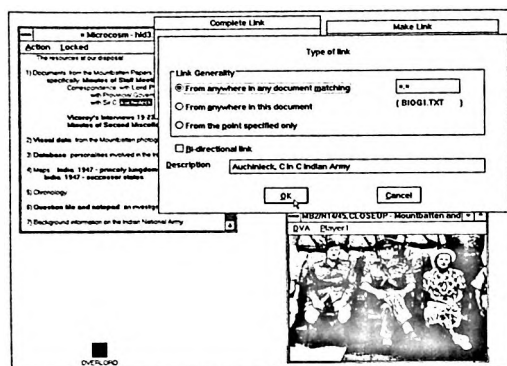


Figure 6: The link is made to apply from anywhere in any document. Thus a generic link has been created on the phrase "Indian National Army" linking it to the Auchinleck biography.

6 Implications

6.1 Authoring effort required—generic links.

Instead of storing the link information in with the documents, it is separated out into a knowledge base. This separation allows link to be *generic*, that is they are not applicable to only one document, but may be used with a whole class of documents. This greatly reduces the authoring effort required, since a new document can be brought into the system and immediately contain links. The Microcosm philosophy is that hypermedia links in themselves are a valuable store of knowledge. If this knowledge is bound too tightly to the documents, then it cannot be applied to new data. Instead it should be pushed away from the documents, into a world of generic knowledge. Microcosm aims to abstract the annotations, cross-references, and other knowledge and tools, so that when new documents are introduced, they immediately have links and cross references available. It should be possible to read a new paper on for example, microbiology, and merely to have to classify the subject of the paper to have a whole wealth of knowledge linked to the it. If the word *amoeba* had been linked to a videodisc clip of a moving amoeba, and the link had been specified as generic, then new document containing the word would automatically have the link. With the addition of simple filters, the information contained in the links can be applied to different areas of knowledge, being as generic or specific as wished.

6.2 Read-only media.

The separation of link information from documents has the benefit of allowing links to be made from read-only media such as CD-ROMs and remote file servers. This also simplifies links to and from media such as video and sound.

6.3 The possibilities for parallelism.

This is particularly important in a distributed hypertext, where it may be necessary to send out messages to remote machines (and slow to respond peripherals such as videodisc players) and then return control to the user while waiting for the remote responses, presumably while displaying some visual clue that they are still waiting for some response.

7 Current State of Development

The first implementation of Microcosm was prototyped in Actor and then fully implemented in C and MS Windows version 3.0. It runs on IBM PC's (or compatibles) with a 286 processor or above. MS Windows provided us with the basic building blocks for Microcosm, namely the windows and icon interface, tools for creating pull-down menus and dialogue boxes as required, and multi-tasking capability. It also provided a convenient message passing system, the Dynamic Data Exchange (DDE), which was used to handle interprocess communication in the Microcosm Document Control System.

The task of developing programs for MS Windows using the Software Developers Kit (SDK) can be very time consuming, especially to start with when there is a steep learning curve to be scaled. For this reason we conducted various experiments with the programming language ACTOR [Davis & Heath 1989] which is a single inheritance object oriented programming language similar to Smalltalk, but using MS Windows for its development environment, and producing MS Windows executable programs. While we found development in the ACTOR environment to be much faster, we found that the memory required was too great to allow it to run with other programs, and this prevented us developing the full loosely coupled communication process model in ACTOR. However, we were able to construct a restricted version of the Microcosm system using ACTOR in a relatively short period of time (about 3 weeks of part-time work by one person), and this turned out to be an ideal way of prototyping the user interface of the full system.

The first full implementation of Microcosm in MS Windows was ready for application development in January 1990. Our first application has been in the area of history and in particular in the development of an electronic version of the Mountbatten Archive. This archive is kept and maintained by the Hartley

Library at the University of Southampton, and contains approximately 250,000 text documents, 50,000 photographs, 35 reels of cine film, and various audio and video tapes from the collection of the late Lord Mountbatten of Burma. The challenge is to make this archive available to historians in a way that permits explicit and implicit relationships between the different elements of the archive (text, photographs, film etc.) to be stored and analysed. Together with other colleagues at the University we have created a prototype of the electronic archive, taking documents and photographs from the period 1947-8, when Mountbatten was Viceroy of India. Currently, the text documents are stored as ASCII files on the hard-disc and the photographs are stored on a laser videodisc. Images from the latter are displayed in windows on the VGA screen of the computer workstation via a VideoLogic DVA-4000 board. This application provided the content of Figures 3-6, and has allowed us to explore the potential of the various features that are built into Microcosm. A fuller report of this application and our work with the Department of History at Southampton is given in [Colson & Hall 1990].

8 Research Directions

Navigation facilities within Microcosm are currently very limited. Users move around the system by following links, initiating database searches and manipulating the windows and icons in which the information appears. This is clearly unsatisfactory for the majority of users especially when the information space is very large (as in the Mountbatten application). An immediate priority is to explore how some of the 'standard' hypertext navigation features, such as contents and index lists, overview maps, and guided tours, can be applied and possibly extended in Microcosm, and if any new navigation tools are suggested by the Microcosm model. One of the main problems inherent in a loosely coupled system such as Microcosm is to let the user know what areas of the screen are active. In most hypertext systems the link anchor is specific, and so some visual clue is displayed to indicate where the anchor is, but in Microcosm, the generality of the linking mechanism makes such clues more difficult to create.

We conducted various experiments using the ACTOR version of Microcosm, and found that so long as a reasonably fast database is used to store the links, it is not unreasonable to search the entire database for links that are specific to the document at load time, and then to search the document file for the positions at which the visual clue should be inserted. However, inserting clues for links with a greater degree of generality is a problem. Clearly it is not reasonable to attempt to match every generic link in the database into the file, if the number of links is

very large. Neither is it reasonable, if the file is very large, to attempt to match every possible text string in the file against the database. We found that the most acceptable solution to this problem was to allow the user to select an area of interest, and to select a menu option to match all possible strings in the selection against the database, and to display any matches found.

Another problem area in current hypertext systems that we plan to explore in Microcosm is version control and cooperative work. A number of solutions to this problem have been suggested in the hypertext literature [Akscyn 1988, Yankelovich *et al* 1985, Delise 1987]. The next version of Microcosm will have a system of installable filters. It is intended to include a system for version control similar to Intermedia's webs, but which would allow for separate users by allowing a hierarchy of installed filters, so that for example, the main filter might add the links for a particular user.

9 Conclusions

One of the main barriers to progress in the field of hypertext is the effort required to convert linear text to hypertext. Although vast numbers of pages are available on line, few of them have been converted to hypertext, and even fewer were intentionally authored as such. Users may rapidly come to expect to find very large amounts of information available as hypertexts, so methods of converting linear texts are an area of current research. Most attempts to convert linear documents [Furuta *et al.* 1989, Rahtz *et al.* 1989] so far have concentrated on the use of the structure of the document to break the document into appropriately sized nodes, then on using adapted indexing and cross referencing tools to automate link production, in order to produce a framework hypertext, which will usually require further manual cross referencing and linking. Nielsen [1990b] states that links in hypertext systems are "almost always anchored" at their departure point. In Microcosm we have move away from this principle: the selection and action mechanism, which provides the generic linking capability within the system, gives us a different, and we believe very powerful, method for assisting the authoring effort. This has already been proven extremely effective in providing hypertext/hypermedia access to a large multimedia archive. A word or phrase has only to be designated a generic link once for this to apply across the whole set of documents in the system and any new documents that are brought into the system. This facility then applies directly to documents stored on read-only media without any extra processing of the documents.

The main principles of Microcosm's design - its open nature, multi-tasking capabilities, bi-directional process communication and the separation of links

into a separate database - have allowed us to create a very powerful and flexible hypermedia system. In Microcosm it is possible to run Guide, say, as a Windows application and thus read Guide documents exactly as they were authored, but at the same time to apply links created within the Microcosm environment to the Guide document using the selection/action mechanism of Microcosm. Links can launch database searches or activate videodisc access, so that if we bring a new document into the system, in whatever format, words or phrases in that document can be linked, for example, to a videodisc database that has been previously created within Microcosm, with no extra authoring effort. These facilities enable the user to fully integrate documents authored in other hypertext/hypermedia systems into their own hypermedia knowledge-base.

Acknowledgements

Representations of photographs from the Mountbatten archive appear by kind permission of Trustees of the Broadlands Archives Trust.

We are extremely grateful to colleagues in the Department of History HiDES project, the Archives and Manuscripts Department of the Hartley Library, and the Department of Teaching Media for their invaluable support and assistance in the application of Microcosm to the Mountbatten Archive. Thanks are also due to colleagues in the Department of Electronics and Computer Science who contributed so much to the early discussions on the design of Microcosm.

References

- [Akscyn 1988] R. AKSCYN, D.L. McCracken & E. Yoder, "KMS: A Distributed Hypertext System for Managing Knowledge in Organizations", *Communications of the ACM*, Vol. 31, No. 7, 1988, pp 820-835.
- [Brown 1986] P.J. BROWN, "Interactive Documentation", *Software Practice and Experience*, March 1986, pp 291-299.
- [Bush 1945] V. BUSH, "As We May Think", *Atlantic Monthly*, July 1945, pp 101-108.
- [Colson & Hall 1990] F. COLSON & W. HALL, "Prologue to Partition: Viceroy Mountbatten and the Handling of the INA Crisis", to be published in the *J. of the Association of History and Computing*, Autumn 1990.
- [Davis & Heath 1989] H. DAVIS & I. HEATH, "An Evaluation of Actor for Developing Large Windows Applications", *Department of Electronics and Computer Science CSTR 89-5*, University of Southampton, 1989.
- [Delisle 1987] N. DELISLE & M. SCHWARTZ, "Contexts - A Partitioning Concept for Hypertext", *J. of the ACM Transactions on Office Information Systems*, Vol. 5, No. 2, April 1987, pp 168-186.

-
- [Englebart 1963] D.C. ENGELBART, "A Conceptual Framework for the Augmentation of Man's Intellect", in *Vistas in Information Handling*, Vol.1, Spartan Books, London, 1963.
- [Furuta *et al.* 1989] R. Furuta, C. PLAISANT, & B. SHNEIDERMAN, "A Spectrum of Automatic Hypertext Constructs", *J. of Hypermedia*, Vol. 1, No. 2, 1989, pp 179-195.
- [Halasz 1988] F.G. HALASZ, "Reflections on Notecards: Seven Issues for the Next Generation of Hypermedia Systems", *Communications of the ACM*, 31(7), July 1988, pp 836-852.
- [Nelson 1967] T.H. NELSON, "Getting It Out of Our System", in Schechter G. (ed.) *Information Retrieval: A Critical Review*, Thompson Books, Wash. D.C., 1967.
- [Nielsen 1990a] J. NIELSEN, "The Art of Navigating Through Hypertext", *Communications of the ACM*, 33(3), March 1990, pp 297-310.
- [Nielsen 1990b] J. NIELSEN, *Hypertext and Hypermedia*, Academic Press, 1990.
- [Rahtz 1989a] S. RAHTZ, L. CARR & W. HALL, "New designs for Archaeological Reports", *Science and Archaeology*, no. 31, 1989, pp 20-34.
- [Rahtz 1989b] S. RAHTZ, L. CARR, & W. HALL, "Creating Multimedia Documents", in *Hypertext : the State of the Art* (ed. McAleese) Intellect Ltd, 1990.
- [Shneiderman 1987] B. SHNEIDERMAN, "User Interface Design for the Hyperties Electronic Encyclopedia", *Proceedings of the ACM Hypertext'87 Conference*, Chapel Hill, NC, November 1987, pp189-194.
- [Shneiderman *et al.* 1989] B. SHNEIDERMAN, D. BRETHAUER, C. PLAISANT & R. POTTER, "The Hyperties Electronic Encyclopedia: An Evaluation Based on Three Museum Installations", *J. American Society for Information Science*, 40 (3), May 1989, pp172-182.
- [Shneiderman & Kearsley 1989] B. SHNEIDERMAN & G. KEARSLEY, *Hypertext Hands On*, Addison-Wesley, 1989.
- [Yankelovich *et al.* 1985] N. YANKELOVICH, N. MEYROWITZ & A. VAN DAM, "Reading and Writing the Electronic Book", *IEEE Computer*, October 1985.
- [Yankelovich *et al.* 1988] N. YANKELOVICH, B. HAAN, N. MEYROWITZ & S. DRUCKER, "Intermedia: The Concept and Construction of a Seamless Information Environment", *IEEE Computer*, January 1988, pp 81-96.