# The Microcosm Link Service and its Application to the World Wide Web

**Les Carr, Wendy Hall, Hugh Davis and Rupert Hollom**

**Department of Electronics and Computer Science**
**University of Southampton**

**E-mail: {lac,wh,hcd,rjh}@ecs.soton.ac.uk**

## Introduction

For some time now, designers of hypermedia systems have recognised the need to move away from closed systems to open environments which separate the link structure from the data in the system, and enable separate link and data processing. The main motivations behind this development are the need to reduce authoring effort in large-scale hypermedia applications and to make them more easily modifiable, customisable and extensible. Microcosm is one such open hypermedia system which has been developed at the University of Southampton. At the heart of Microcosm is the Link Service, which allows links maintained by the system to be applied to information native to third-party applications in the host environment. This paper describes the Microcosm system and discusses the benefits of combining its link service capabilities with the WWW distributed technology.

## Open Hypermedia Systems

The hypertext research community has continued to explore the development of hypertext systems that handle information on a large scale. It is clear that authoring effort and the management of links are major issues in the development of large hypertext and hypermedia applications. This has led to the design of systems which separate the link data from the document( data. This enables information about links to be processed and maintained like any other data rather than being embedded in the documents themselves. Research effort has been concentrated on the development of open hypermedia systems that enable hypermedia functionality to be integrated into the general computing environment and allow linking from all tools on the desktop. The hypertext management system then becomes much more of a back-end process than a user interface technology.

The paper by Malcolm *et al* [1991] is an excellent summary of user requirements for hypermedia that extend well beyond the scope of the current generation of commercial hypermedia systems. The paper in fact argues for the development of open hypermedia systems that fulfil the following criteria: an adaptable environment for the integration of data tools and services that do not confine users to a particular suite of editors and specialised software packages; a system that is platform independent and distributed across platforms; a system which makes it easy for users to find, update, annotate and exchange information; and a system in which all forms of data and media are treated in a conceptually similar manner.

It is clear that the current generation of hypermedia systems do not fit any of these criteria. This is because they are all to a greater or lesser extent closed systems. A closed hypermedia system provides a fix set of encapsulated applications which are normally tightly integrated with the hypermedia linking mechanisms [Leggett *et al,* 1993]. The hypermedia links are generally embedded in the data which is stored in a proprietary document format. It is therefore not possible to access the data or the links from outside of the hypermedia system. In contrast, an open hypermedia system provides a protocol that allows any application to participate in the hypermedia service. The applications are loosely integrated with the hypermedia linking mechanisms but the degree of openness can vary considerably according to the restrictions that the protocol imposes on the applications: ranging from complete control to none at all. A hypermedia link service and an associated protocol is therefore an essential part of any open hypermedia system.

In his book Literary Machines [Nelson, 1981], Nelson describes Xanadu as a "universal open hypermedia environment". Unfortunately Xanadu has never been fully implemented but Nelson's ideas certainly laid the foundation for the development of such systems. The Intermedia project [Yankelovitch *et al*, 1985] was also a pioneering project in this respect. Sun's Link Service [Pearl, 1989] was probably the first implementation of a hypertext system that could be truly described as an open system. Running on Sun workstations, in a distributed environment, it consists of a link database service that is integrated with registered applications through a library that implements a communication protocol. It therefore provides a link service to existing applications and dedicated applications for editing and processing information are not required. However an application has to be aware of the link service and be registered with it in order to make use of the service. It must be able to send information to the link service that identifies the element in the document from which a link is to be followed.

Following on from these early developments a number of research groups have developed various models of open hypermedia systems. These include Microcosm [Davis *et al*, 1992], Multicard [Rizk & Sauter, 1992], Hyper-G [Kappe, 1991], and HyperBase [Schutt & Streitz, 1990]. Such systems are largely still in the province of the research laboratory although commercial versions of some of them are envisaged. The openness of the system and the scope of the link service varies in each case. We shall present the Microcosm system in detail and explain how it can be applied to multimedia information management.

## The Microcosm Approach

The Microcosm system has been under development at the University of Southampton since 1989. The philosophy behind its development and on which we based the original design is shown in Figure 1. This three-layer model is similar to the three-level architecture of a traditional DBMS. Just as the conceptual level of a DBMS maps the user's view of the data onto the internal storage structure of the database, so the link service maps the data in the application to the multimedia information catalogued by the document management system.

Our principle criteria in the design of Microcosm were the separation of links from data; the ability to maintain a database of links; the ability to apply links to data native to other applications and to read-only material; the use of the selection-action metaphor as the basic link interaction mechanism; a declarative model for link specification; an open system to encourage portability across different hardware platforms; a highly modular system to allow for easy extensibility.

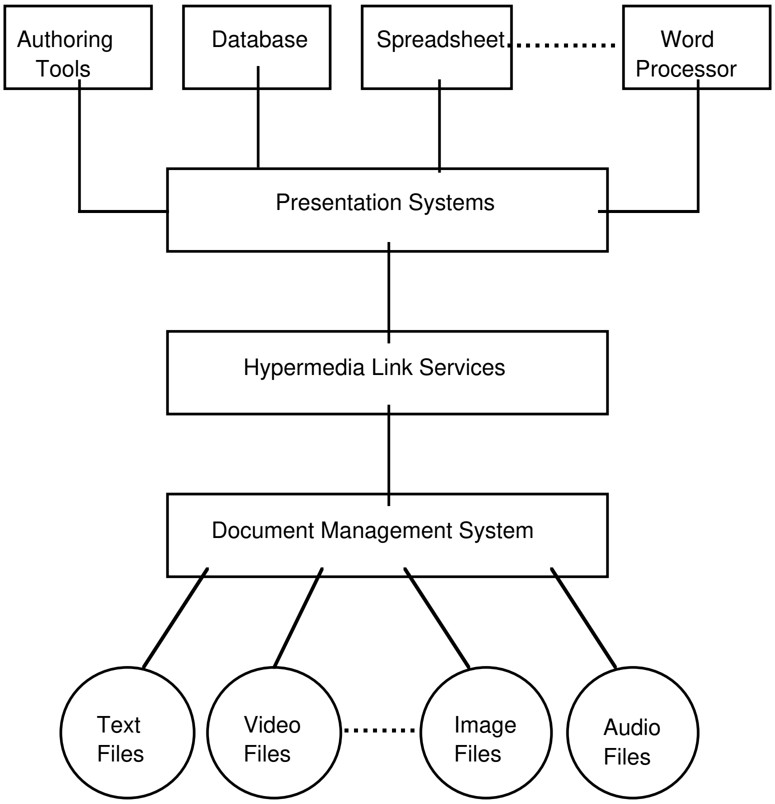


Figure 1: *The Microcosm System Architecture*

The consequences of these design principles were the need to develop a link management system and a method of allowing links to be applied to applications that are not under the control of the hypermedia system. We also wanted to keep the system as open as possible so that link following was just one type of action that could be applied to an object in a data file. Our aim was to produce a generalised multimedia information management environment based on the availability of an open hypermedia system.

Microcosm is currently implemented on MS-Windows 3.1. Versions of the software are also currently under development for the Apple Macintosh and Unix systems running an X-Windows front end. The model was first described in Fountain et al. [1990], but has subsequently been fundamentally altered and extended in order to keep all aspects of the system open [Davis *et al,* 1992]. It is best understood as a set of autonomous communicating processes (agents) which supplement the facilities provided by the operating system.

In Microcosm the user interacts with a *viewer*. A viewer is any application in which data may be displayed. Messages to perform actions are sent from the viewer to Microcosm, which then dispatches the message through a chain of *filters*. Each of these filters is then given the opportunity to respond to the message by blocking it, passing it on or changing it before passing it on. Based on the message contents, some filters may add new messages to the chain. Eventually the message(s) will emerge from the filter chain and arrive at the Link Dispatcher. This will examine the messages to see if they contain any available actions (such as links to follow), and if so it will offer these actions to the user.

Filters in Microcosm are independent processes, and may be dynamically installed, removed or reordered [Hill *et al*, 1992]. Particularly important filters are the link databases, or linkbases as we call them, which contain all the information about link availability. When a message arrives at a linkbase requesting a link to follow, the process looks up the source anchor in a data file and, if found, returns details of the link destination, which are passed on to the link dispatcher. There can be any number of linkbases in the filter chain: the user can install and delete linkbases at run-time so it is easy to present different views of the data to different users. Other filters include processes to aid navigation (such as a history mechanism and local map device) and processes to compute dynamic links.

The Microcosm model for creating and following links is based on the process of *selection* and *action.* The user selects the information they are interested in, which may be for example a word or a phrase in a text document or an area in an image, and chooses an action from a menu to be performed on this action, such a follow or make link. This enables us to create a spectrum of link types. Specific links are defined on an object at a specific point in a particular source document. Local links are defined on an object at any point in a particular source document. Generic links are defined on an object at any point in any document. The local and generic links, which have dynamic source anchors, are the features that enable the Microcosm link service to be applied to other applications. This spectrum of links is naturally extensible to links created dynamically by any process that can be applied to the source selection. For example, we have implemented text-retrieval links which are created dynamically using a text-retrieval search based on the source selection.

We have implemented buttons in Microcosm, but they are merely a special case of a specific link, where the "follow link" action is tied to the specific source selection. When a document is viewed through Microcosm, the linkbases are automatically queried to search for buttons in that document, which are then highlighted on the screen. When the user clicks on a button the follow link query is automatically initiated without the user having to select the action. This side-effect of what is essentially a declarative model for hypertext linking causes a number of problems. Buttons can only be displayed in viewers which can receive messages from Microcosm

We currently have three categories of viewers. Fully aware Microcosm viewers are generally ones we have written ourselves to display different multimedia data types including text, graphic images, video, animations and sound. They have the full range of Microcosm functionality. Partially aware viewers are applications from external sources, such as word processors, databases and other hypermedia systems, that have been adapted to be Microcosm aware i.e send messages to Microcosm. For example, in the Microsoft Windows environment almost any application that has a level of programmability and access to the DDE can be made Microcosm aware. Unaware viewers are applications that cannot sent message to Microcosm: in the Windows environment this usually translates into applications that have no DDE access. To enable Microcosm link services to be applied to such applications, we have introduced the idea of "clipboard links". Microcosm can be set-up to monitor the contents of the clip-board and to perform link processing actions on this data. In this way, the Microcosm link service can be applied to different desktop applications as well as used to support hypermedia authoring in the large.

In order to ensure that the system is kept as flexible as possible, we have adopted a tagged ASCII message format. Any viewer or filter may introduce any tag and data it likes into the message. Filters respond to the tags they know and ignore the rest. Microcosm for Windows makes use of the Dynamic Data Exchange (DDE) facility to pass the messages from process to process The Macintosh system uses Apple events, and the Unix version uses sockets for the message passing.

Writing a new filter is a simple task that could be undertaken by any Windows programmer without any special understanding of Microcosm internals. A procedure is written to analyse the incoming message for the required tag(s) and to take appropriate actions. This is then inserted into a standard shell which takes care of all the communication with the DDE channels, and the new filter is then compiled. This flexible modular approach makes it easy for any programmer to make major changes to the functionality of the system.

We have developed our own Document Management System (DMS) which registers the multimedia information available within particular applications. This has been implemented to provide users with a catalogue of available information and can be used directly to find and retrieve information. The list of data types that can be stored in the DMS is extensible and can contain virtually any media data type. Additionally links can lead to third-party application programmes, these can also be declared as data types in the DMS.

## Hypertext in the Large

Although Microcosm was designed to deal with hypertext on a large scale, it has been implemented in the context of a personal workstation environment and currently lacks the mechanisms to deal with a distributed document set. In contrast WWW addresses the issue of hypertext in a global context—not just a single text or group of intimately related texts, but hypertext as a universal literature resource.

The experience that most other hypertext systems provide is in the realm of individual documents or local document collections with a controlled environment and context. The design of the WWW project has kept the node and links model of these traditional hypertext systems intact but extended the node addressing scheme to allow remote nodes and defined a node transport mechanism to allow the hypertext to be extended across a network.

This simple node-links model and the familiar authoring paradigm that it accompanies has particular implications for the scalability and maintainability of a very large and highly distributed corpus.

## WWW Navigation Issues

Navigation of the Web by an user is undertaken in one of two ways:

> from any given document the user selects a linked document by clicking on an anchor, so that a path of nodes is traversed in order to reach the intended destination

the user can ‘jump’ to the exact document required by specifying the known address (URL) of the document.

The former mechanism requires the user to follow semantic cues in the contents of the documents in order to repeatedly choose the correct links to follow. The latter requires the user to to make use of an already-known address, which may come from:

a compiled-in list of well-known documents provided with the Web client viewer software

a short hotlist of remembered addresses of previously visited documents which were deliberately noted by the user

a comprehensive list of every document ever seen by the Web viewing software

outside the Web environment: new sites advertising their URLs on other electronic services (mailing lists and network news), or word of mouth from colleagues

*i.e.* apart from link following, it is only possible to navigate to a document if you have already been there, or if you are provided with a handle to it by its author or by someone else who has been there. This is then a ‘pure’ link-following environment, without recourse to text searches or comprehensive document catalogues; it is almost impossible to navigate the Web with the aim of finding all documents about a particular topic.

The problem of topic-based navigation of the Web is similar to the problem of finding a file on a particular subject on the Internet’s anonymous FTP service. In that environment at first enthusiastic volunteers published regular lists of sites and kinds of files at each site. Some sites also used to provide a file containing a complete list of all the files available from their machine. Eventually a single site provided a database of the names of files available at all of the well-known anonymous FTP sites; an interactive query service (known as *archie*) allowed any user to find out where a file was archived given a fragment from that file’s name. This service has now been replicated across several dozen sites across the whole Internet, so that any user can obtain a list of potentially relevant files *as long as the name of the file is indicative of its contents*. A similar system could be applied to the Web; already software is available to allow the administrator to automatically catalogue each of the Web server’s files.

**WWW Authoring Issues**

*Link fossilisation* is a significant disadvantage of WWW and occurs because link specifications have to be published as part of the document and cannot be changed without revising the document. *Link decay* is also seen since links refer to their destination anchors via a specific machine name and path name. Any change to the position of the destination requires every source document which refers to it to be changed—once published a document can never be moved or deleted. Although this is not an insurmountable problem in a locally controlled context, WWW used as a world-wide publishing mechanism assumes that every document is forever associated with its published address. *Dead ends* frequently occur in WWW because only native

WWW documents can have embedded links. If traversing a link leads to a foreign document being displayed by a foreign application (an RTF file displayed by Word) then no WWW links may be followed from it.

**Authoring and Linking in Microcosm**

A generic link, the most common link type in Microcosm hypertexts, allows the author to associate a document with any occurrence of a particular textual string in any document. At first sight this may seem to be just a text retrieval operation, however there are certain key differences. Firstly, from a practical point of view, a generic link requires no indexing of the possible destination documents, nor a searching operation on every document in the hypertext in order to satisfy the link—a generic link has none of the overheads associated with text searching. Secondly, the difference between generic links and text retrieval is the difference between intentional and non-intentional hypertexts: a link expresses an author's knowledge of a relationship between the meaning of two entities in the hypertext, whereas a text retrieval operation expresses a statistical similarity in textual features of two hypertext entities. It is possible to liken a generic link to a text retrieval operation in reverse: a generic link defines a collection of applicable sources, whereas a text retrieval operation describes a collection of applicable destinations.

The flexibilty of Microcosm link sources provides a 'reversed' hypertext authoring paradigm: 'which other nodes may be linked to the current node' instead of 'where can the current node lead to?'. Effectively, the author, using the generic link mechanism, is labelling the document with key words or key phrases. Thus the authoring paradigm has become *declarative* in nature, describing the data rather than the processes involved in document links.

**Scalability of the Microcosm Link Model**

Hypertext packages are frequently difficult to author in a scalable or generic fashion which allows for expansion or economic re-use for different purposes. The links, authored for a particular purpose, are fixed inside the document content and fixed to specific destinations.

Updating a Microcosm hypertext by adding new nodes involves one of two scenarios. If the nodes are new general resources (primary materials) then a group of new generic links must be added which will retrospectively apply to the existing hypertext components. If instead they are new secondary materials (*e.g.* student essays or teacher commentaries on the primary materials) then they will already be affected by the existing links. In this respect the Microcosm hypertext model is incrementally scalable.

Changing the purpose of the hypertext may involve keeping the collection of nodes substantially the same, but reworking links to provide different structures of access. In many hypertext environments including the Web, changing the links means rewriting the texts because the links are embedded in the texts. In Microcosm it simply means applying a new set of linkbases to the same material, in a similar way to Intermedia's use of webs. Another advantage of Microcosm is that material which is added during the 'repurposing' process will be automatically affected by any retained linkbases. Since many hypertext environments provide embedded point-to-point linking (*i.e.* from *here* you can go *here*) they fail to offer such expandability or maintainability.

As a particular example of the advantages of this authoring paradigm, consider setting up a multiple-choice test based on material in a standard course text. In a normal environment containing only specific links between nodes, for each possible wrong link (*i.e.* wrong answer) a separate correcting explanation must be written for the user, recalling the material in the original sources. Using Microcosm, the question, the text of each answer and any explanations written will automatically be linked back to the concepts in the original sources.

**Microcosm Meets WWW**

The World-Wide Web is characterised by a number of components: (i) a single, well-defined native data format for use with a document viewer, (ii) a universal addressing scheme with associated transfer protocol and (iii) a hypertext authoring scheme in which precise destination addresses of links are specified as part of the source documents. In comparison with WWW, Microcosm is characterised by (i) a co-operative framework for diverse document viewers and (ii) a hypertext authoring strategy which is based on generic relationships between source and destination documents.

Microcosm does not suffer from some of the problems of the Web. Dead ends do not occur because almost any program can be used as a Microcosm viewer for many different kinds of data: links can be followed not only between text and graphic files, but between wordprocessed documents, CAD documents, spreadsheets, databases, video documents and simulations *etc*. Links do not get fossilised because they are not embedded in the documents to which they refer, they do not decay (as often!) because they represent rules for linking sets of documents together, rather than specific hardwired document references.

One approach which we have implemented is to use Microcosm as a local viewing environment for the World-Wide Web (a true hypermedia replacement for the standard Mosaic viewers). However, in this paper we describe a more general approach that provides flexible link authoring and following for WWW users who do not have a local Microcosm environment. What follows is a description of the authors' efforts to translate some of the major components of the Microcosm model into the WWW environment and so to provide Microcosm services to all users of the Web.

The major features of Microcosm are the selection and action link-following paradigm, external linkbases and the message passing framework. WWW provides a message-passing framework: requests (in the form of URLs) are sent by a client viewing application via HTTP to a WWW server and documents (in HTML format) is received back. A client that wants to obtain Microcosm link services can then express a link request message in URL format and send it via HTTP to a Web server. The server can invoke a process which mimics the action of the Microcosm linkbase, and sends back (in HTML format) a list of destination documents that were produced by the linkbase. That document would be displayed to the user as an equivalent of Microcosm's Link Dispatch dialogue box, allowing the user to choose from among the available destinations by clicking on the HTML buttons which describe them.

An essential pre-requisite for Microcosm generic links is the ability to make arbitrary selections within documents, not just to click on predefined HTML buttons. For this purpose a simple Motif application called ‘Microcosm Lite’ has been written which presents the user with a single button labelled ‘Follow Link’. When this button is pressed the application grabs the current selection (whether from the Mosaic viewer, or any arbitrary window) and turns it into a URL. This URL is passed to the Mosaic viewer for ‘opening’. In this way, whichever application the selection was made in, the link request and destination display is performed by Mosaic.

## Conclusions

The flexibility of the Microcosm model makes the possibilities endless. We are currently experimenting with the development of many different types of filters to automatically generate links to reduce authoring effort, and to create links dynamically according to different algorithms. For example, we are using rule-based algorithms to create an intelligent agent device. The integration with visualisation systems such as Autocad is allowing us to experiment with different metaphors for the development of user interfaces to a large sets of multimedia data.

An open hypermedia system like Microcosm has an intrinsically different feel from closed hypermedia systems. The onus is on the user to interrogate the system in order to ask for more information rather than expecting the system to announce to the user that there is more information about a particular subject. The infinitely more flexible model allows us to customise the hypertext environment to the user’s needs.

By enhancing the Web with Microcosm’s link services, WWW readers would be freed from the tyranny of the button. Currently, in order to access a piece of information on the Web it is necessary either to know its address or to be able to find a document that contains a link which references it. In an environment which has no alternative methods of navigation (*e.g.* a hierarchical structure) this can cause considerable problems, especially if documents are revised. Although a problem in a localised hypertext environment, this is especially significant in a global, unco-ordinated information system. Using Microcosm’s ‘generic links’ the reader can instead select any relevant text to act as a link to the required information.

Similarly, WWW authors would have greater freedom in the authoring process: instead of providing explicit buttons for navigation to every relevant piece of material, generic links can be used to provide standard services across a whole domain of information.

## References

Davis, H.C., Hall, W., Heath, I., Hill, G.J. & Wilkins, R.J. “Towards an Integrated Information Environment with Open Hypermedia Systems” In Proceedings of ECHT’92, ACM Press, pp 181 - 190 (1992).

Fountain, A.M., Hall, W., Heath, I. & Davis, H.C. “Microcosm: An Open Model for Hypermedia with Dynamic Linking”. In Proceedings of ECHT’90, Cambridge University Press, pp 298 - 311 (1990)

Hill, G.J., Wilkins, R.J. & Hall, W. “Open and Reconfigurable Hypermedia Systems: A Filter Based Model” to appear in Hypermedia 5, 2 (1993)

Kappe, F., Maurer, H. & Sherbakov, N. “Hyper-G - a Universal Hypermedia System”, IICM Technical Report, Graz University of Technology, Austria (1992).

Legget, J.J., Schnase, J.L., Smith, J.B., & Fox, E., “Final Report of the NSF Workshop on Hyperbase Systems”. TAMU-HRL 93–002, Hypermedia Research Laboratory, Texas A&M University, USA (1993).

Malcolm, K.C., Poltrock, S.E. & Schuler, D. "Industrial Strength Hypermedia: Requirements for a Large Engineering Enterprise". In Proceedings of Hypertext'91, ACM Press, pp 13 - 24 (1991)

Nelson, T. “Replacing the Printed Word: A Complete Literary System”. In Lavington, S.H. (ed.) Proceedings of IFIP Congress 1980, North Holland, pp. 1013 - 1023 (1980).

Pearl, A. “Sun’s Link Service: A Protocol for Open Linking”. In Proceedings of Hypertext’89, ACM Press, pp 137 - 146 (1989).

Rizk, A., Sauter, L., “Multicard: An Open Hypermedia System”. Proceedings of the 4th ACM Conference on Hypertext 1992, pp 4–10, ACM Press (1992).

Schutt, H.A. & Streitz, N.A. “Hyperbase: A Hypermedia Engine based on a Relational Database Management System”. In Proceedings of ECHT’90, Cambridge University Press, pp 95 - 108 (1990).

Yankelovich, N., Meyrowitz, N. & van Dam, A. “Reading and Writing the Electronic Book”. IEEE Computer 18, 10, pp 15 - 30 (1985).