Open and Reconfigurable Hypermedia Systems: A Filter-Based Model

Gary Hill
Rob Wilkins
Wendy Hall

CSTR 92-12

University of Southampton

Department Of Electronics and Computer Science
University of Southampton
Southampton SO9 5NH.

Abstract

The need for open hypermedia systems has been well-argued by various authors. In this paper we discuss the Microcosm model for open hypermedia and show how filters have been used to extend this model both for greater efficiency and to make it totally reconfigurable. This enables users to tailor the functionality of the system to meet their own requirements, for example to select different sets of links, or to add navigation tools. The implementation of a management system for filters is described, and examples of how it has been used to extend the functionality of the model are presented, in particular the use of filters to design and implement a set of navigation tools for Microcosm. This latter case study demonstrates the power and flexibility of the filter-based model since all navigation tools for the system can be implemented as filters. The filter technology is discussed in the context of Microcosm but can be generalised to other hypermedia systems.

Contents

1. Introduction....4
2. The Microcosm Model....5
3. Filters In Microcosm....7
   3.1. Reconfigurable Hypermedia....7
   3.2. Filter Management....7
   3.3. Building a Hypermedia System from Filters....8
4. Case Study : Filters for Navigation....10
   4.1. Navigation in general.....10
   4.2. Navigation Filters in Microcosm.....11
      4.2.1. History tool.....13
      4.2.2. Mimic, guided tours.....13
      4.2.3. Short term memory dispatcher.....14
      4.2.4. Local maps....14
   4.3. Future work.....15
5. Future Development....15
   5.1. Dynamic Filter Management....15
   5.2. Distributed Systems....16
6. Conclusions....17
Bibliography....17

Introduction

A number of authors have suggested that existing hypermedia systems are not satisfactory for large information sets, for example (Halasz, 1988; Malcolm et al, 1991; Halasz, 1991). Many hypermedia systems are now being developed with an open integrated architecture in order to address this problem. Although at present there is no absolute definition of an open hypermedia system, various aspects are described in (Pearl, 1989; Shipman et al, 1989; Creech et al, 1991; Fountain et al, 1990; Davis et al, 1992b; Maurer & Tomek, 1990).

The key difference between open and closed hypermedia systems is in their relationship to the user's existing environment. A typical closed system is a separate, application that is isolated from the user's other applications such as electronic mail, word processors, spreadsheets etc. Integrating a closed system with a diverse set of existing applications such as this is very difficult.

In contrast, a typical open hypermedia system will act as a virtual linking layer abstracted from the information bound to various existing applications. Whenever possible it will use existing applications to present the information, for example the Microcosm system described below and in (Fountain et al, 1990; Hall et al, 1992; Davis et al, 1992b) allows the user to follow links from any Microsoft Windows program that has clipboard support. This transforms the user's previously discrete set of tools into a fully integrated information environment. The hypermedia system simply acts as a framework that provides the relationships between documents, no linking information is embedded within the documents, instead it is maintained in external link databases. This has a number of side effects, the link information is easily analysed, the applications that generated the documents can still be used to edit and maintain them, and each user is able to configure link sets according to their own requirements.

In this paper we discuss the Microcosm model for open hypermedia and show how filters have been used to extend this model both for greater efficiency, and to make it totally reconfigurable. The latter allows the system to be extended in a way that goes far beyond the capabilities of a scripting language. Not only can new program modules be easily added, but also the user is able to select and combine different link databases, and apply the resulting set of links to data in the system and in other applications.

The Microcosm Model

Microcosm is an open hypermedia system that aims to address the issues raised above. The initial prototype of the system has been previously described in (Fountain et al, 1990). This section describes the model developed from that system, and the resulting implementation.

The earlier paper describes what is now the front-end of the current system, a set of document viewers for the various types of supported media. These viewers are managed by the Document Control System (DCS), which deals with starting new viewers, and informing viewers of documents to be displayed.

In the Microcosm prototype, linking functionality was provided by a separate link database, but control of the database was encapsulated within the DCS. There was no way to easily extend functionality with additional databases or other tasks. To solve this problem, the processing of messages from document viewers has been moved from the DCS to an additional task called the Filter Management System (FMS). In turn, the FMS manages a set of tasks that provide all message processing functionality. These tasks or 'filters' process messages with certain actions. The FMS receives messages from document viewers via the DCS (for example 'Follow Link', 'Make Link' etc.) and passes them on to the current set of filters. The logical structure of this new model is shown in figure 1.

By implementing filters and document viewers as separate tasks, it is possible to change or extend both the functionality and document types available quickly and easily. This is true both for developers and users. A user can update their personal configuration to allow new tasks to become available, whilst the developer does not require any recompilation of existing code to provide new functionality, the new task is simply required to adhere to the communication specification for a viewer or filter.

Additionally, since the DCS and FMS are also separate tasks, they can also be easily modified to experiment with different ideas with minimal alteration to other aspects of the system. For example to vary the way filters are managed, only the FMS needs to be altered, and possibly the filters required. No changes need to be made to the DCS or the document viewers.

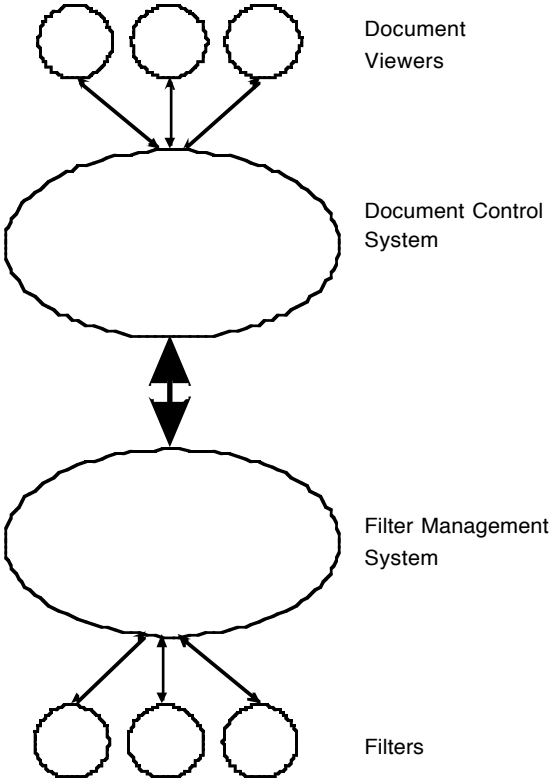

Figure 1 : Logical Model of Microcosm

Communication between these various tasks takes place by means of messages passed between them. The basic format of these messages is that described by Fountain et al. They consist of an *action* such as 'Follow Link' or 'Create Link', a *selection* upon which the action is to be carried out, and may contain other information such as the document in which the selection has been made, and the offset of the selection within the document.

In order to allow messages to be extended to incorporate any extra information required by new actions, the messages are in a tagged ASCII format, where fields are identified by arbitrary tags, immediately followed by the information for the field. Viewers and filters can introduce any new tags they wish, as well as any tags from a pre-defined core set. Because the size of the message, and the fields within it are not pre-defined, viewers and filters are able to share additional information simply by defining new tags.

This filter-based model allows maximum flexibility in the configuration, and future development of hypermedia systems based on the Microcosm model. Further details of the design and implementation of Microcosm can be found in (Fountain et al, 1990) and (Heath, 1992).

Filters In Microcosm

Reconfigurable Hypermedia

As is described above, the core Microcosm processes (DCS, FMS) have no integral hypermedia functionality. The DCS simply starts up viewers as required and sends messages indicating documents to be viewed. Messages from viewers are passed by the DCS to the FMS.

The FMS acts as a junction box which routes messages between the active filters. It is this set of active filters that defines the functionality available to the user and by changing filters or adding additional filters the user is able to tailor the system to their specific needs. Users are able to invoke functions in various filters by sending a message specifying the appropriate action. At any time, each user is able to reconfigure their personal view of the system to include new filters, or to replace filters with alternatives.

It is this total flexibility in the range of functions that a filter can provide that allows a Microcosm 'system' to develop and expand over time as the user's requirements change. For example, filters may be used to provide navigation facilities as described later in this paper, or as a way to add alternative linking paradigms such as string search (Faloutsos et al, 1990), set classifications (Parunak, 1991) or text retrieval (Li et al, 1992). Similarly, as document viewers are added to handle new media types, any special functionality required to handle their particular requirements can easily be added to the system by creating new filters.

Filter Management

The Filter Management System takes complete responsibility for the execution of filters, setting up communication with them, and the routing of messages from viewers to the current set of filters. The model that the current FMS uses is a straightforward linear chain of filters. Messages are received by the FMS which passes them to the first filter in the chain. As messages are received by the FMS from filters, they are passed on to the next element of the chain. In this way, messages make their way through to the end of the chain and are returned by the FMS to the DCS which may act on the message, perhaps to start a viewer in order to open a document, or send requested information to a viewer.

It is from this simple chain model that filters were originally named, at any point in the chain a filter process can examine a message it has received and make a decision about whether to process it. If the filter does not wish to process a message, for example if the action of the message is not understood by the filter, then it is returned immediately to the FMS to continue along the chain. If the message action is understood, then the filter will generally do one of two things. It may generate additional messages based on the content of the original message, for example a text retrieval filter will generate messages describing documents that match a source selection in the original message. Alternatively, it may use certain conditions to block the message such as user rights or context matching. Only if the appropriate conditions are met is the message passed back to the

FMS and hence the rest of the filter chain. The following section describes how filters working in this way can be quickly built into a functional hypermedia system.

Building a Hypermedia System from Filters

It is possible to build a functional hypermedia system from quite a small set of filters. The basic requirements are the ability to create links, access links and present links to the user.

Since Microcosm stores links separately from the documents being viewed, it is necessary to have a filter providing a database of links, which we call a 'linkbase'. This is probably the most important filter, since it provides access to a specified database of links. Its primary function is to process messages with the 'Follow Link' action. These messages contain a selection that the user wishes to try to find a link from, along with the source document and offset within the document, of the selection. If any links are found in the database that match these criteria, 'dispatch' messages are sent back to the FMS indicating the destination and description of the link. These messages continue along the filter chain and may be processed by other filters, and eventually presented to the user. At any one time there may be a number of different linkbases active or available in the system providing different 'views' of the relationships within the information being accessed. Full details of the design and implementation of the linkbase are given in (Heath, 1992).

The linkbase will also process certain other messages. Viewers can interrogate the database with 'Find Buttons' messages which return locations of buttons in the source document allowing important links to be highlighted. Also processed are messages with the 'Create Link' action which provide details of new links to be added to the linkbase.

These messages are formed by a separate 'linker' filter which captures 'Start Link' and 'End Link' messages from viewers, and then allows the user to bind them together to provide a full link definition with source and destination details, description and type of link (types of link are described in (Fountain et al., 1990)) to form a new message which is given the 'Create Link' action. Although this function could be included within the linkbase itself, by keeping it separate it is possible for other filters to add links, perhaps automatically as a result of document analysis. The ability to make links can also be easily controlled by restricting user access to the linker filter.

One other filter is required to make the basic link functionality of the filters described above more manageable. This is the 'dispatch' filter which captures dispatch messages that are generated as a result of following links. Without this filter, all these messages would be routed straight back to the DCS and all the destination documents identified would be loaded into viewers. If a large number of links were found this would become confusing. The dispatch filter simply displays the descriptions of all the links found in a dialogue box from which the user can select those links considered appropriate.

These filters present a very basic hypermedia system to the user, however they are merely a basis around which a larger system can be

built by adding more filters with additional functionality, or replacing existing filters with ones that provide alternative functions. The following list describes a few of the different filters that have been built to enhance Microcosm systems.

Memory Dispatcher. This filter is an alternative to the straightforward dispatch filter described above. It adds a 'short-term memory' of links recently followed, and is able to guide the user towards new material in a similar way to the work described in (Briggs, 1991).

Compute Links (Li et al., 1992). This filter provides an alternative to authored links. Documents are pre-indexed to find keywords. When a message with the 'Compute Link' action is received, the filter attempts to find documents that match the selection in the message. The possibilities are presented to the user in the dispatch filter's dialogue box with a similarity rating.

HiDES (Colson & Hall, 1991). This filter was developed in collaboration with the History Department at Southampton University, who are interested in computer aided learning. Students are set questions and browse the hypermedia network to investigate the background material. They can then submit an answer to the HiDES (Historical Document Expert System) filter. The filter analyses the answer and generates a suitable response which may include links to further relevant documents.

The above filters show just some of the possible extensions that can be provided within the Microcosm framework. For example, we also have a number of filters used for development work that monitor, display and log the messages that pass through the filter chain. There are many other possibilities, a synonym generator applied to selections could expand the number of links found in a sparsely linked system, whilst an intelligent filter could build a context model of the user's requirements and restrict inappropriate links in a heavily linked system to help reduce 'link overload'. Some other examples can be found in (Davis et al, 1992a). The next section presents in detail the provision of navigation facilities within our filter model showing how it allows a multitude of additional and interchangeable functions to be incorporated.

Case Study : Filters for Navigation

Navigation in general.

Applications developed in open re-configurable hypermedia systems are destined to be far more complex and detailed than their closed counterparts, for the simple reason that as their name suggests their structure and content are able to dynamically change. Various structural analysis methods can be employed to reduce this complexity, for example hierarchies, metrics and clustering as discussed in (Botafogo & Mossé, 1992). However, even with this reduced complexity it is likely that navigation tools will still be needed. Since it has been proposed that navigation tools play an important part in the use of closed systems (Hutchings et al, 1992), it

would therefore seem highly likely that their role will be essential in more complex systems.

Navigation tools within hypermedia environments attempt to enhance the user's cognitive model of the information (Marchionini & Schneiderman, 1988) by increasing spatial awareness of their current position, and more importantly the position of the user's goals within the system.

The following is a brief taxonomy of existing navigation devices, that will provide the user with such cognitive navigation clues, more detail is provided in (Wilkins & Hall, 1992).

Direct access tools, (Bourne et al, 1989; Nielsen, 1990b; Yankelovich et al, 1988; Gloor, 1991). These provide a one to one mapping between the documents within the system and some form of unique icon set. Link information between documents can also be presented to the user.

History tools, (Bernstein, 1988; Mayes et al, 1988; Nielsen, 1990a; Nielsen, 1990b). Two cognitive clues are presented to the user, the user is able to retrace their steps (analogous to flipping back through a book), and the tool is able to prompt the user when they view the same document repeatedly.

Tours, (Hammond & Allinson, 1988; Trigg, 1988; Zellweger, 1989). These temporarily remove the problem of navigation within a hypermedia environment by providing a linear pathway through the non-linear hyperspace. The user can be allowed to branch from the tour at any time, in which case the tour provides a backbone for exploration instead of a rigid corridor which must be followed.

Guide Mechanisms, (Laurel, 1990; Laurel et al, 1990; Mayes et al, 1988; Oren et al, 1990). Such mechanisms attempt to pre-empt the users next document choice. The tool builds a user model which is enhanced as the user progresses through the data space. The system merely advises the user who is not bound by its suggestions.

Text retrieval, (Salton, 1989; Salton & Buckley, 1989; Frisse, 1988a; Frisse, 1988b; Li et al, 1992). This technique effectively supplements the pre-authored link information embedded within a hypermedia application. Instead it uses the inter-node relationships provided by text analysis of the document content. The user is provided with the system calculated relevant information in addition to that provided by authored links.

The following section describes how examples of the above navigation tool types have been implemented within the Microcosm system as filters.

Navigation Filters in Microcosm.

The functionality of any navigation filter that requires an accurate and complete overview of the information and connections contained within the system will be drastically reduced with the introduction of

open, reconfigurable architectures. Expansion of the system at any point implies the filter information will need constant recalculation. In contrast if the filter simply relies on previous navigation by the user then any such extension will have no effect.

The following properties are necessary for a navigation device implemented as a Microcosm filter to maintain a high proportion of its functionality when the more complex systems are introduced:

Low author overhead for the set-up and maintenance of the filter.

The filter's knowledge should be instantly extensible to cope with the expansion of the hypermedia information.

The filter should be able to function with an incomplete overview of the system information.

If the navigation tools are implemented as filters they have access to all the messages passing throughout the system. These messages contain a great variety of information, for example document names, whether the document is opening or closing, or the selection the user followed the link from, etc.

As the navigation tools are implemented as filters the user has full control over which ones they wish to use. The tools have to be independent of each other. They can however be constructed so that they utilise shared information if other filters are currently available within the system.

The workbench aspect of the system is further displayed by the way in which different navigation tools can be written that essentially provide the user with the same underlying functionality, for example the dispatcher and the enhanced memory dispatcher filters.

Filters are ideally suited for implementing navigation tools, in most cases they simply have to monitor the system messages, selectively store them and redispatch them when necessary. For example a history filter can be constructed. It simply has to store all messages regarding the opening of documents that it receives in chronological order. The relevant messages are reconstructed from this list and sent through the filter network whenever the user wishes to look at a previously viewed document.

If a navigation filter needs messages that do not already exist within the current model they can be dynamically added. Nothing else in the system will need to be changed, as the other areas simply ignore any messages that they do not understand.

The following navigation filters have been implemented within the Microcosm model. As described above the user's environment is completely reconfigurable, they are free to choose all or some of the tools described below.

History tool.

The user is able to retrace their steps one document at time by simply clicking on the icon displayed by the history filter with the mouse. By

double clicking on the filter they can gain access to a complete list of documents, selecting any of these will take them to that document. The filter can also inform the user when they repeatedly view the same document, which is a simple indication of being 'lost in hyperspace'. The user can preserve any history for use in a later session, thus providing a simple form of continuity between sessions. History files can also be used as guided tours, or mimics as they are known within the Microcosm system. One extra feature is that the history tool gives a visual indication of which documents are currently open on the desktop.

Mimic, guided tours.

They are implemented as the following modules.

Mimic engine is implemented as a filter, it is the back-end half of the mimic process.

Mimic viewer this is implemented as a viewer, it is the front-end half of the mimic process.

M*imic generator*, this allows the user to produce mimics in a variety of ways. It is implemented as a stand alone application that can connect to Microcosm should it be available within the current environment.

Any document type can be inserted in a mimic, any application can be used as a viewer, although full control over these viewers can only be provided if the application is completely Microcosm aware. The mimic progression method is pre-set by the author. It can be controlled by the user, for example for text documents. The system can control it so that dynamic media documents can be automatically sequenced by the mimic engine and the relevant viewer sends messages to the engine filter. The computer can also control it so that any document can be displayed for a pre-set time limit. The three control types can be freely mixed within a mimic, or even within one document. Documents can be displayed in sequence or in parallel, mimics can be nested to any level, linked to or from. Communication between the mimic engine and viewer is by means of custom messages that only they understand.

Short term memory dispatcher.

This tool provides a list of destination documents for any given link or collection of links. The user simply selects the document they wish to view from the list. Because it is implemented as a filter the tool works by intercepting and sieving certain messages. The basic facilities for this tool have been extended by incorporating a 'short term memory', similar to the idea implemented in (Briggs et al, 1991). This can be set to any reasonable length in both time and number of documents. The tool then uses variations in grey levels to show the placing in the memory of the destination documents in the current list. This gives the user some form of visual indication concerning the 'length of time' elapsed since the document had first been viewed.

Local maps.

In Microcosm the local maps are implemented as two separate applications :-

The map viewer. This is implemented as a Microcosm filter which allows the user to interact with the maps so that they can use them as a direct access method to the documents which they cover. As the user discovers new links the information contained in the map system is dynamically updated. The filter displays the local map for the document currently being read. This map is updated if the current document or any of its children changes. The user can call up a map for any document, or navigate through the local map network with a number of mouse clicks. The maps update themselves by monitoring various subsets of messages flowing through the filter network.

The map generator. This tool is implemented as a separate application. The user can specify the type of links that the map portrays, and the application can also be used to update, analyse or verify a previously constructed set of local maps.

The maps not only show the user relationships within the information that they have discovered, but also links from this body of information not yet followed. A map of any particular document will show not only links out of the parent to the children, but also links in from the children to the parent.

The use of filters to implement navigation tools within the Microcosm system allows a very powerful and easily reconfigurable navigation system to be constructed quickly and in a modular fashion.

Future work.

As stated previously, local and global map type tools are destined to lose a proportion of their functionality in the move towards open reconfigurable systems. As such, future navigation tools within the Microcosm system will have a heavy bias towards the guide type mechanisms detailed above. As filters they will be able to monitor the user's movement through the system, they will be able to construct a user model, and use this in conjunction with AI techniques to offer advice to the user in their passage through the available information.

Future Development

Dynamic Filter Management

Whilst the simple chain filter topology works well, we have often encountered situations where its very simplicity can be restrictive. In particular, its linear nature makes the filter order crucial to the correct behaviour of the system. For example the dispatch filter must be located after the linkbase filters in order to receive dispatch messages, whilst the linker filter needs to be before the linkbase so that the linkbase will receive create link messages.

Normally the filters will be defined in a configuration file, however if the user decides to add a filter, they must work out where in the chain to place it. This is generally straightforward with a little basic knowledge of the way the system works. It can be a little harder with more subtle functions such as the monitoring of messages carried out by navigation filters. Also, the chain can be quite inefficient as all messages pass through all active filters whether the filter processes that type of message or not. The message flow in the filter chain topology is shown in figure 2 below.

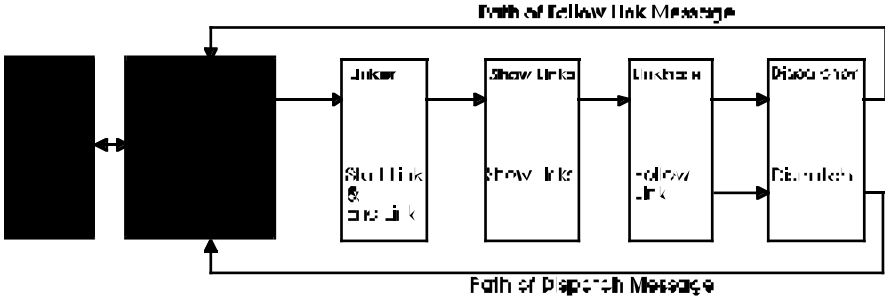

Figure 2 : Message flow in the current Filter Management System

To overcome these problems, we are investigating more advanced filter topologies. For example, by making filters publish the type of messages they understand, the FMS can build a table of actions that can be processed, and filters that will process them. On receiving a message with a particular action, it can be intelligently routed to only those filters that will understand it. In this way, the filters do not need to be explicitly ordered by the user, and message flow is much reduced as shown in figure 3, where the same user message to the same set of filters results in far fewer messages being passed to and from filters.

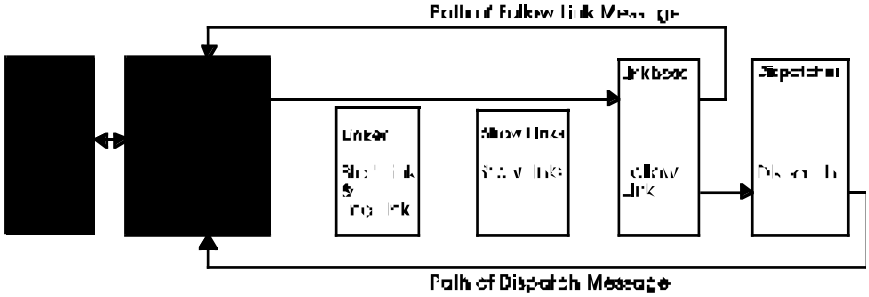

Figure 3 : Message flow in the new Filter Management System

The various methods for intelligent filter management and their implications are addressed in (Hill & Hall, 1992).

Distributed Systems

While we consider Microcosm able to provide a framework for the integration of a users information into a coherent environment (Davis et al., 1992b), this is currently limited to the user's local machine. Remote information can be used, but only if mounted as a virtual file system e.g. using PC-NFS or other network software. To provide a truly useful information environment, a distributed version of

Microcosm is needed, where both information and filter functions can be shared by networked workstations. Ultimately this will include various hardware platforms such as the Apple Macintosh, and UNIX based X-Windows workstations for which versions are currently under development. By using the Microcosm message format in conjunction with a predefined protocol for communication between Filter Management Systems, the move from a homogenous system to the heterogeneous system described above should be straightforward.

Conclusions

In this paper we have presented a model for an open and reconfigurable hypermedia system that is based on the concept of filters. The resulting environment allows the user to reconfigure the database of available links and other aspects of the system's functionality dynamically, which means that in the same session they can view the data from many different perspectives, in many different ways.

We have also shown that the functionality of the hypermedia system can be extended easily through the use of filters. In particular, we discussed the requirements for navigation tools in an open hypermedia system and showed how these have been implemented as filters in a very straightforward way. Debugging, testing and evaluation are also easier in the filter-based model. The concept of filters has been presented in the context of an open model for hypermedia but many of the results apply equally well to a closed system.

Bibliography

Bernstein M., (1988) The Bookmark and the Compass. In ACM Bulletin 9, 4 of the Office Information System Group, (October 1988), 34 - 45

Botafogo R. & Mosse D., (1992) Analysing Hypertext Structures. In Proceedings of HyperOz '92, A Workshop on Hypertext Activities in Australia., February 21, 1992., Ed. Hunter-Thomas B., University of South Australia Level Campus.

Bourne J. R., Cantwell J., Brodersen A.J., Antao B. & Huany Y. C., (1989) Intelligent hypertutoring in engineering. In Academic Computing (Sept) 42 - 49

Briggs J. H., Tompsett C. & Oates N., (1991) Intelligent Hypertext Resources for Professional Updating. In Proceedings of the 6th International PEG Conference Knowledge Based Environments for Teaching and Learning. (Rapallo, Italy, May 1991)

Colson F., Hall W., (1991) "Multimedia Teaching with Microcosm-HiDES: Viceroy Mountbatten and the Partition of India," to be published in History and Computing 3(2)

Creech M. L., Freeze D. F. & Griss M. L., (1991) Using Hypertext in Selecting Reusable Software Components. In Hypertext 91: Proceedings of Third ACM Conference on Hypertext (San Antonio, TX Dec. 15-18) 25 - 38

Davis H., Hall W., Hutchings G., Rush D. & Wilkins R., (1992a) Hypermedia and the Teaching of Computer Science: Evaluating an Open System. In Proceedings of the first Conference on Developments in the Teaching of Computer Science. University of Kent, 1992.

Davis H., Hall W., Heath I., Hill G., Wilkins R., (1992b) Towards an Integrated Information Environment with Open Hypermedia Systems. Computer Science Technical Report, University of Southampton, U.K., 1992. (In Preparation)

Faloutsos F., Lee R., Plaisant C., Shneiderman B., (1990) "Incorporating String Search in a Hypertext System: User Interface and Signature File Design Issues," Hypermedia, 2(3), 1990, pp 183-200.

Fountain A., Hall W., Heath I. & Davis H., (1991) Microcosm: an Open Model With Dynamic Linking. In Hypertext: Concepts, Systems and Applications (Proceedings of the European Conference on Hypertext, INRIA, France, November 1990) Rizk, A., Strietz, N. & André, J Eds. 298 - 311

Frisse M. E., (1988a) From text to hypertext. Byte (Oct.) 247 - 253

Frisse M. E., (1988b) Searching for information in a hypertext medical handbook. In the Communications of the ACM (month ?) 880 - 886

Gloor P. A., (1991) CYBERMAP - Yet another way of navigating in hyperspace. In Hypertext 91: Proceedings of Third ACM Conference on Hypertext (San Antonio, TX Dec. 15-18) 107 - 122

Halasz F. G., (1988) Reflections on Notecards: Seven issues for the next generation of hypermedia systems. In the Communications of the ACM 31, 7, (July 1988) 836 - 852

Halasz F. G., (1991) Seven issues: revisisted. In Hypertext 91: Proceedings of Third ACM Conference on Hypertext (San Antonio, TX Dec. 15 - 18), Closing keynote address

Hall W., Heath I., Hill G., Davis H. & Wilkins R., (1992) Microcosm: State of the Art. Computer Science Technical Report, University of Southampton, 1992. (In Preparation)

Hammond N., Allinson L., (1988) Travels around a learning support environment: rambling, orienteering or touring ? In CHI '88: proceedings of the fifth ACM-SIGCHI Conference on Human Factors in Computing Systems (Washington, D.C., May 15-19) Soloway, E., Frye, D. & Sheppard, S.B. Eds. 269 - 273

Heath, I. (1992), An Open Model for Hypermedia: Abstracting Links from Documents, Ph.D. Thesis, University of Southampton, UK, 1992. In Preparation.

Hill G., Hall W., (1992) Microcosm : Intelligent Filter Management, Computer Science Technical Report, University of Southampton, U.K., 1992. (In Preparation)

Hutchings G. A., Hall W. & Colbourn C. J., (1992) A Quantitative Study of Students' Interactions with a Hypermedia System. In Computer Science Technical Report, University of Southampton, U.K., CSTR 92-04, 1992.

Laurel B., (1990) Interface Agents: Metaphors with character. In The art of human computer interface design Laurel B Ed. Addison Wesley 355 - 366

Laurel B., Oren T. & Don A., (1990) Issues in multimedia interface design: media integration and interface agents. In Empowering people: CHI '90 conference proceedings (Seattle WA, April 1-5) Carrasco Chew, J. & Whiteside, J. Eds. 133 - 139

Li Z., Hall W. & Davis H., (1992) Hypermedia links and information retrieval. In Proceedings of the 14th British Computer Society Research

Malcolm K. C., Poltrock S. E. & Schuler D., (1991) Industrial Strength Hypermedia: Requirements for a Large Engineering Enterprise. In Hypertext 91: Proceedings of Third ACM Conference on Hypertext (San Antonio, TX Dec. 15-18) 13 - 24.

Marchionini G. & Schneiderman B., (1988) Finding facts vs. browsing knowledge in hypertext systems. IEEE Computer 21(1), 190 - 196Colloqium on Information Retrieval., Lancaster University 1992.

Maurer H., Tomek I., (1990) Broadening the scope of Hypermedia Principles. Hypermedia 2(3), pp 201-220.

Mayes J. T., Kibby M. R. & Watson H., (1988) StrathTutor: The development of a learning by browsing system on the Macintosh. Computers in Education 12, 221 - 229

Nielsen J., (1990a) Hypertext and hypermedia. Academic Press, London

Nielsen J., (1990b) The art of navigating through hypertext. Communications of the ACM 33, 298 - 310

Oren T., Solomon G., Kreitman K. & Don A., (1990) Characterizing the interface. In The art of human computer interface design Laurel B Ed. Addison Wesley 367 - 382

Parunak H. Van Dyke, (1991) "Don't Link Me In: Set Based Hypermedia For Taxonomic Reasoning," In: Hypertext '91, Proceedings of the 3rd ACM Conference on Hypertext, San Antonio, TX, 15-18 December 1991, pp 233-242

Pearl A., (1989) Sun's Link Service: A Protocol for Open Linking. In Hypertext 89 Proceedings (Pittsburgh PA, November 5-8) 137 - 146

Salton G., (1989) Automatic text processing. Addison Wesley

Salton G. & Buckley C., (1989) On the automatic generation of content links in hypertext. Cornell University, Ithaca NY Technical report TR 89 - 993

Shipman F. M. III, Chaney R. J. & Gorry G. A., (1989) Distributed Hypertext for Collaborative Research: The Virtual Notebook System. In Hypertext 89 Proceedings (Pittsburgh PA, November 5-8) 129 - 136

Trigg R. H., (1988) Guided Tours and Tabletops: Tools for Communicating in a Hypertext Environment ACM Trans. Office Information Systems 6(4), 398 - 414

Wilkins R., Hall W., (1992) Implementing Open Hypermedia Navigation Tools Using Filters, Computer Science Technical Report, University of Southampton, U.K, 1992.

Yankelovich N., Landow G. & Heywood P., (1988) Designing hypermedia "ideabases" - The Intermedia experience. Brown University, IRIS technical report 87 - 4

Zellweger P.T., (1989) Scripted documents: hypermedia path mechanism. In Hypertext 89 Proceedings (Pittsburgh PA, November 5-8) 1 - 14