UNIVERSITY OF SOUTHAMPTON

DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

AGENTS FOR DISTRIBUTED MULTIMEDIA INFORMATION MANAGEMENT

David C. DeRoure, Wendy Hall,

Hugh C. Davis and Jonathan Dale

University of Southampton

April 1996

Presented at PAAM'96

Agents for Distributed Multimedia Information Management[1]

David C. DeRoure, Wendy Hall, Hugh C. Davis and Jonathan Dale

Multimedia Research Group

Department of Electronics and Computer Science

University of Southampton

UK

{dder, wh, hcd, jd94r}@ecs.soton.ac.uk

This paper discusses the role of agents in a distributed multimedia information system (DMIS) engineered according to the principles of open hypermedia. It is based on the new generation of Microcosm, an open hypermedia system developed by the Multimedia Research Group at the University of Southampton. Microcosm provides a framework for supporting the three major roles of agents within open information systems: resource discovery, information integrity and navigation assistance. We present Microcosm and its agents, and discuss our current research in applying agent technology in this framework.

1. Introduction

Whether we look at corporate information systems or the Internet, it is clear that the volume of information available is increasing. Users require assistance to avoid being overwhelmed by this wealth of information and information providers require assistance in authoring and maintaing it. The large volume of highly dynamic, distributed information involved is an ideal candidate for agent-style processes. These agents will assist the user as they interact with the distributed multimedia information system (DMIS) of the future: they will help the user to discover new sources of information, to navigate the vast information space and to author and maintain the information.

During the last six years the Multimedia Research Group at the University of Southampton has been developing an open hypermedia system called Microcosm [8,4,11,12,13,3,21]. Microcosm allows its users to integrate any disparate pieces of information into a cohesive web of documents that are viewed with the user's everyday applications (for example, Microsoft Word). It is essentially an open framework for agents that control and relate the individual documents, and it integrates and extends the World Wide Web [1].

Microcosm has been applied to a variety of projects, many in collaboration with industrial or commercial partners. In an educational context it is used for courseware delivery at a number of UK educational sites, and current projects include two major historical multimedia archives.

The concept of open hypermedia is presented in the next section. This is followed by an explanation of the Microcosm model, which extends the linkbases of open hypermedia to arbitrary agents, in section 3. This is followed in section 4 by a description of two classic Microcosm agents, then a discussion of agents for distributed information management in section 5. Interface issues are discussed in section 6, and language issues briefly in section 7.

2. Open Hypermedia

The World Wide Web (WWW) is a distributed hypermedia system: hypermedia because it deals with documents (nodes) of various media types (such as text and images) and provides `hyperlinks' between source anchors (locations in documents) and destination anchors (other documents, or possibly locations in them). Although it is an open system in many respects, the way it is commonly used is

[1] This research was funded by an EPSRC Research Studentship and partially funded by EPSRC grants GR/K36409 and GR/K30773.

classified as *closed* hypermedia. This is because it contains just one major document format that supports links (HTML), and those links are embedded within the HTML documents.

Open hypermedia is a concept that has been established in the hypermedia community for some time [4]. A key architectural feature of open hypermedia systems is that the information about the links is stored separately from the documents; the sets of links are called link databases or *linkbases*. The linkbase concept can be abstracted out further and viewed as a link service [5,2].

The Workshop on Open Hypermedia at the ACM conference in Edinburgh (1994) defined open hypermedia as follows (due to Davis):

- It should be possible to import new nodes, links, anchors and other hypermedia objects without any limitation, to the size of the objects or to the maximum number of such objects that the system may contain, being imposed by the hypermedia system;
- The system should allow the import and use of any data format, including temporal media;
- The system should allow any application to access the link service in order to participate in the hypermedia functionality;
- The hypermedia system should not impose a single view of what constitutes a hypermedia data model, but should be configurable and extensible so that new hypermedia data models may be incorporated. It should thus be possible to interoperate with external hypermedia systems, and to exchange data with external systems;
- It should be possible to implement the system on multiple and distributed platforms;
- The system must support multiple users, and allow each user to maintain their own private view of the objects in the system.

A number of features of open hypermedia are a direct consequence of designing the system with separate links ('separable hyperstructure'). Documents can still be edited by the applications that created them and links can be applied to read-only documents, such as those commonly found on the Internet or on a CD-ROM. Different users can use different sets of linkbases at different times as befitting their requirements, that is, there is not a single, static view of the information space. Maintenance of link integrity is more straightforward because only the linkbases need be updated: in a closed system, a change to one (destination) document might require multiple (source) documents to be updated. It is easy to create new sets of links, private or shared, while the documents remain in their 'native' formats. Finally, one set of linkbases can be used to assist the author in building a new linkbase for publication; that is, it facilitates the authoring process.

3. The Microcosm Model

Two of Microcosm's most powerful features are its ability to overlay links into any document, irrespective of type, and the fact that it is constructed from a collection of simple processes working together. In Microcosm, a linkbase is just an instance of a link resolution agent and many other types of link processing agents (know as *filters* in this model) are possible: new filters can be constructed and slotted into the existing system. The majority of Microcosm's hypermedia functionality is embodied in two filters: a link creator and a linkbase.

Microcosm can be considered as a community of agents, a group of processes working together for a common aim. Agents within a system such as Microcosm can either have their own interface, or can be controlled through interaction with the user's everyday tools. Agents in Microcosm are sub-divided into two classes, *viewers* and *filters*: viewers allow the user to interact with the information in the system and filters provide a variety of functions ranging from navigation tools to link construction and storage tools. This model can be seen in figure 1.

Each agent in the Microcosm system can communicate with other agents through the use of a dedicated messaging mechanism. The current message format is constructed from plain ASCII, sub-divided into named sections (*tags*) and any process can add any tag to a message. If an agent encounters tags which it does not understand it will simply ignore them. Such a system provides great deal of flexibility as new tags can be added without needing to change the majority of the system.

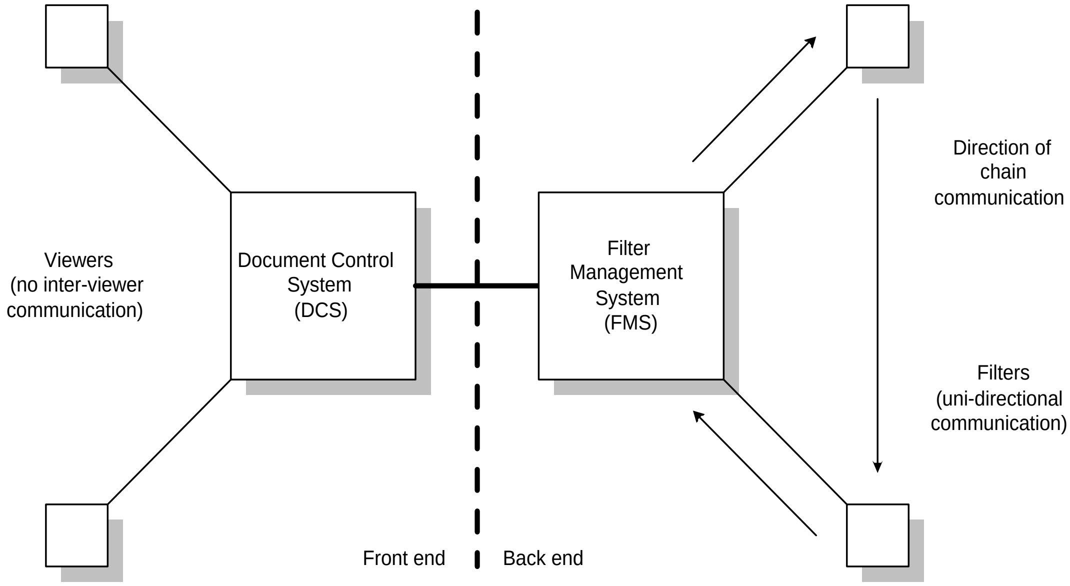

Figure 1 : The Microcosm Model

The communication model underlying Microcosm has evolved throughout the project. Early implementations were standalone with strict communication paths between agents; this model was subsequently relaxed to permit direct communication between agents, and extended to permit the agents to exist anywhere on the network in a loosely coupled system.

Many of the Microcosm applications demand whole or partial automation of some aspects of using the system, be it authoring, maintenance or navigation. For example, the assimilation of new documents into the system needs to be a purely mechanical process for historical archives, while some business processes involve information changing in real time which cannot be monitored manually. This automation is achieved by extending the system with new agents dedicated to a particular application.

4. Example Agents

The basic Microcosm system provides a small number of filters, such as the linkbases mentioned earlier. New agents can be added to meet the requirements of particular applications. One such agent is the *Prolog filter* which was built for a specific project but is an example of a generic technology. Another is the *Advisor Agent*, a general purpose tool which serves to illustrate the advantage of adopting common communication interfaces between the agents in the Microcosm system. These two agents are described below.

4.1. The Prolog Filter

A filter that interfaces Microcosm to a Prolog knowledge base has been developed as part of a three year EPSRC funded project in conjunction with Pirelli Cables [3]. The construction of this filter embodies the traditional view of an agent as a process capable of deductive reasoning over a particular knowledge domain. In the case of the Pirelli application, the filter embodies knowledge about the maintenance procedures for a large cable packaging machine. The user interacts with the filter through a viewer specifically designed for the task. This presents the user with a list of symptoms for a particular fault. If the symptoms match those of the real machine then the filter uses its knowledge base to suggest solutions to the current problem. The knowledge base can be updated by the user to cover new faults and symptoms, learning about the relationships as they are discovered by the current user.

The filter is provided as part of a functionally enhanced electronic representation of the existing manuals for the cabling machine. The user is able to see the symptoms in the context of related material contained in this manual. Furthermore, the knowledge based agent filter is able to present the solutions to the current problem in the context of the same information.

The agent exists as a self-contained entity; it constructs and maintains its own knowledge base and it does not interact with other processes in the system during the deduction process. However, the burden of presenting information to the user (both in terms of query symptoms and solutions) is removed from the filter. Both tasks are achieved by issuing commands to the rest of Microcosm via the message system.

4.2. The Advisor Agent

The Advisor agent [22] acts as an arbitrator between the user and the plethora of navigation processes provided by Microcosm. This agent presents the user with a navigational overview of their current situation. This overview is generated by querying all available sources of navigation information within the Microcosm system (for example, the linkbases, the full-text retrieval filter and local map filter). Each source responds with a set of advice, the agent then combines these disparate advice sets into one overall advice strategy.

The agent can be dynamically configured to act either autonomously or semi-autonomously. The user entrusts the agent to query each available advice source, combine the advice and present it. The alternative procedure would require the user to perform the querying and combination themselves; the agent is relieving the user of the burden for this task.

The majority of the advice sources are constructed by adapting existing processes within the current Microcosm system, although some are implemented as completely new navigation tools. All advice sources are capable of functioning on their own and do not need the agent, but in this mode the user must interact with each individually. When the agent is activated, it provides an interface between the user and the disparate advice sources.

5. Agents in DMIS

We now address agents in the particular context of *distributed* multimedia information systems, such as the recent versions of Microcosm [6]. The Internet itself is an example of such a system; large corporate networks are another. The role of agents within open DMISs, such as the Internet, can be sub-divided into three categories, which are described in the following sections.

5.1. Resource Discovery

This is probably the most common conception of an agent. A resource discovery agent is a process that, either semi-autonomously or autonomously, searches for and presents new sources of relevant information to the user. How relevant this information is depends upon the accuracy of the agent's analysis process. As more information is made available electronically, more such agent processes will be required.

Examples of simple implementations of this type of process (usually working semi-autonomously) can be seen in any WWW search robot. These mechanisms can are implemented as two separate processes; the first searches for WWW information and indexes its content, and the second accepts search terms from the user and responds with relevant material.

Autonomous variations on this procedure could be designed so that agents periodically query the indexes with a constant search term, providing the user with results when located. A better implementation would only provide the user with new results discovered by the last search (as opposed to all results each time). This idea could be extended to allow the agent to constantly monitor the user's interaction with the WWW, in an effort to construct queries automatically to use on the index. The results of the automatic queries would then be presented to the user.

The current WWW resource discovery agents are text based; they construct inverted text indexes from the content of the HTML documents. However, by using technology similar to that under development in the MAVIS system [23,24,14] there is no reason why the agents should not be extended to incorporate other media.

5.2. Information Integrity

The distributed nature of open information systems such as the WWW leads to problems in information integrity. For example, the destinations for the majority of links are hard coded into the HTML documents that contain the link sources. Should the document at the destination of a link be moved (or deleted) then the link will be left dangling. When a user attempts to follow such link, they will be greeted by some form of error.

Microcosm provides a partial solution to the dangling link problem by storing all links separately in link databases. Although this does not actually solve the problem, it opens the way for solutions to be implemented. If the links are all stored centrally in link database, then a process only needs to interact with this database to locate any dangling links. In the WWW the process must examine all of the HTML documents to locate the dangling links.

The other problem with integrity involves the indexes generated by the resource discovery agents. Suppose a document is indexed then after indexing is edited; there is now a integrity problem between the index and the document itself. This problem can only be removed by re-indexing the documents concerned.

Both integrity problems could be solved by agents autonomously interacting with the WWW servers and any indexing processes. If an integrity problem is located, the agent could fix it by locating the missing document on the server or re-indexing the edited documents. If the agent could not fix the integrity problem then it could flag the user for some help (at some convenient point).

The implementation of such repair agents may require additional functionality to be incorporated into the WWW servers. The size and complexity of the WWW means that it is impossible to check information integrity when the documents are being edited and linked (as is possible in a closed hypermedia system). The use of clean-up agents would seem to present an alternative solution to the problem. The system would not guarantee that the information was 100% correct at all times, but it would be able to cope with the majority of integrity problems as they are discovered.

5.3. Navigation Assistance

An increase in available information is likely to produce an increase in resource discovery agents. It is also likely that no-single agent will become dominant, it would seem that some form of arbitrating agent (such as the Advisor agent) will be required. Without such an intermediary agent the user is just as likely to be swamped with advice as they are with original information.

These agents would interact with available resource discovery agents integrating the disparate results and presenting the user with a global navigation strategy. Oren [15] summed up this role of an agent within an open MMIS as '...A parallel would be the human reference librarian who does not comprehend the material in articles being sought, but does understand the conventions of card catalogues, abstract collections, citation indexes and bibliographic references...'. In a similar manner to a trained librarian they would make use of the analogous tools to provide the user with guidance without necessarily understanding the subject of the searches.

6. Agent Interfaces

For agents implemented within the Microcosm system there are two choices as far as user interface mechanisms are concerned. Firstly, they can provide the user with dedicated interfaces, specifically tailored for their needs. Secondly, they can adopt principles such as outlined in the Envoy framework [16]. This suggested that existing applications could be adapted allowing users and agents to interact with each other through familiar interfaces. For example, the user would issue a query to the agent through the use of the interface of their favourite database application. In return, the agent would use the same application interface to present the user with the results of its searches.

The Envoy framework also suggested that the back-end of the application could be driven by the agent. The agent would effectively become an automated interface between the front and back-ends of many applications. Such interaction would be achieved with use of set Application Programming Interfaces (APIs). Separate APIs would be designed for each interface, for example the interface between the agent and the front-end of the application would be different to that between the agent and the back-end of the same application.

Microcosm allows both interface mechanisms to co-exist and any process within the Microcosm system can have its own interface. But, in addition, it is common for the user to interact with Microcosm processes through their everyday tools (for example, Microsoft Word). The APIs in the Envoy framework are replaced by shared sets of messages in the Microcosm system. For example, the user could control an agent through any application that understood its shared set of messages. In the current system, the user can interact with the linkbases (following links, etc.) from any application that can generate the appropriate message sets.

Given the above, it is conceivable that the naive user would never actually need to interact directly with any agents. Instead they would control them through their everyday applications. This is highly parallel to the Microcosm system, where the naive user does not need to interact directly with the back-end processes, instead all interaction is achieved through their usual applications.

7. Agent Implementation

The experience of the Microcosm project confirms that open hypermedia is a powerful approach to engineering distributed information systems. We have also found that the open model requires more processing to occur 'behind the scenes', and hence the choice of agent technology is particularly important.

Our current projects involve many new agents for DMIS. For interoperability it is important that we establish the appropriate standard interfaces between agents, but the implementation language of individual agents may vary. Some agents will need to be mobile (a concept perhaps most familiar in the context of resource discovery), which typically requires an interpreted language for heterogeneity and due attention to safety.

For the communication language we have currently adopted KQML [7]. This language consists of three layers:

- *Content layer*. KQML contains no definitions for the content of this layer, it can be any structure ASCII or binary.
- *Message layer*. Used to encode a message that one process sends to another. The message can be one of two types: *content,* containing information being offered or sought, or *declaration* announcing a process, registering its name or providing descriptions of the general types of information that the process will send/receive.
- *Communication layer*. A wrapper around a message which specifies some communication attributes, such as a specification of the sender and recipient processes.

The outer layers of the language allow the agent processes to route messages without actually needing to understand the message content. The process descriptions contained within the communication layer can range from physical network addresses to service descriptions. Arbitrating agents called *facilitators* can route messages sent with service descriptions to the appropriate agents, these agents having previously registered with the facilitators.

Of the available scripting languages for implementing agents, the following three are perhaps best known[1]:

- *Telescript* [9,20], created by General Magic Inc., provides a powerful scripting language that is specifically designed to handle the types of network communications required by agents. Agents can meet at specified locations (called *places*) and physically move from place to place during their execution. The scripts that represent the agent are interpreted at the various locations by the local Telescript engine. (http://www.genmagic.com/)

[1]There are many other scripting languages becoming established. We have also evaluated the following:

- *Guile*. (http://www.cygnus.com/library/ctr/guile.html)
- *Obliq*. (http://www.research.digital.com/SRC/Obliq/)
- *Python*. (http://www.python.org/)
- *Scheme*. (http://www.ai.mit.edu/projects/su/)
- *ScriptX*. (http://www.kaleida.com/)

- *Tool Command Language* (TCL) [17,20], created by John Ousterhout, provides a general purpose scripting language that can be used to construct agent processes. The language is freely distributed and has been adopted by Sun Microsystems as the scripting language for the Internet. It is likely that TCL will prove to be a good language for designing agents. (http://www.sunlabs.com/research/tcl)
- *Java* [10]. This language is supported within WWW browsers including Hot Java and the Netscape Navigator. It allows static HTML documents to be brought to life by implementing the capability to add arbitrary client-side behaviour by downloading Java programs (*applets* and *applications*). (http://java.sun.com/)

We have opted for Java as our primary implementation language as we believe it has the appropriate language features required by our applications (distribution, portability, dynamic extensibility, etc.) and it promises wide availability. We are also working with distributed symbolic languages [18] as these are particularly well matched to dealing with dynamic, unpredictable computations and to building abstractions that enable programmers to work at a higher level. Recent work in this area includes addressing the language issues in code mobility [19].

8. Conclusion

We have presented the Microcosm model as an agent architecture for distributed multimedia information systems. Agents implemented within the Microcosm system immediately have the functionality provided by the underlying system at their disposal. Microcosm's ability to store the documents and relationships separately will ease the implementation of agents that fulfil the information integrity role discussed previously.

As the Microcosm philosophy is applied to new projects we will build generic Microcosm agents customisable to the task at hand, focusing on:

- Resource discovery,
- Information integrity, and,
- Navigation assistance.

The open hypermedia approach has many significant advantages, and agents are the key to handling the extra distributed work that must go on to realise these advantages to their full extent.

9. Acknowledgements

Many of the members of the Multimedia Research Group have contributed to the work described in this paper. In particular, Gary Hill redesigned the filter model and implemented the first distributed filters, Rob Wilkins introduced a new communication model and pioneered the Microcosm agent work through the Advisor Agent, and Ian Heath built the Prolog Filter.

References

[1] BERNERS-LEE, T., CAILLIAU, R., GROFF, J., and POLLERMANN, B., World-Wide Web: The Information Universe. *In:* Electronic Networking: Research, Applications and Policy, 2(1), pages 52-58, 1992.

[2] CARR, L. A., DeROURE, D. C., HALL, W. and HILL, G. J., The Distributed Link Service: A Tool for Publishers, Authors and Readers. *In:* Fourth International World Wide Web Conference: The Web Revolution, Boston, Massachusetts, USA, December 1995.

[3] CROWDER, R. M., HALL, W., BERNARD, R. and HEATH, I., Open Hypemedia Systems for Training and Maintenance. *In:* European Conference on Automation and Robotics Training (November), London, 1993.

[4] DAVIS, H. C., HALL, W., HEATH, I., HILL, G. J. and WILKINS, R., Towards an Integrated Information Environment with Open Hypermedia Systems. *In:* Proceedings of the Fourth ACM Conference on Hypertext (ECHT '92), Lucarella, D., Nanard, J., Nanard, M. and Paolini, P., *Eds.*,

Milan, Italy (December), pages 181 - 190, 1992.

[5] DeROURE, D. C., Networked Multimedia: Hypermedia Link Services. *In:* Telecommunications (September), pages 117-120, 1995.

[6] DeROURE, D. C., HILL, G. J., HALL, W. and CARR, L. A., A Scaleable, Distributed Multimedia Information Environment. *In:* MediaComm'95, Southampton, UK (April), pages 77-85, 1995.

[7] FININ, T., KQML - Knowledge Query and Manipulation Language, 1994.

[8] FOUNTAIN, A., HALL, W., HEATH, I. and DAVIS, H. C., Microcosm: An Open Model With Dynamic Linking, *In:* Hypertext: Concepts, Systems and Applications, Proceedings of the European Conference on Hypertext, Rizk, A., Strietz, N. and André, J., *Eds.*, INRIA, France, pages 298-311, November 1990.

[9] General Magic, Telescript Technology: The Foundation for the Electronic Marketplace, 1993.

[10] GOSLING, J., Java Intermediate Bytecodes. *In:* ACM SIGPLAN Workshop on Intermediate Representations (IR'95), San Francisco (January), 1995.

[11] HALL, W., HEATH, I., HILL, G. J., DAVIS, H. C. and WILKINS, R. J., Microcosm: State of the Art. *In:* Computer Science Technical Report CSTR 92-18, Department of Electronics and Computer Science, University of Southampton, U. K., 1992.

[12] HEATH, I., An Open Model for Hypermedia: Abstracting Links from Documents, PhD Thesis, University of Southampton, UK, 1992.

[13] HILL, G. J., WILKINS, R. J. and HALL, W., Open and Reconfigurable Hypermedia Systems: A Filter-Based Model. *In:* Hypermedia, Vol. 5, No. 2, pages 103-118, 1993.

[14] LEWIS, P. H., DAVIS, H. C., GRIFFITHS, S., HALL, W. and WILLKINS, R. J., Content Based Retrieval and Navigation with Images in the Microcosm Model. *In:* MediaComm'95, Southampton, UK (April), pages 86-90, 1995.

[15] OREN, T., The Architecture of Static Hypertexts. *In:* Proceedings of Hypertext '87, November 1987, pages 291-306, Association for Computing Machinery 1989.

[16] PALANIAPPAN, M., YANKELOVICH, N., FITZMAURICE, G., LOOMIS, A., HAAN, B., COOMBS, J. and MEYROWITZ, N., The Envoy Framework: An Open Architecture for Agents. *In:* ACM Transactions on Information Systems, Vol. 10, No. 3, ACM Press (July), pages 222-264, 1992.

[17] OUSTERHOUT, J. K., TCL and the TK Toolkit, Professional Computing Series, Addison-Wesley, Reading Mass. and London (April), 1994.

[18] QUEINNEC, C. and DeROURE, D. C., Design of a Concurrent and Distributed Language, *In:* Robert, H., Halstead, Jr and Takayasu, I., *Eds.*, Parallel Symbolic Computing: Languages, Systems and Applications, Lecture Notes in Computer Science LNCS 748 (October), pages 234-259, Springer-Verlag, 1992.

[19] QUEINNEC, C. and DeROURE, D. C., Sharing Code through First Class Environments. *In:* International Conference on Functional Programming, 1996. To appear.

[20] WAYNER, P., Free Agents. *In:* BYTE (March), pages 105-114, 1995.

[21] WILKINS, R., HEATH, I. and HALL, W., A Direct Communication Model for Process Management in an Open Hypermedia System, Computer Science Technical Report CSTR 93-14, Department of Electronics and Computer Science, University of Southampton, UK, 1993.

[22] WILKINS, R. J., The Advisor Agent : A Model for the Dynamic Integration of Navigation Information within an Open Hypermedia System, PhD Thesis, University of Southampton, UK, 1994.

[23] WILKINS, R. J., GRIFFITHS S., LEWIS, P. H., HALL, W. and DAVIS, H. C., Content-Based Navigation within Microcosm for Multimedia Documents. *In:* ECHT'94, Proceedings of the Sixth ACM Conference on Hypertext, Edinburgh, Scotland (September), ACM Press, (presented as poster), 1994.

[24] WILKINS, R. J., GRIFFITHS, S., HALL, W., LEWIS, P. H. and DAVIS, H. C., The MAVIS Project -

Extending Generic Links and Content Based Retrieval to Non-textual Documents in the Microcosm Model. *In:* 1994 Research Journal, Southampton University, UK pages 167-169, 1994.