Link Services or Link Agents?

L. A. Carr, W. Hall, S. Hitchcock
Multimedia Research Group,
Department of Electronics and Computer Science
University of Southampton, Southampton, UK
Tel: +44 - (0) 1703 594479
E-mail: lac@ecs.soton.ac.uk

ABSTRACT
A general link service for the WWW has been used within an Electronic Libraries' project. Experience using it shows that as the links become increasingly interesting to the user, processing them becomes increasingly expensive. Eventually textual analysis, ontological services and remote database lookups conflict with the goal of prompt delivery of documents. This paper summarizes the history of the Link Service software behind the Open Journal project together with the kind of links that it has been used to produce. Building on this work it then discusses how the paradigm, architecture and user interface of the DLS have been newly modified both in response to user feedback and also to allow more linking facilities to be added to the WWW environment. We then introduce AgentDLS, an agent-style system that offers suggestions to help the user's browsing and information discovery activities.

KEYWORDS: Links, hypertext, open hypermedia, link services, autonomous user interface agents.

INTRODUCTION

The work described on the Distributed Link Service has been in progress since 1994 and has been previously described (most particularly in [5] and [2]). It is re-presented here in an updated form as an historical context for the implementation of the linking facilities together with the experiences and feedback that have shaped their development. In particular, we believe that it is this historical description which makes sense of the shift in perspective from a "link service" to a "link agent".

For those familiar with the DLS, the next section "History of a Link Service" contains no new information until the paragraphs on *Link Prioritization* and *Link Semantics*. The following section "Putting the Link Service to Work" contains newer material in the form of a summary of the various kinds of links that the DLS has been used to generate. The keyword links which have always been the bedrock of the DLS functionality have been previously described together with the citation links [14]. However, the new experiments with "person" and "concept" links help to fill in the spectrum of link-making strategies that point to a necessary evolution of the linking mechanism. This new mechanism is described in the section "A New Interface", which outlines the reasons for reworking the DLS in the style of an agent.

THE HISTORY OF A LINK SERVICE

The experience of the Microcosm project [6] in providing Open Hypermedia services for information resources led to experiments with providing those services for early WWW browsers. This allowed WWW documents to contain *generic* and *computed* links that were more flexible than the native HTML links. However, it led to the situation where the browser offered a porthole into the distributed data of the World Wide Web, but the hypertext links were maintained locally on the PC. To make use of the links required the user to obtain both the Microcosm linking software and the link databases themselves.

The open nature of the framework upon which the WWW is based (clients and servers communicating through well-defined text-based protocols) meant that it was possible to extend the hypertext model of the WWW to support an independent link service. In effect, the WWW infrastructure could be used as the communication framework for a distributed hypertext link service. The Distributed Link Service (DLS) was developed as such a system. It is able to work in conjunction with existing WWW resources to support an additional underlying link service. Like Microcosm, the DLS utilized a variety of link database processes to offer flexible hypertext functionality to a wide range of end-user applications.

A Simple Interactive Link Service

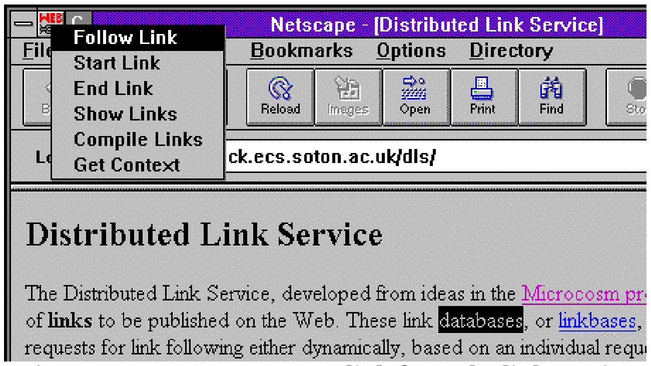
The DLS [5] was originally composed of two parts: the server facilities that were accessed via the WWW, and the client interface that worked in conjunction with a WWW browser.

Link Server: The link server facilities of the DLS were first implemented as CGI scripts invoked by a standard WWW server and subsequently as independent modules of a pseudo-WWW server. This pseudo-server interacted with clients as if it were a normal WWW server, using enough of the hypertext transport protocol to allow normal interaction with a browser, but it did not store or return any documents. Instead, modules were available to allow the creation, traversal and editing of links, stored in a number of link databases using SGML markup. Each entry in the database recorded the source and destination attributes of the link, the type of the link, its creation time and a link description.

There were several different link database categories supported by the system: at the most general level was the server database that applied whenever the system was queried. Link databases were also provided for particular documents. In addition, a variety of 'context' link databases were available for the user to select according to their current focus of interest. The user was also provided with a personal link database for private links to which only they had access.

The server received details from the DLS client of the user's selection, the document in which the selection was made, and the current context. The *followlink* module determined which link databases were required, and gathered these together to satisfy the request. The *editlink* module provided an HTML form that allowed the user to select from the available link databases and edit the individual links. The *createlink* module accepted details of end points for a link and entered the new link into a specified link database or into the user's personal link database. The *context* module provided a list of the different context link databases available on the server and was used by the client to present a menu to the user.

All of these modules were driven by an explicit command coded as a URL and sent to the link server as a normal HTTP request. Alternatively the server could return a form to the browser, and the user would fill in the fields to support a particular action. However, form-filling was never a convenient option for actions like creating a link or editing a large link database, simply because there were too many pieces of information to be entered manually (*e.g.* source and destination URLs, link title).

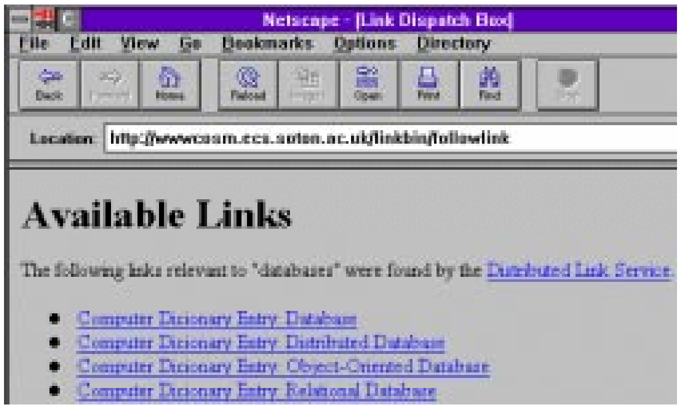


Distributed Link Service

The Distributed Link Service, developed from ideas in the Microcosm pr
of links to be published on the Web. These link databases, or linkbases,
requests for link following either dynamically, based on an individual requ

Figure 1a: A user requests a link from the link service using the client interface

Client Interface: To provide a more direct interface than form filling for many of the link service operations, a DLS client was produced for various platforms. The client presented itself as a set of menus attached to the title bar of any document viewer. It was a simple utility that formulated DLS requests and communicates these to the selected link server via a WWW browser [7]. The client allowed the user to select a predefined context, or topic of interest, from one menu and reacted to DLS requests from a second menu (figure 1a). Details of the selection the user made, the document in which this selection was found, and any selected context were encapsulated as an HTTP request and communicated to the WWW client browser using the platform's appropriate IPC facilities. The link server returned the results of the DLS request to the WWW



Available Links

The following links relevant to "databases" were found by the Distributed Link Service.

- Computer Dictionary Entry Database
- Computer Dictionary Entry Distributed Database
- Computer Dictionary Entry Object-Oriented Database
- Computer Dictionary Entry Relational Database

Figure 1b: The link server responds with a page of available destinations

browser, which then presents the results to the user. For example, the result of a Follow Link request might offer a list of appropriate links (figure 1b), or indicate that no links were found.

The client could extract details from any application (not just a Web browser) and thence create link requests that were passed onto the link server via a Web browser. The DLS could therefore provide links for data maintained by applications that may not have their own hypertext linking mechanisms. As an alternative, it became possible to combine the chosen links back into the original document, if its format was capable of representing hypertext links. Hence an option for producing hypertext material was to develop it using the interactive clients described above, and then to *compile* a chosen set of link databases into a specific set of document resources (in HTML, RTF or PDF format [21]) which were then independent of the link service. By varying the compilation parameters, different webs of links could be produced over similar material for different audiences.

An Interfaceless Proxy Link Service

To provide a convenient interface to the facilities of the DLS, the interactive client required the ability to find information such as the current selection and the identity of the current document from many different applications.

Eliciting such information required a ‘non-standard’ software solution at a time when WWW browsers and inter-application communication were constantly evolving. Further complications came from revisions in operating system implementations, and meant that it was difficult to make the client work consistently as the WWW environment was constantly upgraded.

Independently of software development problems, feedback from users of the DLS (publishers who were partners in the Open Journals project described later) expressed the opinion that the link service should not change the user’s experience of the WWW by making them have to use extra menus or learn any new concepts. It was also seen as desirable that no software should be downloaded to the user’s computer, and no special installation procedure should need to be supported.

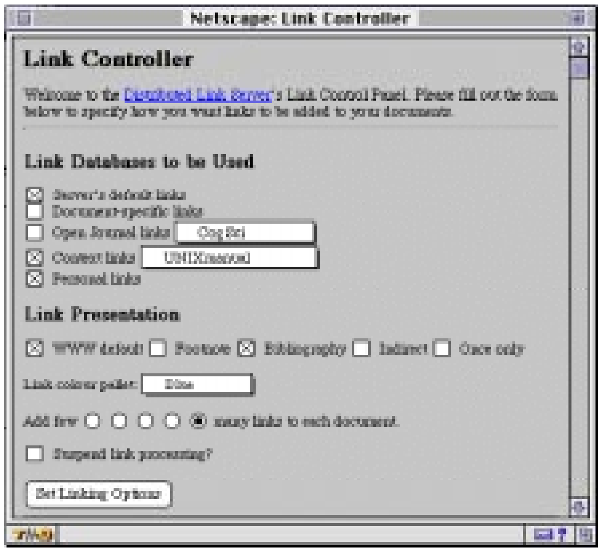
Hence an alternative, ‘interfaceless’ approach was investigated: to make the link service transparent to its users by embedding it in the Web’s document transport system, compiling links into documents *as they were delivered to the browser* by a specially adapted WWW proxy server.

This approach required no extra client software for the user, which is an immediate practical benefit, but it does suffer from a number of disadvantages. Firstly, abandoning the client made it impossible to create a link by the usual method of making a selection and choosing *Start Link* from a menu. It also changed the browsing paradigm from “reader-directed inquiry” to “click on a predefined choice” [13]. Using the client software the reader chose their focus of interest in a document (by making a selection in the browser) and then requested the system to find any relevant links (by making a menu selection). This allowed the reader to stay in charge, whereas without the client the reader is reduced to clicking on a predefined set of buttons.

The second disadvantage is that behind-the-scenes link compilation is applicable only to documents delivered via the WWW and which are coded in well-understood document formats that can themselves support some form of hypertext link. These requirements abandon some of the advantages of the open system previously described, since there are relatively few document formats into which links can be embedded.

Since the interfaceless link server requires at least some initial configuration (for example choosing applicable sets of link databases), a method for communicating with the server has been developed. This takes the form of a kind of a “link remote controller” which is an HTML form displayed as a separate window and whose results are interpreted by a module in the link server (figure 2).

The controller establishes a session (a binding of a user and host together with a set of link server parameters) which is used to control the behavior of the link server from that point in time onwards for that particular user. It is intended


Netscape: Link Controller

Link Controller

Welcome to the Distributed Link Server's Link Control Panel. Please fill out the form below to specify how you want links to be added to your documents.

Link Databases to be Used

[✓ Server's default links]
Document-specific links
Open Journal links: CogSci
[✓ Context links]: UNIXmanual
[✓ Personal links]

Link Presentation

[✓ WWW default] Footnote [✓ Bibliography] Indirect Once only

Link colour pallet: Blue

Add few ○ ○ ○ ○ [✓ ●] many links to each document.

Suspend link processing?

Set Linking Options

Figure 2: Link Service “Remote Controller”

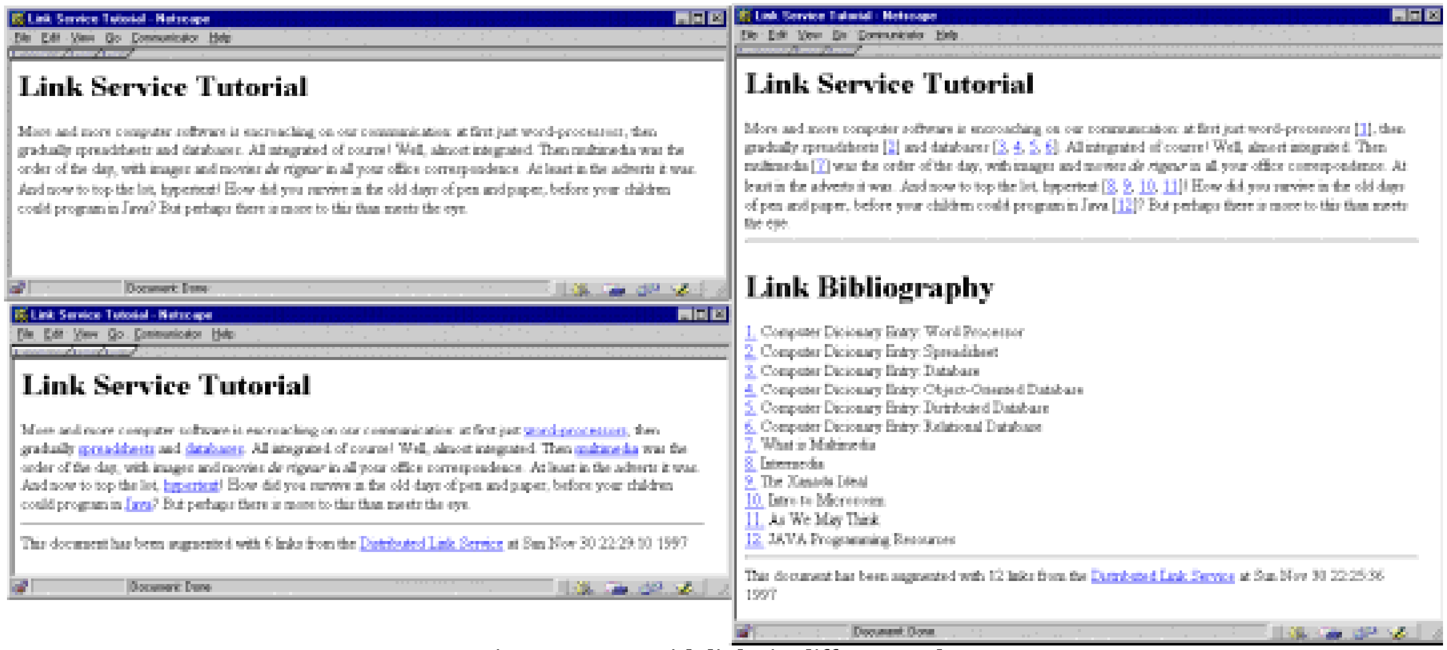
that the user will invoke the controller just once to set a preferred configuration, and only again afterwards to adjust the configuration—links will always be added automatically to the documents according to the last settings of the controller.

Controlling Links

The purpose of the controller is to give to the user the ability to choose how links are displayed and used within the processed documents. Previous versions of the software elevated links to first-class objects by giving them a status independent of their containing documents, but the DLS now allows the user to directly manipulate links to control presentation and navigation features. The following section briefly describes some of these facilities; for greater detail the reader is referred to [2].

Link Inclusion: The control panel in figure 2 gives the user the ability to choose which of the server’s installed linkbases are to be combined with requested documents, or to completely bypass the link compilation if a ‘normal’ document viewing mode is required. The Open Journal Framework [4] makes use of this kind of control panel to help the user navigate through large suites of collected but separate Internet resources, all integrated by the use of linkbases. By introducing a model of Internet resources (collections of documents and associated link databases) and aggregations of these resources (collections of collections of documents and associated link databases), it is possible to define the user’s location in a document space, and hence to know what hypertext actions are applicable at each point in that document space. Without this model the same sets of link databases are applied to any document which the user sees.

Link Presentation: A link is selected for inclusion in a document by virtue of its presence in a chosen linkbase and its applicability to the current document. Having determined a match, the link server presents the link within the document according to a specific presentation format


Link Service Tutorial - Netscape
File Edit View Go Communicator Help
Document: Done

Link Service Tutorial

More and more computer software is encroaching on our communication: at first just word-processors, then gradually spreadsheets and databases. All integrated of course! Well, almost integrated. Then multimedia was the order of the day, with images and movies *de rigeur* in all your office correspondence. At least in the adverts it was. And now to top the lot, hypertext! How did you survive in the old days of pen and paper, before your children could program in Java? But perhaps there is more to this than meets the eye.

Link Service Tutorial

More and more computer software is encroaching on our communication: at first just word-processors, then gradually spreadsheets and databases. All integrated of course! Well, almost integrated. Then multimedia was the order of the day, with images and movies *de rigeur* in all your office correspondence. At least in the adverts it was. And now to top the lot, hypertext! How did you survive in the old days of pen and paper, before your children could program in Java? But perhaps there is more to this than meets the eye.

This document has been augmented with 6 links from the Distributed Link Service at Sun Nov 30 22:29:10 1997

Link Service Tutorial

More and more computer software is encroaching on our communication: at first just word-processors [1], then gradually spreadsheets [2] and databases [3, 4, 5, 6]. All integrated of course! Well, almost integrated. Then multimedia [7] was the order of the day, with images and movies *de rigeur* in all your office correspondence. At least in the adverts it was. And now to top the lot, hypertext [8, 9, 10, 11]! How did you survive in the old days of pen and paper, before your children could program in Java [12]? But perhaps there is more to this than meets the eye.

Link Bibliography

1. Computer Dictionary Entry: Word Processor
2. Computer Dictionary Entry: Spreadsheet
3. Computer Dictionary Entry: Database
4. Computer Dictionary Entry: Object-Oriented Database
5. Computer Dictionary Entry: Distributed Database
6. Computer Dictionary Entry: Relational Database
7. What is Multimedia
8. Intermedia
9. The Xanadu Ideal
10. Intro to Microcosm
11. As We May Think
12. JAVA Programming Resources

This document has been augmented with 12 links from the Distributed Link Service at Sun Nov 30 22:25:36 1997

Figure 3: Page with links in different styles

set by the control panel (see figure 3 for examples).

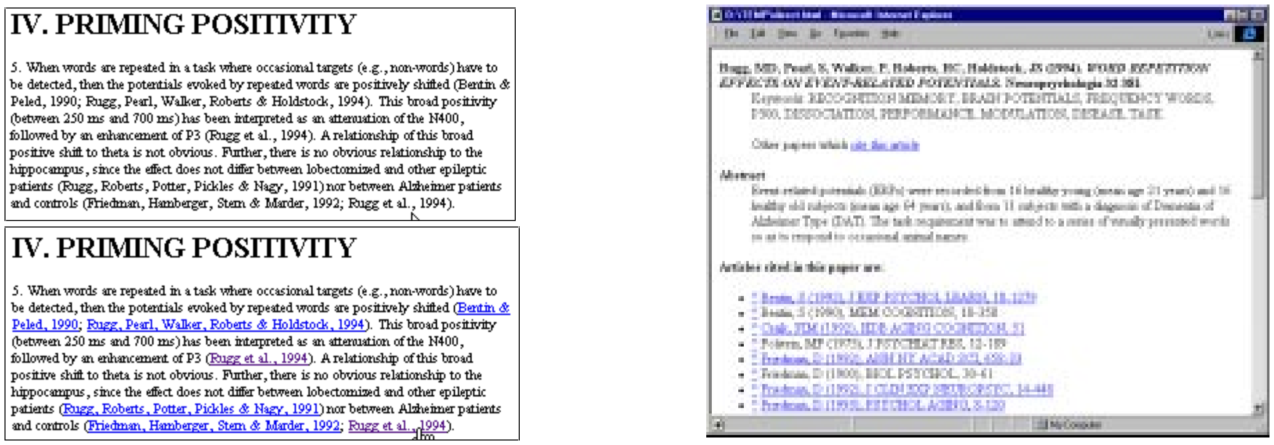
default	the matched content is chosen as a direct site for linking. The chosen text is displayed as all normal WWW links.
footnote	add a footnote marker after the identified link site. The marker and not the text carries the WWW link.
citations	add an annotation which looks like a bibliographic citation after the identified link site. The annotation carries the web link, not to the ultimate destination, rather to the matching entry in a link bibliography appended to the foot of the document. The entry in this pseudo-bibliography contains a link to the ultimate destination.
indirect	the link is presented in any of the above styles, but does not go directly to the ultimate destination. Instead, it invokes the link service to produce an intermediate page describing the ultimate destination.

The presentation is specified by the end user, not the author or the publisher, and given according to the above fixed set of styles which are mapped onto the underlying presentation system as best they can. For example, plain links are blue underlined text in HTML documents or black outlined buttons in PDF. An important role of presentation is not *decoration* but *discrimination* as users must be easily able to distinguish between links which are added by the server and those which are native to the document (*i.e.* links calculated by a computer vs. links inserted by the original author).

Link Prioritization: Further aspects of presentation are governed by a link's priority, so that a link may be rendered with a discrete or striking visual appearance, or even abandoned altogether as not sufficiently urgent to warrant the user's attention. To allow the user to discriminate between different links the server must maintain some concept of the pertinence of different kinds of links. This measurement can also be used to cull links from a potentially over-annotated document (the curse of the indiscriminate generic link). Since there may be many dozens of link databases active, an important task of the server is to throttle over-zealous link producers.

The consideration of priority has both author and user facets. An author may specify a link's significance (how useful a role it is expected to play *a priori*) in the link markup, but only the end user (or his or her agent) can judge its importance relative to the current task in hand, taking into account many factors such as the users experience in the particular knowledge domain. The link controller enables this distinction by allowing the user to set the relative importance of the various linkbases that are used (so scaling the intrinsic link priorities) and also by choosing among various presentation modification schemes which alter the font style or color of the link anchors. These enable (for example) brighter links to correspond to more direct and specifically authored links created by a human author, and duller colored links to correspond to general links created by a simple dictionary lookup or a statistical lexical operation.

Link Semantics: The default link behaviour of a WWW browser is to change the document view when a user activates the link, but more complex behaviours are often useful. Presenting a summary of the ultimate destination as an intermediate stage may be useful if the sizes of the document, the anticipated access time or the cost are large. Checking the chosen destination against a history list or a recommended navigation guide may be helpful for naive users. The link server can provide for all these activities by specifying a script to be invoked when the link is activated



IV. PRIMING POSITIVITY

5. When words are repeated in a task where occasional targets (e.g., non-words) have to be detected, then the potentials evoked by repeated words are positively shifted (Bentin & Peled, 1990; Rugg, Pearl, Walker, Roberts & Holdstock, 1994). This broad positivity (between 250 ms and 700 ms) has been interpreted as an attenuation of the N400, followed by an enhancement of P3 (Rugg et al., 1994). A relationship of this broad positive shift to theta is not obvious. Further, there is no obvious relationship to the hippocampus, since the effect does not differ between lobectomized and other epileptic patients (Rugg, Roberts, Potter, Pickles & Nagy, 1991) nor between Alzheimer patients and controls (Friedman, Hamberger, Stern & Marder, 1992; Rugg et al., 1994).

IV. PRIMING POSITIVITY

5. When words are repeated in a task where occasional targets (e.g., non-words) have to be detected, then the potentials evoked by repeated words are positively shifted (Bentin & Peled, 1990; Rugg, Pearl, Walker, Roberts & Holdstock, 1994). This broad positivity (between 250 ms and 700 ms) has been interpreted as an attenuation of the N400, followed by an enhancement of P3 (Rugg et al., 1994). A relationship of this broad positive shift to theta is not obvious. Further, there is no obvious relationship to the hippocampus, since the effect does not differ between lobectomized and other epileptic patients (Rugg, Roberts, Potter, Pickles & Nagy, 1991) nor between Alzheimer patients and controls (Friedman, Hamberger, Stern & Marder, 1992; Rugg et al., 1994).

Rugg, MD, Pearl, S, Walker, P, Roberts, RC, Holdstock, JS (1994). *WORD REPETITION EFFECTS ON EVENT-RELATED POTENTIALS.* **Neuropsychologia 32 381**

Keywords: RECOGNITION MEMORY, BRAIN POTENTIALS, FREQUENCY WORDS, P300, DISSOCIATION, PERFORMANCE, MODULATION, DISEASE, TASK

Other papers which cite this article

Abstract

Event-related potentials (ERPs) were recorded from 16 healthy young (mean age 21 years) and 16 healthy old subjects (mean age 64 years), and from 11 subjects with a diagnosis of Dementia of Alzheimer Type (DAT). The task requirement was to attend to a series of visually presented words so as to respond to occasional animal names.

Articles cited in this paper are:

- Bentin, S (1990), J EXP PSYCHOL LEARN, 16, 1270
- Bentin, S (1990), MEM COGNITION, 18-358
- Craik, FIM (1992), HDB AGING COGNITION, 51
- Folstein, MF (1975), J PSYCHIAT RES, 12-189
- Friedman, D (1992), ANN NY ACAD SCI, 658-[illegible]
- Friedman, D (1990), BIOL PSYCHOL, 30-61
- Friedman, D (1992), J CLIN EXP NEUROPSYC, 14-448
- Friedman, D (1993), PSYCHOL AGING, 8-120

Figure 4: Use of Citation Links in Cognitive Science Open Journal

by the user (overriding the default behaviour in the browser). However, the range of behaviours is dependent on the scripting language that the browser supports.

PUTTING THE LINK SERVICE TO WORK: DIFFERENT KINDS OF LINKS

Links are the connections between different parts of hypertext and are often visualized as buttons (a graphic or highlighted word on which the user can click) which cause a new piece of the hypertext to be displayed. Links come in many forms: they can embody the concrete connection between two adjacent pages, sections or footnotes in a document or they can represent the organizational connection between a paragraph in a document and an annotation by a reviewer. They can even express the abstract connection between a concept mentioned in one document and an explanation of that concept in a glossary, dictionary or encyclopaedia. Links are informational entities in their own right, and can be stored, processed and used independently of the documents to which they refer. Links may be between the lexical components of a text, the objects or elements of the format in which it is expressed (XML or DOC) or between the concepts and ideas embodied in the writing.

Keyword Links

A significant experiment in the use of the link server has been the Open Journals project [4, 14], an electronic library experiment which uses various kinds of external links to integrate independently published academic journal resources.

An ‘Open Journal’ is a collection of documents (journal articles, online topic-specific databases, useful educational resources and bibliographic services) together with collections of associated links that together bind the separate resources into an integrated super-publication. Several subject-specific open journals have been produced, including one in Biology that is based around the familiar keyword-style links.

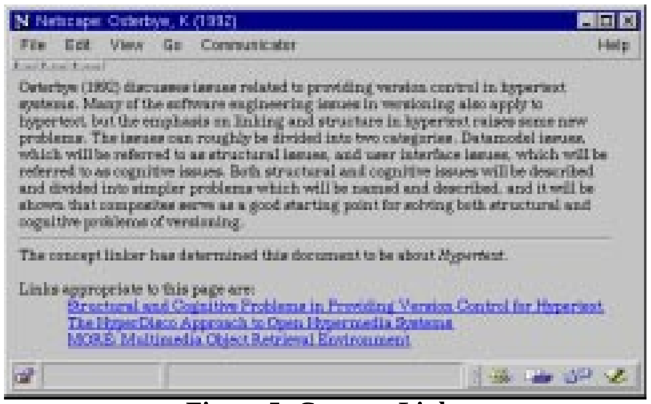
These links were mainly produced by “parsing” the first pages of each PDF journal article to “read” the keywords that describe each article. These author-supplied keywords were used to create generic links that would link the reader to that article if the keyword was mentioned in any other document. A linkbase of keyword links was produced for each journal resource in the “Open Journal”, together with other supported glossary and teaching resources.

The advantage of using this kind of linking is the ease of putting together a modular collection: a set of generic links made for a particular resource easily bind themselves to other resources that are subsequently added to the collection. The disadvantage is that a generic link on a key term may prove to be too broad, or link inappropriately to a homonym. This is partly being tackled by the use of “stop word links” which inhibit links from forming on words or phrases considered too common within a particular collection of documents.

Person Links

Also making use of the link service is the Administrative Information Management System, a project involved in the tracking, versioning and publishing of administrative documents [15]. The role of the link server is to link important personnel and resource information from a management database into an evolving collection of official documents. The link server uses keyword links to connect role names with the personnel information of the individual fulfilling that role, but beyond that recognizes people's names within official documentation such as committee minutes.

In order to link from the use of a person’s name to information about that person, the link server must be able to recognize all of the various ways of referring to a person: in other words that Leslie Carr, Dr LA Carr, LAC and L. Carr are all equivalent references that could be connected to the author’s home page. This is a stage beyond accomplishing generic linking by recognizing the existence of keywords: it is linking by pattern matching on the document's content. Although this effect could have been accomplished by creating a dozen keyword links for each person to match the dozen different ways of writing their name, it became unwieldy in a department of a thousand staff and students. It was more straightforward to provide a



Osterbye (1992) discusses issues related to providing version control in hypertext systems. Many of the software engineering issues in versioning also apply to hypertext, but the emphasis on linking and structure in hypertext raises some new problems. The issues can roughly be divided into two categories. Datamodel issues, which will be referred to as structural issues, and user interface issues, which will be referred to as cognitive issues. Both structural and cognitive issues will be described and divided into simpler problems which will be named and described, and it will be shown that composites serve as a good starting point for solving both structural and cognitive problems of versioning.

The concept linker has determined this document to be about *Hypertext*.

Links appropriate to this page are:

Structural and Cognitive Problems in Providing Version Control for Hypertext
The HyperDisco Approach to Open Hypermedia Systems
MORE: Multimedia Object Retrieval Environment

Figure 5: Concept Links

pattern matcher based on an existing database of personnel that directly recognized any mention of one of those people, and provided the appropriate link.

Citation Links

The Open Journal project further produced an Open Journal in Cognitive Science based on the concept of citation linking. Citation links are the implicit connections between a citation in an academic article, the corresponding bibliography item and the cited article's metadata stored in a bibliographic database. Citation links are recognized by two complex pattern-matching operations (one for each citation, another for each bibliography entry) together with an external database query (to check for the presence of the cited article within the external bibliographic resource). Figure 4 shows a fragment of an article (top left) and the same fragment with the citation links added (bottom left). The linked page from the citation database is shown on the right.

Publishers are now beginning to see the advantage of adding links to bibliographic databases, but this is usually achieved by hard-wiring a point-to-point link in a specially constructed meta-data document that abstracts the content of the real article. Although the link server approach may seem over-complicated, an advantage is that links can be applied directly to citations in any documents, from any publisher, not just those over which the user has editorial control.

Concept Links

Beyond simple pattern matching, information retrieval techniques can be used on a document to derive links based on similarity of vocabulary between the document in hand and other well-known collections of documents. Instead of focussing on a micro-region of the document (a single word, phrase or bibliographic element), this technique creates a link from the contents of the document as a whole, proposing links on the basis of the concepts mentioned in the document. These concept links, shown in figure 5, are based not on simple pattern matching, but on more complex information retrieval processing (here using a Naïve Bayes classifier from the freely available Rainbow package [18]).

Evaluation

As part of the Open Journals project, some informal evaluation of the impact of the linking and design of the open journals was undertaken. The journals were announced by email to pre-selected user groups of specialists in each journal's field. The users were given several weeks in their normal working environment to browse through the contents of the journal and to make informal notes on the quality of the links and various factors of the user interface. The users of one journal received a visit to provide assistance in the use of the journal. Feedback was subsequently collected from a focus group. Users of another journal received less personal instruction, and their feedback was solicited via a form on the Web.

A lot of the feedback from focus groups was directed at the WWW browser environment and the user interface rather than at the linking facilities themselves. Many were confused by the use of multiple windows, including the use of separate helper applications for displaying PDF documents. It also became apparent that many working environments did not allow the users to specify their WWW proxies, so that they could not even connect to the link service! In response to this situation, the link service was altered so that it could be invoked as an "explicit proxy" prepended to the URL of the destination Web page. As the target page is delivered to the user, not only are new links added, but also existing links are rewritten so that subsequently those pages will also be explicitly delivered by the link service.

A majority of users expressed a strong preference for an indirect style of link presentation because it provides them with a list of destinations that they can evaluate before they commit themselves to following a new link. (It also prevented multiple link anchors from cluttering each page.) As users became more experienced with the Web during the project, increasingly sophisticated link actions were requested: for example links that produce drop-down menus listing the available destinations as a lightweight alternative to the separate Available Links page.

The most frequent requests are for links to a greater range of resources than the dozen or so publications that have formed the core of the project. In particular links to research databases are seen as important to some scientists, whereas links to a more complete set of bibliographic data is the key issue for others.

Tempering this desire for more links is the requirement to be able to more accurately control the kind of links which are displayed to the user—some keywords were felt to be too broad or elementary to be useful. These requests led to the implementation of link prioritisation and also sets of stop words.

A NEW INTERFACE

The atomic links of the underlying presentation engine (whether WWW browser or PDF viewer) are used by the link server to implement the navigation transfers implicit in the semantics of the more complex link types, if indeed ‘transfers’ are the best way of presenting the link connection.

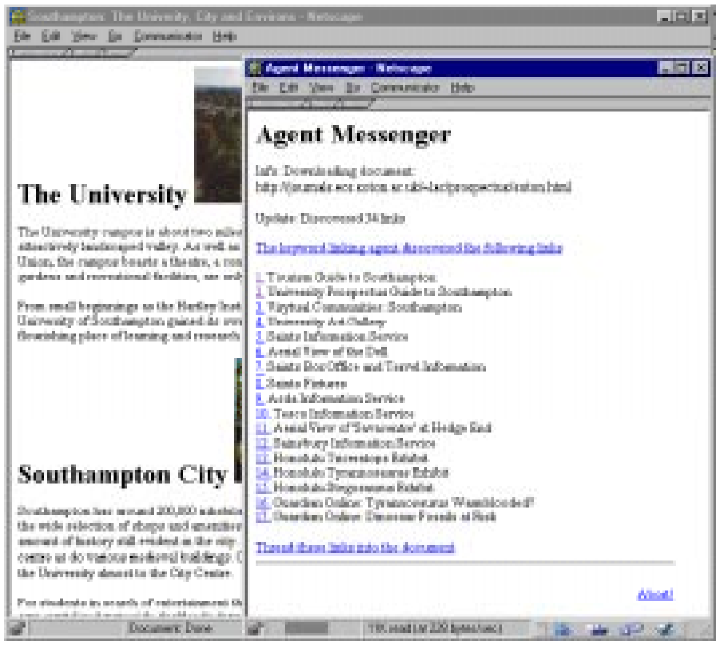
To implement link types more complex than the simple keyword links the link server uses a separate link sub-processor. Based on the URL of the document it is dealing with, the link server can choose to start up an arbitrary process to work on the WWW data before it inserts its simple, keyword links. For any particular application the server configuration files specify the roots of the various URL trees which contain journal articles to have citation links added, or committee minutes to have people links added or technical articles to have concept links added. When a document from one of those trees is seen, the link server starts up the appropriate link sub-processor to delegate the computation required to actualize those links before it sees the document data to add keyword links.

This architecture is currently only a prototype that involves no communication between the link processor proper and the various sub-processors. The lack of communication makes it difficult to build consistent link presentation and behaviour across all the processes that are adding links. A better way of achieving this effect would be to have the sub-processors outputting not linked documents for further processing, but simple point-to-point links that the link processor itself goes on to add into the document with its own consistent style and behaviour rules.

These complex link types are similar to Ashman's functional links [1], where evaluating the description of both the source and destination anchors of a link may require complex processing. However the link server system described in this paper does not yet have a canonical description for the operation of each of the sub-processors that could be coded in a link and processed as part of a link database.

The problem with these complex link types is the amount of processing that is needed to “actualize” them. Efficient data structures and searching techniques mean that adding keyword links to a document does not lead to serious delays, especially in a system that already has intrinsic delays due to delivering data over a global network. However, implementing link services as a WWW proxy does mean that *any* extra processing can begin to contribute to a bottleneck.

The amount of delay involved in producing citation links was tolerable in our experience with the Cognitive Science articles in the Open Journals project. However the very long lists of references commonly found in disciplines such as Biology or Medicine could have unacceptably slowed down the delivery of the document through the proxy to the user. (This would certainly be the case with a comprehensive citation database, although measurements of a publisher-specific citation resolver show that normal PDF articles from a biology journal can be linked in under a second.) Much worse is the amount of time taken for quite short documents to be analyzed for concept links, making the use of that sub-processor out of the question for real applications. But even more heavyweight processing could be anticipated in the use of ontology databases for recognizing the concepts represented by the use of structured vocabularies [10].


Southampton: The University, City and Environs - Netscape
File Edit View Go Communicator Help
The University
Southampton City
Agent Messenger - Netscape
File Edit View Go Communicator Help
Document: Done

Agent Messenger

Info: Downloading document:
http://journals.ecs.soton.ac.uk/~lac/prospectus/intro.html

Update: Discovered 34 links

The keyword linking agent discovered the following links

1. Tourism Guide to Southampton
2. University Prospectus Guide to Southampton
3. Virtual Communities: Southampton
4. University Art Gallery
5. Saints Information Service
6. Aerial View of the Dell
7. Saints Box Office and Travel Information
8. Saints Fixtures
9. Arts Information Service
10. Tesco Information Service
11. Aerial View of 'Sainsburys' at Hedge End
12. Sainsbury Information Service
13. Honolulu Triceratops Exhibit
14. Honolulu Tyrannosaurus Exhibit
15. Honolulu Stegosaurus Exhibit
16. Guardian Online: Tyrannosaurus Warmblooded?
17. Guardian Online: Dinosaur Fossils at Risk

Thread these links into the document

About!

[illegible]

Figure 6: Agent Links window

For this reason, the underlying model of the link service had to be changed. The decision to add links as part of the document delivery process was taken to help a naïve user base to come to terms with an unfamiliar hypertext environment. As a side effect the generation of the links became synchronized with the delivery of the document. By contrast the presentation of highly pertinent related information needs to become an adjunct to the document browsing process. Certainly the requirement to keep the browsing activity unchanged can be relaxed for more experienced users of the now more Web-oriented world.

Asynchronous Link Processing

An essential requirement for the link service is that the links must be processed and delivered asynchronously, *i.e.* they must not hamper the delivery of the document. They must be loosely coupled with the reader’s interaction with the document because there is only any point in delivering any links relevant to a document if the user is still reading that document. If the user browses to another document before the links have been processed they must be abandoned. Consequently, the links have to be displayed separately, in a manner independent of the document that the user is viewing.

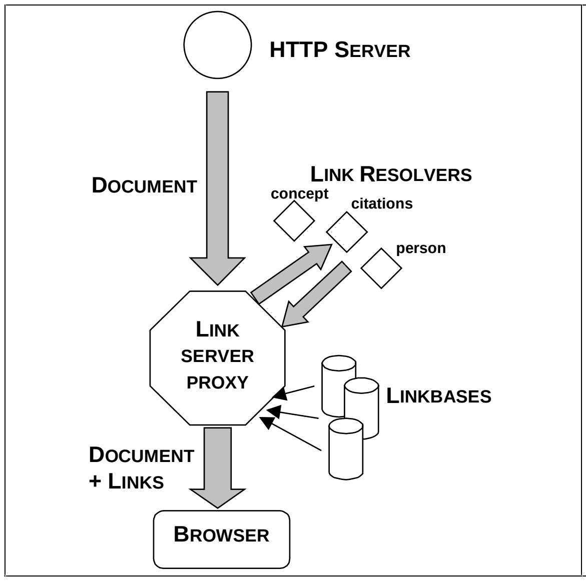


Figure 7a: DLS architecture

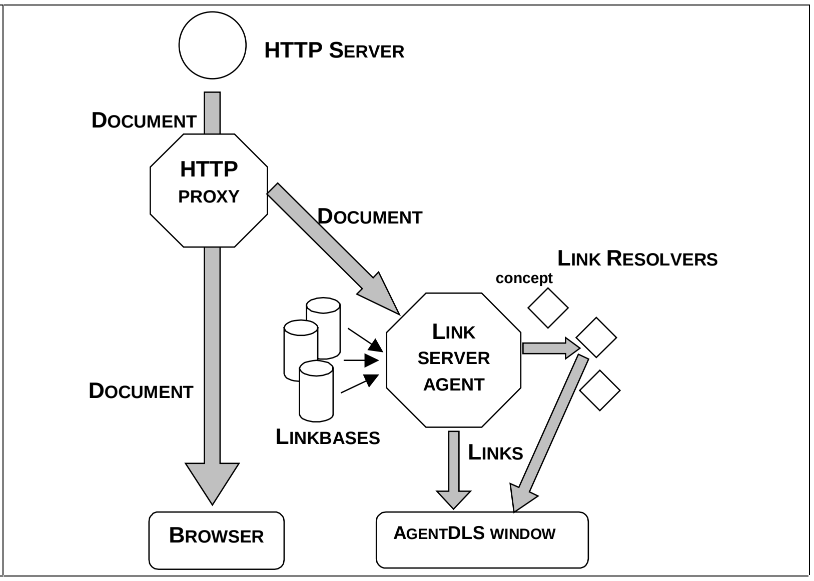


Figure 7b: Agent DLS architecture

Advisory Link Presentation

When the link server incorporated links directly into the contents of each document, the links themselves took on an importance that was immediate as that of the document itself. This was one reason why it was necessary to use different colors and link styles to allow the user to distinguish links originally placed in the document by the author from links added by the link service. If the links are now being presented independently of the document and at some time after the document is delivered, link processing now takes the status of an advisory service which can add value to the user's browsing process, but which is not confused with the original document contents. This is an advantage from the point of view of publishers' copyright and licensing (no derivative works are created) but also with respect to the author's *moral rights* to have a document displayed "without any amendment that impugns his or her integrity or reputation" [20]. The problem with link compilation is that the author's moral right encompasses even the case of having their material linked to a document of which they disprove.

Agent Interface

The combination of independent processing, independent display and advisory role moves the link service into the realm of agents. A user interface agent is "a *personal assistant* who is collaborating with the user in the same work environment" to assist in whatever tasks are being performed or to provide alternative material into related areas of work [17]. The familiar concept of an agent as a piece of software that monitors a user's activities and offers help and advice certainly works well for the new link server, which monitors a user's browsing activity and dispatches linking sub-processes accordingly. It constantly displays the current state of link processing and finally shows any sets of links that each process has discovered. Figure 6 shows the AgentDLS window that has been enlarged to show the complete list of links that have been discovered. Usually the window would be kept to a smaller size in keeping with its 'advisory' role and displaying only a summary of the kinds of links available. The user has options to display the complete list of links (as seen in figure 6) or to thread the links back into the target document (reverting to the link server's original behaviour).

The properties of an agent are often quoted as autonomy, social ability, responsiveness and pro-activeness [25]. Of these four properties, AgentDLS really exhibits only the third: it perceives the user's information environment and responds in a timely fashion to changes that occur in it. Many user interface agents also share the same single 'hard agent' property [16]. In particular the Remembrance Agent [23] responds to the user's browsing by informing the user of similar documents in the local file store (citing 'automatic hypertexting' and 'reference advise for technical papers' as future applications).

The transformation from link service into an agent is subtle but distinct. The Microcosm system that originally inspired the work also processed its links asynchronously, sometimes visibly adding buttons to the document viewers some seconds after the original document had been displayed. Microcosm also displayed an available links box that was independent of the current document viewer. However, Microcosm was normally used as a foreground system that provided a user interface through which one interacted with the available documents, and which required user intervention to search for links. The link service, by contrast, became a background proxy that provided links for the user automatically and continuously with no noticeable user interface. The link agent maintains the background processing but decouples it from the document processing and restores a discrete user interface.

The change in the system architecture is illustrated in figure 7. Although slightly simplified (linkbases can reside on remote HTTP servers, and it is intended that link resolution should also be performed remotely) the main change is that link delivery is now separated from the document delivery. This involves hijacking an HTTP proxy server to split the document delivery into two: one channel feeding directly to the browser and one to the linker. As currently implemented the AgentDLS window is simply an extra Web browser window that is updated from the link server by using HTTP server push: a mechanism where the server repeatedly sends updated versions of the current document. This allows the browser to display an up-to-date picture of the link processing. In fact complex interaction with the server is not easy to maintain through such a mechanism and it is likely that the agent interface will be re-implemented either as a JAVA applet or as similarly extensible object such as an ActiveX control.

However the agent effect is obtained, the question remains "Why bother to make agents perform these calculations on the fly instead of creating a simple, static, point-to-point linkbase that will apply to the documents of interest?" Certainly for any closed domain of documents the results of the agents' processing can be compiled into a linkbase and applied by the simple proxy link server, but the philosophy of the Open Journal project is to work with a potentially unconstrained set of resources. Contrariwise, a linkbase can be considered as the cached output of an agent and so a compromise method could be used, whereby the agent processes are only invoked the first time that a new document is seen.

RELATED AND FUTURE WORK

Based on the extended Dexter model [12] describe a very similar link service mechanism to the DLS in which the links are maintained by a separate server, but combined with the text document by a Java applet embedded in the users browser. This corresponds to the so-called *heavyweight client* model in which part of the server processing is devolved to the client. Another link service (also known as the Distributed Link Service) has been produced for the Aquarelle project [22]. The advantage of this service is that it is built on top of a robust database managing of millions of links in a way that avoids the dangling links problem.

XLink (the eXtended Linking Language, formerly XLL) is a proposed WWW standard for expressing complex links types for the WWW [19]. Built on top of XML, a simplified version of SGML, and in conjunction with XSL (the proposed stylesheet language) it offers the ability to control link behaviour and presentation much more precisely than with HTML. The DLS is being modified to support some of these new linking facilities [3].

Salampris [24] describes a digital libraries system that is based around a collection of co-operating open hypermedia agents. This is a more mature model than the DLS currently uses, allowing proper communication between the link resolving components. Currently the link server does not attempt to manage its sub-processes in a sophisticated manner, but various strategies are being investigated from the field of distributed information management [9, 11]. So far the linking processes have only been tested with small user groups, and low numbers of simultaneously active processes. It is anticipated that adopting these strategies will help to provide complex links to a large, simultaneously active user base by carefully devolving processing to specialised replicated server sites using appropriate caching and pre-caching techniques.

CONCLUSIONS

A hypermedia link service can provide the WWW with a powerful tool with which to address the restrictions of embedded links that are fixed with the publication of a document. We have shown that a simple link service can be implemented using standard Web browsers and servers and described a revised implementation that works without the need for any additional client software. We have also shown how a link service can provide novel user-centered facilities for link presentation, discrimination and navigation and have demonstrated a use of the link server to provide *post-hoc* integration for a set of academic journals.

A significant practical shortcoming of the link server is the requirement to retrieve and apply links to the document while it is being delivered to the user. Any delay in the processing of the links therefore means that the user has to wait for the document. Although this has not proved to be a significant problem for databases of explicit links, the more complicated processing required for citation linking requires certain tradeoffs to achieve the response times that WWW users expect. Furthermore, any kind of on-the-fly linking in the semantic domain (links between concepts) rather than in the purely lexical domain (links on text features) slows the delivery of the document beyond the limits of acceptability. Therefore a new agent-style interface has been designed for the service, so that collections of links can be offered to the user as they are discovered by various independent link processors all acting asynchronously. The aim is to allow the WWW to deal with inherently abstract connections between texts, rather than its familiar simple, point-to-point 'jumps'.

ACKNOWLEDGEMENTS

The work on PDF linking was carried out by the Electronic Publishing Research group at Nottingham University as part of the Open Journals project, JISC grant ELP2/35 from the Electronic Libraries program. The AIMS project is funded in the UK by JISC under the JTAP programme. The citation database shown in this paper was provided by the Institute for Scientific Information (ISI). Thanks are due to the conference reviewers, whose comments have helped to greatly improve this paper.

BIBLIOGRAPHY

1. Ashman H.L. & J.L.M. Verbyla (1993), Externalizing Hypermedia Structures with the Functional Model of the Link, Proceedings of Online Information '93, London, Learned Information, 1993, 291-301.

2. Carr, L., De Roure, D., Davis, H., and Hall, W., (1998) "Implementing an Open Link Service for the World Wide Web". World Wide Web. 1(2). Baltzer. Amsterdam: Netherlands (in press)

3. Carr, L.A., H.C. Davis, D. De Roure and W. Hall (1998), "Application-Independent Link Processing", Proceedings of Seventh International World-Wide Web Conference, May 1998.

4. Carr, L., D. De Roure, W. Hall and G. Hill (1996), "Open Information Services", Computer Networks and ISDN Systems, 28, 7/11, 1027-1036, Elsevier

5. Carr, L., D. De Roure, W. Hall and G. Hill (1995), "The Distributed Link Service: A Tool for Publishers, Authors and Readers", The Web Journal 1, 1, 647-656, O'Reilly and Associates.

6. Davis, H., W. Hall, I. Heath, G. Hill and R. Wilkins (1992), "Towards an Integrated Information Environment with Open Hypermedia Systems," In ECHT '92, Proceedings of the Fourth ACM Conference on Hypertext, Milan, Italy, November 30-December 4, 1992, ACM Press, pp. 181-190.

7. Davis, H., S. Knight, W. Hall (1994), "Light Hypermedia Link Services: A Study of Third Party Application Integration," In Proceedings of the Sixth ACM Conference on Hypertext, Edinburgh, Scotland, September 1994, ACM Press, pp. 41-50.

8. Davis, H., A. Lewis, A. Rizk (1996), OHP: A Draft Proposal for an Open Hypermedia Protocol, at ACM Hypertext '96, Open Hypermedia Systems Workshop, http://diana.ecs.soton.ac.uk/~hcd/protweb.htm

9. De Roure, D., W. Hall, S. Reich, A. Pikrakis, G.J. Hill, M Stairmand, (1998), "An Open Framework for Collaborative Distributed Information Management", Technical Report, Dept of Electronics and Computer Science, University of Southampton, UK.

10. Goble, C., J. O'Neill and J. Bullock (1997) "Authoring using a Terminology Service", Technical Report, Department of Computer Science University of Manchester, Manchester, UK.

11. Goose, S., Dale, J., Hill, G. J., De Roure, D. C. and Hall, W., (1996) "An Open Framework for Integrating Widely Distributed Hypermedia Resources" In: Proceedings of the Third IEEE International Conference on Multimedia Computing and Systems, 364-371, Hiroshima, Japan.

12. Grönbæk, K., N.O. Bouvin and L. Sloth (1997) "Designing Dexter-based hypermedia services for the World Wide Web," In *Proceedings of the Eighth ACM Conference on Hypertext*, Southampton, UK,146-156.

13. Hall, W. (1994), "Ending the Tyranny of the Button", IEEE Multimedia 1,1 pp. 60-68.

14. Hitchcock, S., L. Carr, S. Harris, J. Hey and W. Hall (1997) "Citation linking: improving access to online journals," In *Proceedings of Second ACM conference on Digital Libraries*, Philadelphia, pp.115-122.

15. Hughes, G. (1998) AIMS First Year Status Report. http://aims.ecs.soton.ac.uk/report/report_3_98.pdf

16. Lieberman, H. (1997) Autonomous Interface Agents, In *Proceedings of CHI 97, ACM Conference on Computers and Human Interface*, Atlanta, March 1997.

17. Maes, P., (1994) "Agents that Reduce Work and Information Overload" In: Communications of the ACM, 37(7), 31-40.

18. McCallum, A. and Rennie, J. (1997) "Naive Bayes algorithm for learning to classify text". http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes.html

19. Maler, E. and De Rose, S., (eds) (1997) XML Linking Language (XLink), W3C Working Draft March-3-98, http://www.w3.org/TR/WD-xml-link

20. Oppenheim, C. (1996) "Moral Rights and the Electronic Library" In Ariadne: Elib Magazine Online, Issue 4, http://www.ariadne.ac.uk/issue4/copyright/intro.html

21. Probets, S., Brailsford, D., Carr, L. and Hall, W. (1998). "Dynamic Link Inclusion in Online PDF Journals" In *Proceedings of the Seventh International Conference on Electronic Publishing, Document Manipulation and Typography.* St Malo, France.

22. Rizk, A. and Sutcliffe, D. (1997) "Distributed Link Service in the Aquarelle project" In *Proceedings of the 3rd Workshop on Open Hypermedia Systems*, Southampton, UK. 106-110. CIT Scientific Report number SR-9701. ISSN 1397-5390.

23. Rhodes, B. J. and T. Starner (1996) "Remembrance Agent: A continuously running automated information retrieval system" In *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi Agent Technology* (PAAM '96), 487—495.

24. Salampris, M. (1997) "Agent-based Open Hypermedia Model for Digital Libraries" In *Proceedings of the 3rd Workshop on Open Hypermedia Systems*, Southampton, UK. 116-125. CIT Scientific Report number SR-9701. ISSN 1397-5390.

25. Wooldridge M. J. and N.R. Jennings (1995) "Intelligent Agents: Theory and Practice" The Knowledge Engineering Review 10(2) 115-152.