Chapter 7

Multi-disciplinary approaches and web authoring

Chapter objectives:

This chapter demonstrates how research ideas in hypertext generated from chapters 3 – 6, are incorporated and integrated into HyperAT, a prototype research tool for the web. The web is chosen because it is the largest and most widely used hypertext system. The main points covered in this chapter are to:

- understand the web
- assess the extent of the LIH problem on the web
- survey solutions to address the LIH problem on the web
- design and develop HyperAT to demonstrate how established hypertext and HCI elements can be integrated and implemented into a practical authoring tool

7.1 The web and the "lost in hyperspace" problem

In chapter 2, the LIH problem was examined from both designers' as well as end-users' perspectives, and this thesis argued that the LIH is not just a psychological problem, it is an engineering one. Because LIH is a complex problem, multi-disciplinary approaches as discussed in chapters 3 – 6, are required to understand usability issues. To test these ideas, the World Wide Web was selected as a hypertext example, since it is the most widely used and largest hypertext system ever. In the keynote address by John Smith at Hypertext'97 Conference, he urged the Hypertext Community to embrace the World Wide Web and not just tolerate it, if she wants to continue and create value for its knowledge (Smith, 1997). Before describing how the ideas generated from the multi-disciplinary approaches are adapted and applied to the World Wide Web, this chapter briefly reviews the LIH problem on the World Wide Web, surveys solutions to the problem, and then describes an engineering approach taken in this thesis to address it.

The web project initiated in 1990 was originally created as an on-line information tool for high-energy physics research at CERN (the European Center for Nuclear Physics Research in Geneva, Switzerland). Tim Berners-Lee and colleagues, the originators of the web, built it based on the hypertext paradigm. Information is organised as a series of documents referring to each other with links for search and retrieval of text, images, sound and video. Links in a document may go to other server machines containing the actual information. Based on its likeness to a spider's web, this world of hypertext links is also called the web.

Underlying the working of the web are the basics of networking and several transfer protocols for sending and receiving information. Since this thesis is concerned with design issues regarding the LIH problem on the web, it will not delve into the technical, operational aspects of the web. Very briefly, the web uses a client-server architecture for distributed hypertext that can be accessed over the Internet. The servers around the world provide data to the client software in a standardised format called HTML (HyperText Markup Language) through a standard communication protocol called HTTP (HyperText Transfer Protocol). Documents can be accessed using a Uniform Resource Locator (URL), which contains the network protocol providing information on internet hostname, path and filename. End-users at the client machines can view this HTML data via popular web viewers such as Netscape, HotJava, Mosaic, Lynx, *etc.*

Although the web was first made available in 1991, it was only after the release of Mosaic by the National Center for Supercomputing Applications (NCSA) in January 1993 that it really gained prominence. Mosaic, NCSA's web client, made the web accessible to a wide and diverse user community because of its easy-to-use, graphical interface. Mosaic and the web succeeded in establishing a universal hypertext. With the release of Netscape Navigator in October 1994 by a commercial company co-founded by the original author of Mosaic, the number of end-users on the Internet escalated to a phenomenal figure. Today, the web is used by millions across the world. In just three years after its release in 1991, it had an estimated 30 million end-users (Nielsen, 1995b). It has changed the Internet to the extent that it has become almost synonymous with the modern use of the Internet.

However, the web is not without its problems. With the exponential growth of the web in 1994 and 1995, the problem of finding resources on the Internet became particularly acute, and this is made even worse by the web's reliance on hypertext links and its lack of meta-search facilities (Maurer, 1996). As the use of the Internet and the web grows, scalability not only refers to the handling of the increased number of servers, but also of handling the increased number of end-users. Because the web was not designed to handle so many and such large applications with more and more people using it, there are potential problems associated to the web (Maurer, 1996), of which the LIH is one of the more pressing problems. This view was supported by the results of the 4th Web User Survey by the Graphic, Visualisation and Usability Center conducted over October/November 1995 (Pitkow and Kehoe, 1995). From a sample size of more than 23 000 responses, the report showed that end-users suffered different forms and degrees of "lostness" (discussed previously in §2.3). Another problem reported in the survey that might be related to the LIH problem was that end-users were overwhelmed by the vast amount of information available, hence resulting in end-users not being able to organise the pages and information they gathered (25.8%). Poorly designed web browsers were identified by 3.2% of those surveyed as a problem area.

7.2 Survey of solutions to address the "lost in hyperspace" problem on the web

Much work has been done to address the LIH problem on the web. Some solutions are aimed at helping hypertext end-users, others are aimed at helping hypertext designers (see figure 7.1).

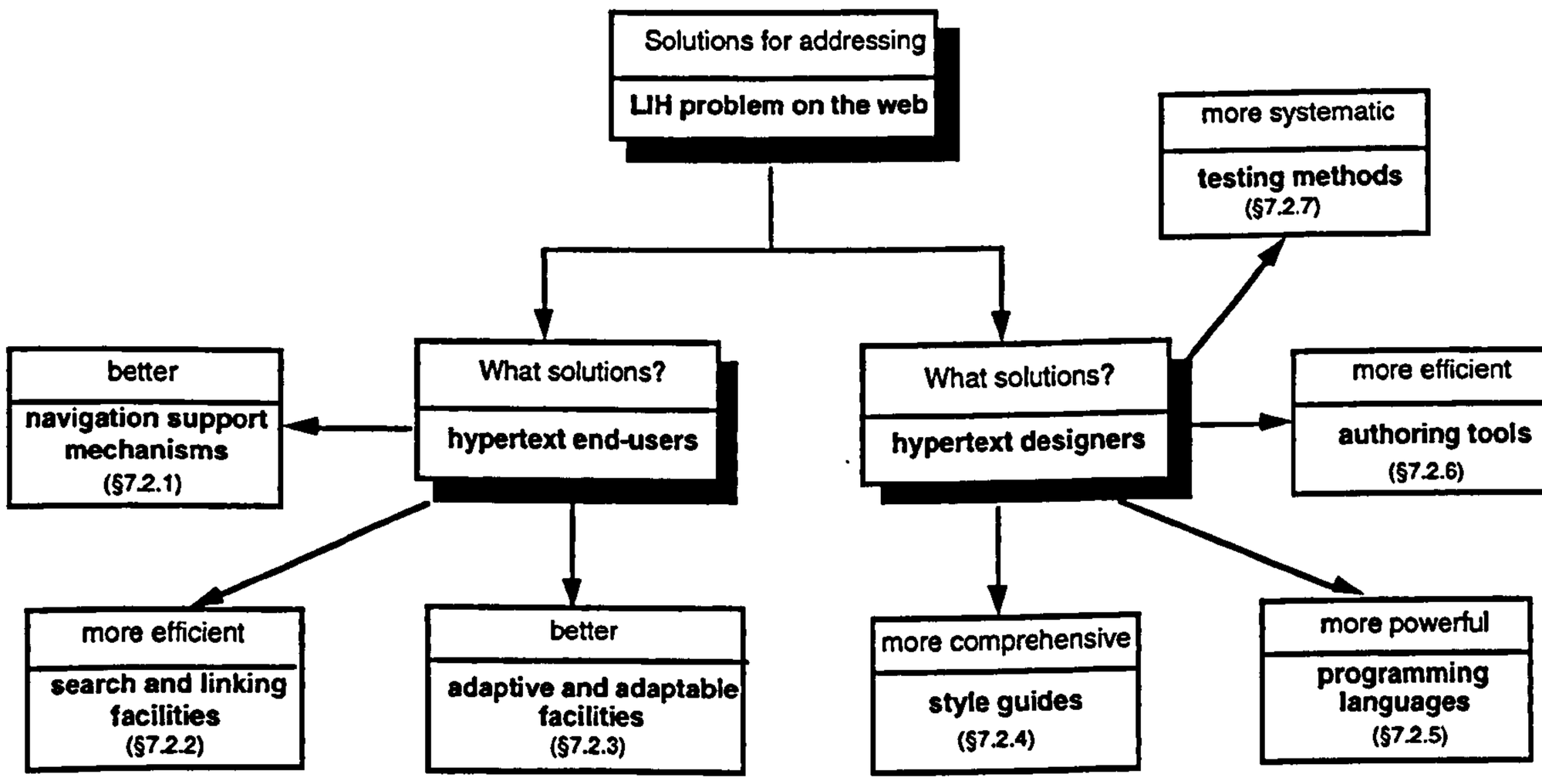


Figure 7.1. Solutions for hypertext end-users and hypertext designers

§7.2.1 – 7.2.3 describe solutions for the hypertext end-users which include providing better navigational support mechanisms, efficient search and linking facilities, and better adaptive and adaptable facilities. §7.2.4 – 7.2.7 discuss solutions for the hypertext designers which come in the form of more comprehensive style guides, more powerful programming languages, more systematic testing methods, and more efficient authoring tools, to help designers cope with increasingly sophisticated requirements to build larger, more complex hypertexts. §7.2.8 examines alternative solutions to the web.

7.2.1 Better navigation support mechanisms

Generally, navigation refers to the "art of plotting the course and finding one's position using maps and instruments" (Oxford Advanced Learner's English, 1985). Similarly, navigating on the web can be perceived as determining one's position within the overall space (document), and how to move to another place (piece of information) using navigational aids (Pittas, 1995). Naturally, to improve navigation within the web, good navigational support mechanisms should be provided. Nielsen (1995b) lists eight navigation support mechanisms that had been implemented in Netscape Navigator (version 2.01), the most popular web browser, to help end-user navigation (see table 7.1):

Table 7.1. Eight navigational support mechanisms implemented in Netscape Navigator

1. *Go to an absolute address.* End-users make use of a standard URL notation to go to an absolute address composing of three elements: the access method by which the client can retrieve the information object; the internet address of the server where the object is stored; and the address of the object in the server's file space.

2. *Hypertext links.* End-users can move to related information by clicking onto hypertext links, represented by underlined text or figure.

3. *Backtracking.* End-users are allowed to return to previously visited nodes using a backtracking feature.

4. *Bookmark.* End-users can build a set of direct jumps to their favourite places in hyperspace using bookmarks.

5. *History list.* End-users can go back to previously visited web pages since the start of the session using a generated history list.

6. *Breadcrumb.* End-users are reminded of web pages they have visited previously by spotting breadcrumbs shown by the change of colour of underlined hypertext links.

7. *Prospective view.* Before end-users make the jump, they are given prospective information about the destination node (URL with path and filename) provided in the footer.

8. *Landmark.* Directory buttons like "Home page" or "What's new?" provide landmarks for end-users to go back to whenever they feel "lost".

7.2.2 Search and linking facilities

As discussed in chapter 4, end-users using hypertexts perform tasks which can broadly be divided into "focused" or "unfocused" browsing. To help end-users accomplish the tasks, efficient search and linking facilities should be incorporated within hypertext systems. Campagnoni and Ehrlich (1989) conclude from their study that most test subjects using the Sun Help Viewer on-line help system for the Sun386i preferred a browsing search strategy over using the index. Book House Project (Pejtersen, 1989) was built using a metaphor which supported four different search strategies such as random browsing, search by analogy, browsing strategy and analytical search. Brown (1988) advocates using a "find" button, which can be viewed as an unstructured linking mechanism. Allison and Hammond (1989) recommend using metaphors such as index for more directed search, and a guided tour facility for exploratory search. However, once the size of the web exceeds browsable proportions, sophisticated search facilities such as keyword search, content search and fuzzy (inexact) search are indispensable for finding specific information (Maurer, 1996).

One of the biggest challenges on the web is finding something specific since there is so much information available on the web. More recent research conducted to provide more accurate, faster and more efficient search and linking facilities on the web include automating indexes (such as web robots or spiders) to walk the entire server tree, text compression techniques, machine learning techniques, *etc.* Examples include:

- Meta-search engines (for example, MetaCrawler Parallel Web Search Service; SavvySearch; ProFusion, *etc.*) use multi-threaded query gateway to query multiple search engines (for example, InfoSeek Search; Lycos; WebCrawler; Web Worm; JumpStation, *etc.*) simultaneously (InterNIC directory and Database Administration, 1997).

- The New Zealand Digital Library for Computer Science uses modern compression techniques to provide access to over 10 000 documents worldwide in computer science, and makes them available over the web through full-text interfaces (Witten, Cunningham, Vallabh and Bell, 1995).

- DEC's Library Information Access Client supports a card catalogue metaphor and represents individual searches as objects that can be moved and stored. The search results are colour-coded to let end-users know which results go with which searches (Scott, 1994).

- Maurer (1996) suggests having overview documents to represent the structure of the web, containing a list of links to other documents, an annotated diagram, map, *etc.* Hyper-G provides a full-text search and allows end-users to view the search results in the context of the collection hierarchy.

7.2.3 Better adaptive and adaptable facilities

The lack of support for typed nodes and links limits the richness of information which can be represented. A project, called MacWeb undertaken by Nanard and Nanard (1993), draws upon knowledge-based approaches to address the LIH problem, by extending the hypertext metaphor with typed links and typed nodes to represent knowledge in the hypertext as a semantic network (Clibbon and Callaghan, 1996). This solution provides great potential for building adaptive and adaptable hypertexts, taking into consideration end-users' needs and browsing patterns (Brusilovsky, 1996). End-users can navigate round hypertexts more efficiently with a reduced chance of getting LIH.

Cockburn and Jones (1995) propose building a graphical browser that dynamically adapts to, and reinforces, end-users' browsing actions and mental models, thus reducing the impact of the LIH problem. In contrast with overview documents, *graphical browsers* rely on dynamically generated structure maps that adapt to end-users' needs and they can come in various forms (Maurer, 1996): *global* maps show the entire hyperspace; *local* maps show the "vicinity" of the current node in terms of hyperlinks to and from other related nodes; and *fisheye* views focus attention on important nodes by deliberately distorting the view.

To help end-users tackle the problem of information overload as well as not to be "lost" in the galore of information available, there is a growing interest in using interface agents to make the web more adaptive to end-users' needs. Interface agents make software more active and work autonomously without waiting for end-users' command. One example of the use of software agents on the web is the investigation of personalised information filtering systems to help end-users eliminate irrelevant information and bring relevant information to end-users' attention (Maes, 1994).

7.2.4 More comprehensive style guides

Style guides describe the design principles and guidelines used to create hyperdocuments on the web. Many style guides have been written to help designers produce better, usable hyperdocuments. According to Tilton (1996), end-users' perception and assumptions about the organisation of websites can have a major impact on the usability of the page and site design. Therefore, designers need to give end-users a feeling of knowing where they are. Berners-Lee (1995) suggests structuring hyperdocuments using a tree structure, and designers can use this structure to organise files into directories.

To help end-users identify the origin and relationships of web pages, consistent and predictable web pages should be produced, which are important attributes of any well-designed interactive systems. For the purpose of the investigation, a sample collection of style guides such as the official Web Organisation Style (Berners-Lee, 1995), Carnegie Mellon's Web Style Guide (Tilton, 1996), Middlesex Style Guide (Thimbleby, 1995c), and Yale's Web Style Manual (Lynch, 1995) were surveyed to extract widely-accepted principles and guidelines. The essential elements that should appear on each web page are (table 7.2):

Table 7.2. Essential elements of a web page

- *Document header*. Unlike paper documents, designers of web pages can never be sure what other pages the end-user has seen before linking to the current page. Therefore, a meaningful title should occur at the head of the document to identify the content of the document in a fairly wide context.

- *Button bars or other navigation aids*. Text-labelled buttons provide fixed links between a series of pages to bind them into a document, for example, "previous", "next", "home", "table of contents" buttons, *etc.* Text-labelled buttons are preferred over icons because they clearly indicate their functions, and hence there is no confusion over their usage.

- *Links to other related pages in the local website*. These provide end-users with the ability to move to other related materials, and get to the desired information as quickly as possible without wasting time searching for it.

- *Page footer*. This is important in identifying the origin, authorship and author contact information. End-users can get an impression of the authority of the document and how reliable the authors are. They can also trace and clarify with authors regarding issues raised in the documents. Dates of creation and last modification in the footer provide end-users with timeliness and relevance of the materials. Web page footers should include a standard copyright designating the author or host institution as the copyright holder of the contents of the document.

Other elements that are crucial for good web page writing include the following:

- *Device-independent*. Because the HTML language used to write hyperdocuments does not contain information about fonts and paragraph shapes and spacing, it presents a great advantage in that the hyperdocument can be displayed on any platform (Tim, 1995). Therefore, designers should make the HTML codes as device-independent as possible, and not try to write them to suit just one particular client's machine.

- *Meaningful links*. Designers should not expect people to read web pages following designers' "sensible" path of flow through the work (Thimbleby, 1995a). Therefore, links should be put in to explain themselves so that end-users know where *they* are going, *etc.*

- *Length of web page*. Determining the proper length for any particular web page requires understanding the relationship between the page and screen size (Lynch, 1995). The loss of local context within scrolling pages is particularly disorientating when basic navigational elements like linkages to other local pages in the website disappear off-screen as the end-user moves through very long HTML pages. Berners-Lee (1995) provides a rough guide for the size of a hyperdocument. For on-line help with menus giving access to other things, they should be small enough to fit on 24 lines. Sizes of textual documents range from half a A4 page to 5 pages.

- *Keep the language simple and clear.* When describing links, do not say "see above", "see below", "described later", *etc.* because in hypertext, there is no order where above, below and later and they do not make sense (Thimbleby, 1995c).

- *Don't use too many typographical styles.* Flashing and fancy features (for example, blinking, large text, *etc.*) make reading difficult and unpleasant.

7.2.5 More powerful programming languages

All browsers display pages written in HTML. Netscape has been the driving force in the advancement of HTML, and is responsible for HTML tables, frames and background patterns, giving designers more control over the look of a web page. However, HTML does not allow authors to have much control over page and presentation layouts. But pushed by Microsoft, Netscape, and the Web Consortium, HTML is evolving. Web Consortium's HTML 3.2 specification includes style sheets which allow designers to specify many attributes of a paragraph using a single code. While HTML provides information about content, style sheets consist of style rules that tell a web browser how to present a hyperdocument (Pozadzides and Quinn, 1997). At the time of this writing, Microsoft Internet Explorer 3.0 and 3.01 are the only browsers supporting Cascading Style Sheets, which means that several different style sheets, each with a different order of importance, can be combined in order of importance to create a presentation style (Wium and Bos, 1996; Tilton, 1996). Netscape also pledges its support for style sheets in the next browser release, being part of the Netscape Communicator Suite.

JavaScript, a programming language from Netscape incorporated in their browsers, is also gaining great popularity because end-users are attracted to fanciful, animation features it can produce on the web. There are several features JavaScript can provide for existing websites (Harold, 1996) such as letting server draw pictures in a window on the client; using graphics primitives to create desired web page, putting less load on the server and allowing more user interaction. Recognising the potential in JavaScript and Style Sheets, the Web Consortium has defined a new standard called JavaScript Style Sheets.

7.2.6 More efficient authoring tools

HTML documents can be written in any text editor. Because of the popularity of the web, anyone can be a designer, resulting in a variety of authoring tools. These authoring tools can be divided in terms of functions and their purposes (see figure 7.2). HTML conversion tools and HTML editors are being developed to cater to the needs of different groups of designers with varying computing abilities and experience. HTML conversion tools allow designers to continue to use the word processor or page layout software, and then save these pages into HTML format. HTML editors, on the other hand, allow designers to create a web page from scratch using commands specially designed to give control over HTML's complex tags. Whatever the purpose for which the tools are built, the main and fundamental objective is to make authoring and maintenance of web documents as easy as possible. How sophisticated and well-explored these facilities are within the authoring tools depend much on the purposes for which they are built and for whom they are built. For this reason, this thesis divides the tools available in the market broadly into two categories: one for commercial

purposes; and the other for research purposes. For commercial tools, designers have the additional pressure to make the tools commercially viable and competitive, often at the expense of usability dysfunctionality. Research tools, in general, suffer from the lack of aesthetic appeals.

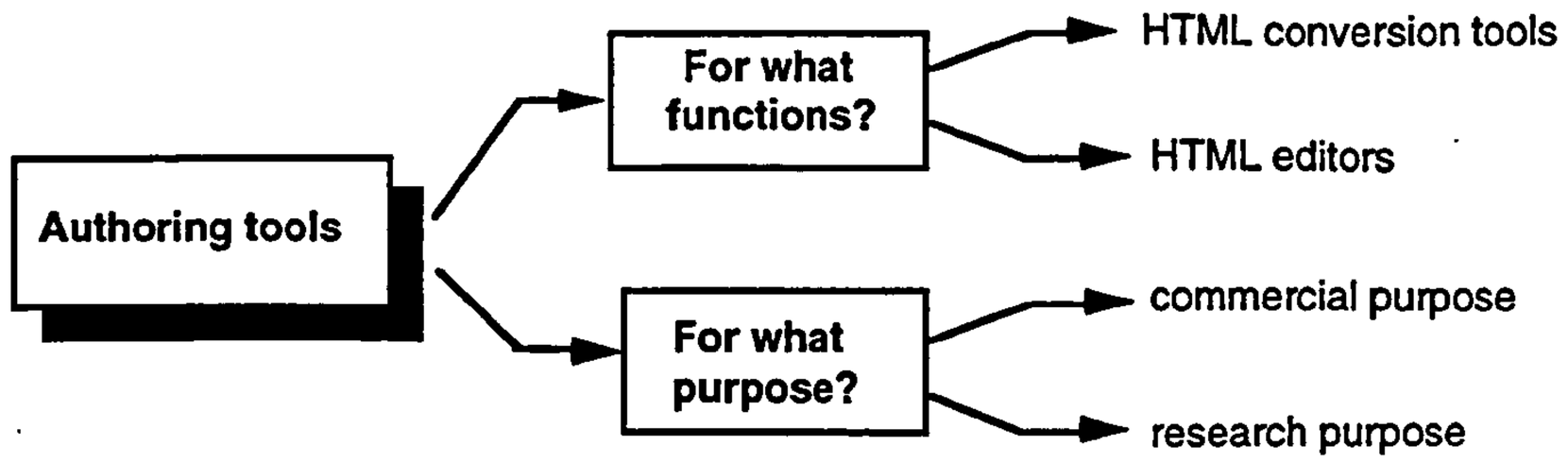


Figure 7.2. Types and purposes of authoring tools

- **Commercial tools**

  The range of web authoring tools available in the market is daunting. Therefore, this thesis will review only a few of the more popular web tools such as HoTMetaL PRO 3.0, Microsoft FrontPage 1.1, Navigator Gold 3.0, PageMill 2.0, NetObjects Fusion 1.0 and HotDog Pro (see table 7.3).

Table 7.3. Description of some popular commercial tools

HoTMetaL PRO 3.0 is a tool for experienced HTML coders. It offers numerous templates to work from, and a great selection of graphics files to brighten up the pages.
Microsoft FrontPage 1.1 is more than a simple web design aid. It is a powerful system for implementing and managing websites. It provides easy to use web editing tools.
Navigator Gold 3.0 looks just like the standard version of Navigator, which is familiar and easy to use. Its output code is neat and tidy. It is easily the most widely available of all the HTML editors.
PageMill 2.0 has a WYSIWYG interface which is excellent and easy to use. It has good support for Netscape plug-ins and animated GIF.
NetObjects Fusion 1.0 is a powerful page editor and site management tool. The entire website is stored as a database, and allows designers to have a high level of control over page layout.
HotDog Pro is more like a text editor which has been modified for use with HTML coding. It has good facilities for generating tables and forms.

HoTMetaL PRO 3.0 fares well with regard to the quality of code generated, allowing the designers to specify which version of HTML is to be used. HoTMetaL PRO 3.0 is less a design aid but a tool for experienced HTML coder (Fox, 1996). It does not, however, support link verification and site management tools. What is lacking in HoTMetaL PRO 3.0., Microsoft FrontPage 1.1 compensates. In fact, Microsoft FrontPage 1.1 tries to be all things to all people, being a collection of tools for website design including HTML editing, page testing, site management and team working. However, its great strength is also its great weakness, in that one can be confused over the structure of separate editor, security administrator, server and site administrator applications (Colborne, 1996).

Navigator Gold 3.0's simple and familiar interface makes anyone accustomed to using the Netscape browser feels comfortable with it. But it can be quite frustrating to use since all

the documentation, help and templates are on-line. The inflexibility of the template to offer customisation is often criticised for the creation of drab, unimaginative pages (Colborne, 1996). PageMill 2.0's interface is consistent, engaging, and familiar to anyone who has used page layout software. According to Colborne (1996), nothing comes close to its ease of use. There is, however, a lack of site management tools to keep track of links and page hierarchies. NetObject Fusion 1.0 is a site management tool which leads designers through the process of mapping out a site, choosing a "look-and-feel", and finally generating individual pages from database information. One of the criticisms is the use of inflexible templates to generate standard, monotonous pages, thus restricting designers' thinking. HotDog Pro is more like a text editor which has been modified for use with HTML coding, and provides good facilities for generating tables and forms simply.

- **Research tools**

  Like commercial tools, much work has also been done in the research arena to explore facilities not fully exploited in popular commercial tools. This thesis will review a few of the more recent development of these tools here to give a flavour of the kind of authoring facilities which have not yet been fully investigated in commercial tools. Thimbleby's Gentler (1997) is built using the premise that readers and authors do semantically similar things, and hence dual user interface requirements are needed to support their activities. Gentler provides a basic text-oriented HTML page editor, where navigation links are generated and automatically maintained, requiring no additional effort from designers. Pontin and Neto's link-oriented tool (1996) evaluates the quality of the web documents by performing an analysis on the structure of the links. The authors claim that this will lead to less costly and lengthy modelling stages since the quality of the authorship is enhanced and better hyperdocuments are produced.

  Pitkow and Jones' Atlas (1996) has been conceived to solve the many problems daunting the continued utility and manageability of the web. Its main contributions involve the automatic maintenance of links across distributed servers, and the infrastructure for supporting efficient end-user visualisation. Creech's CLT/WW (1996) approach attempts to address the major issue of keeping web structures consistent whenever pages are moved, deleted or changed. It is an author-oriented link management technique that notifies authors of needed changes and automatically updates authors' documents. The author claims that the CLT/WW approach is a viable way of moving sites and enterprises toward a state of web consistency. Mea, Beltram, Roberto and Brunato's HistMaker (1996) is a HTML generator that allows the generation of maintainable, multilingual hypermedia medical cases for diagnostic reference and training. It makes use of a general-purpose construct used to structure parts of a document as database records, and to automatically generate HTML codes, thereby enabling more effective authoring, browsing and searching of web documents.

7.2.7 More systematic usability testing methods

Nielsen (1995a) recommends using competitive-usability analysis to evaluate the conceptual design of web documents, after visiting a competitor's website but before designing and prototyping the new interface. This would help to guide the design of web documents, and detect usability problems as early as possible in the usability engineering life-cycle, taking advantage of the heavy investment competitors have made in web design. Jones and Hewitt (1997) found that using experts to carry out heuristic evaluation of websites extremely useful and cost-effective. Berners-Lee (1995), on the other hand, advocates carrying out more systematic testing on web documents to ensure that they are well-designed and structured. Thimbleby's Gentler (1997) provides mathematical insight into hypertext document design and quality control by visualising link connectivity as graphs and working out network statistics. Designers should always proof-read to avoid making "silly" spelling mistakes. Designers should test-run the web document using several different client programs to ensure that it has been coded in a device-independent way (Tilton, 1996; Thimbleby, 1995c). Server log files can be used to monitor the readership of the web pages (Shneiderman, 1997; Busch, 1997; Berners-Lee, 1995). Another way to test web documents is to invite feedback from readers (Berners-Lee, 1995).

7.2.8 Alternatives to the World Wide Web

So far the solutions discussed above are aimed at improving the performance of the web. Due to the exponential growth in web usage as well as an avalanche of servers, documents and hyperlinks, there is a grave concern as to how websites can be efficiently maintained. Hence, alternative solutions to the web are sought. A well-known example is the development of Hyper-G at the Graz University of Technology, headed by Hermann Maurer and Frank Kappe. Hyper-G, a second-generation hypermedia information system released in 1994, tries to combine the advantages of the web, WAIS (Wide Area Information Service), and Gopher while minimising their disadvantages. Hyper-G claims to have overcome some of the shortcomings of the web, such as providing bi-directional linking and automatic link consistency, storage of hyperlinks in a separate link database, hierarchical structuring and composition facilities, integrated indexing and search facilities. These improvements made over the web aim to reduce the impact of the LIH problem, experienced by both web designers and end-users alike (Maurer, 1996).

Another example is the development of Microcosm, an open hypermedia system for creating, browsing and querying large collections of web documents, by the University of Southampton. Microcosm does not suffer from some of the problems of the web (Hall, Carr and Roure, 1995). It has been applied successfully to unstructured hypertexts on the web, providing a Microcosm interface to web page design and retrieval, thus making navigation and information retrieval simpler and faster. While the web is synonymous with breadth in hypertext systems, the designers of Microcosm claim that it brings depth of open hypermedia to the web. The two distinguishing features of the open hypermedia approach are the separation of links from the documents they apply to, and the use of existing documents and other media sources without having to change the originals in any way. Webcosm, Microcosm's initial web server

implementation, interprets each web page on the fly. When a given end-user links to a new page, Microcosm generates an instance of the page with standard web hypertext links for each of the links that the linkbases stored in the server, that are relevant for the new page. The flexibility of Microcosm separating the link structure from the data in the system to enable separate link and data processing, makes authoring easier for designers.

7.3 HyperAT: Implementing the proposals

Although much research effort has been invested to address the LIH problem, solutions have only been marginal. *Ad hoc* methods of designing, constructing and validating hypertexts are not enough. If designers get "lost" in hypertext, end-users do too. This suggests that tools for designing hypertext should provide improved support for designers. Nielsen (1996) predicts that due to a change in the dominating styles for websites over recent years, a real contribution essential for web design should consist of further research into these different knowledge areas (see figure 7.3): (i) knowledge of icon design; (ii) knowledge elicitation to discover appropriate information space structures; (iii) usability testing; and (iv) task analysis techniques.

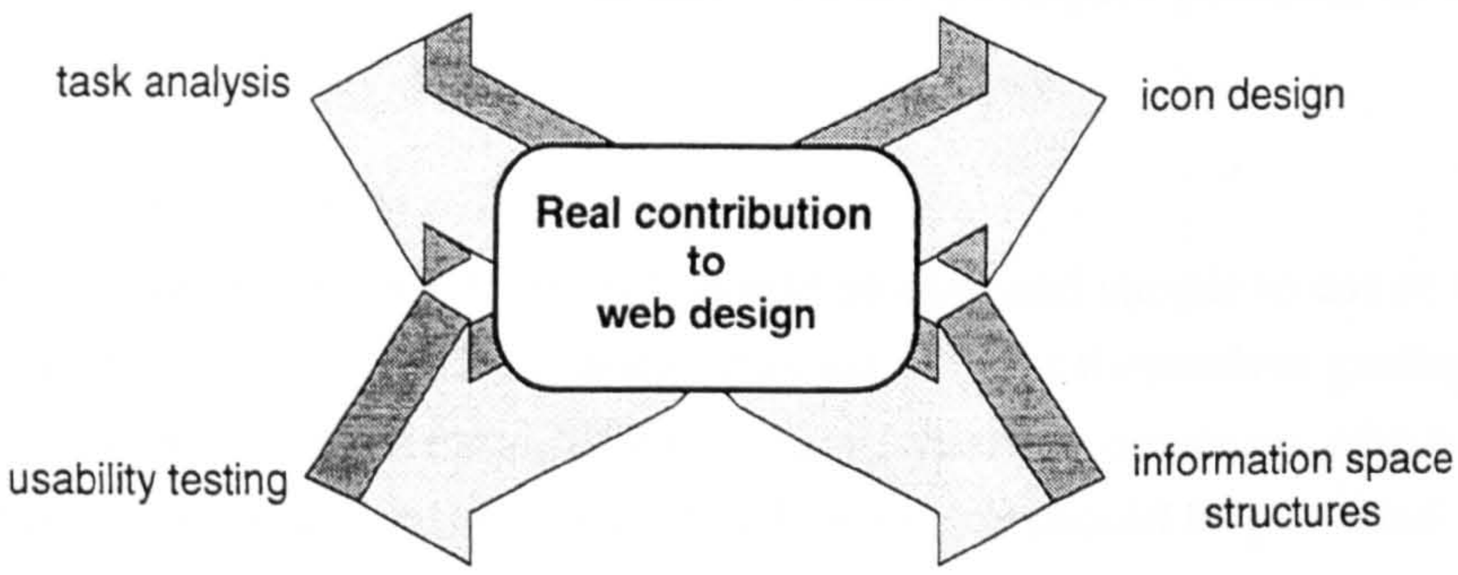


Figure 7.3. Real contribution to web design

This thesis agrees that this is the way forward if the performance of the web is to be enhanced. But searching for solutions in isolated disciplines, and recommending them to designers in the hope that they would somehow remember to put them into practice, may not be as simple as it sounds. Many factors could have prevented well-intentioned designers putting these good suggestions into practice. One of which could be that designers might be too overwhelmed, and/or seemingly might not have the time and capacity to attend to all these authoring details.

In order for Nielsen's suggestions to be truly effective and implementable, this thesis argues that it should go beyond just providing designers with a list of do's and don'ts. Designers need authoring help. If some of these ideas could be automated so that designers need not worry about their implementation, chances are that better hyperdocuments could be produced since designers would be freed to concentrate on other critical issues that cannot be automated, but require sound human judgement and expertise.

The remaining part of this chapter describes HyperAT, a research tool to help designers manage the complexity of the design and validation processes without themselves getting

"lost". The approach taken in HyperAT is novel in that *multi-disciplinary* approaches are integrated and culminated onto a practical authoring tool.

7.3.1 Design requirements

HyperAT stands for "Hypertext Authoring Tool". HyperAT is a prototype designer tool for authoring hypertext and web documents. It is implemented in Macintosh Common Lisp (version 3.9) for PowerPCs. This thesis would like to emphasise at this stage that it is not the intention of HyperAT to provide a full range of editing facilities with attractive interface, and HyperAT does not claim in any way capable of competing with commercial tools in this aspect. However, being a research tool, HyperAT aims to investigate facilities not seen or fully exploited in popular commercial tools which are crucial in helping designers build more usable web pages. This thesis intends HyperAT to be contributing in these areas:

- *An experiment in collaborative efforts involving many disciplines.*

  HyperAT draws upon and integrates knowledge and findings in seemingly diverse disciplines such as hypertext, human-computer interaction, cognitive psychology and software engineering that may be necessary to solve complex problem, of which the LIH is an example.

- *A practical authoring tool.*

  As a practical authoring tool, HyperAT should be easy and simple to use so that designers can manage the complexity of the design process without themselves getting "lost". Since HyperAT is aimed at novice and/or elementary hypertext designers who may have little knowledge in web authoring, automated authoring aids should be provided.

- *An analytical research tool.*

  To help designers build usable web documents so that end-users can navigate them without feeling "lost", HyperAT should provide facilities to help designers perform usability testing on the web pages produced.

7.3.2 General overview and underlying concepts

Figure 7.4 gives a general overview of HyperAT, its inputs and outputs. Inputs refer to the *multi-disciplinary* approaches (as described in chapters 3 – 6) that underlie the design of the authoring and usability components. Because the web is a special hypertext, these approaches had to be adapted for use on the web. Many interesting ideas came out from investigating these approaches. To recapitulate, Approach One (as discussed in chapter 3) examines design principles, and in HyperAT, it is examined in the form of good web style guides. Approach Two (as discussed in chapter 4) emphasises the importance of understanding end-users' needs and the tasks they perform. In HyperAT, this thesis explored end-users' browsing needs on the web. Approach Three (as discussed in chapter 5) stresses good structuring, and therefore in HyperAT, good web page structure to help both designers and end-users is investigated.

Outputs are the deliverables produced by HyperAT. Besides providing the basic authoring facilities to produce web pages, HyperAT also delivers usability results to designers regarding any usability problems that might be detected during its analysis.

It is a well-known fact that an understanding of HCI elements essential for any interactive system is crucial in designing successful interactive systems. In designing the authoring components, incorporated within HyperAT are two underlying design concepts: (i) the need to impose a structure (Approach Three), and (ii) the need to incorporate good web style principles (Approach One).

For the usability components in HyperAT, features that help designers understand end-users and their browsing behaviour better (Approach Two) are implemented. §7.3.3 - 7.3.4 describe the implementation of the authoring and usability components within HyperAT.

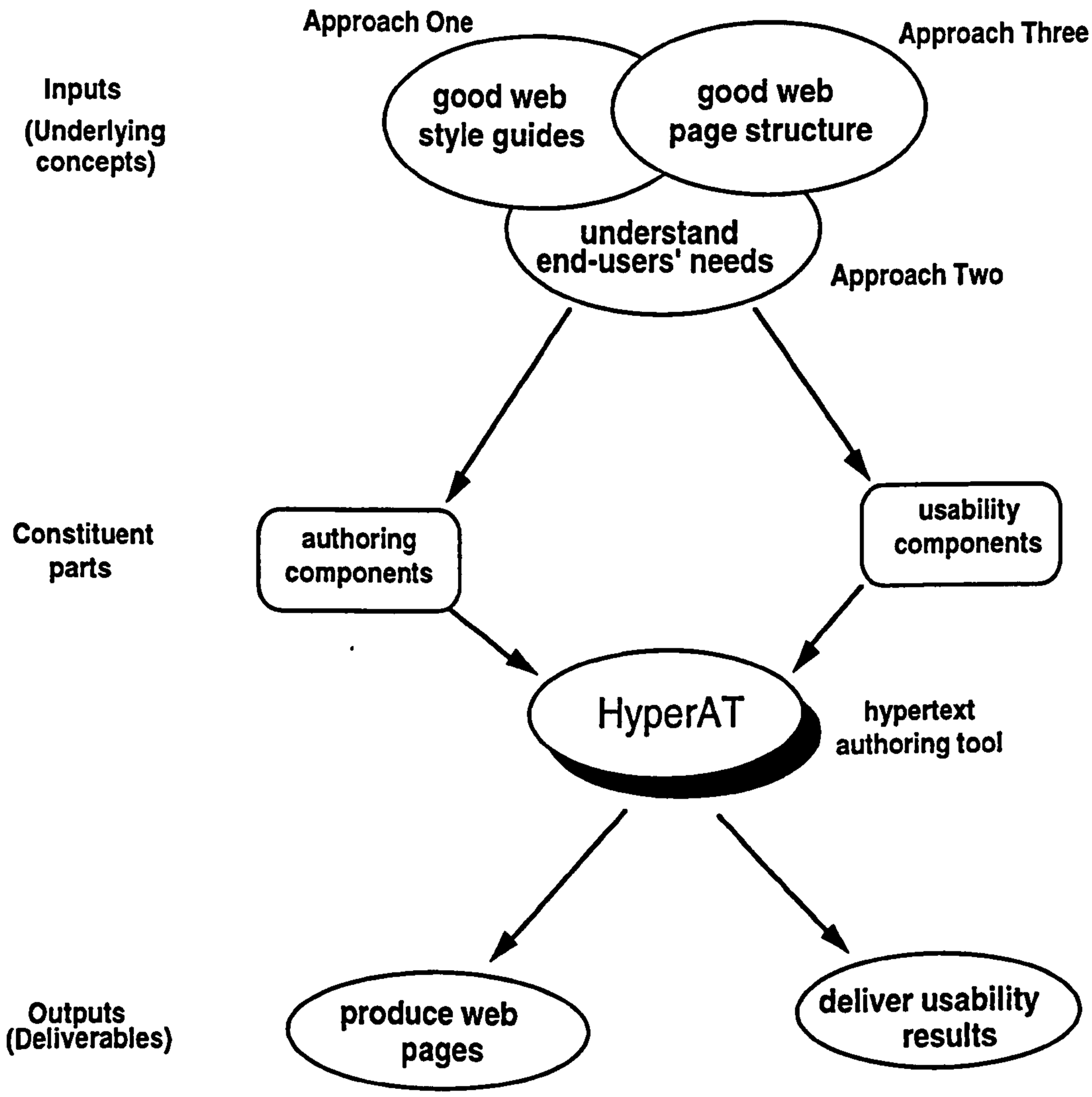


Figure 7.4. General overview of HyperAT, its inputs and outputs

7.3.3 Authoring components

The main objective of HyperAT is to help novice and/or elementary designers build usable, well-structured web documents. By that, this thesis refers to a web document with the following characteristics: (i) no links that go nowhere; (ii) no pages that are not linked; and (iii) minimum number of links traversed to reach required pages. The authoring components in HyperAT provide the basic authoring environment for the creation, loading and modification of hyperdocuments (see figure 7.5). To reduce the complexity in authoring, inbuilt into HyperAT is the automated structuring of nodes (Approach Three), thus freeing novice designers to concentrate on the content of the web pages to be created. Node relationships are described by automatic assignment of links. During the conversion of the hyperdocuments into HTML codes, HyperAT also generates a table of contents, a hierarchical representation of the structure of the hyperdocuments so as to provide end-users with global and local views for informed navigational decisions.

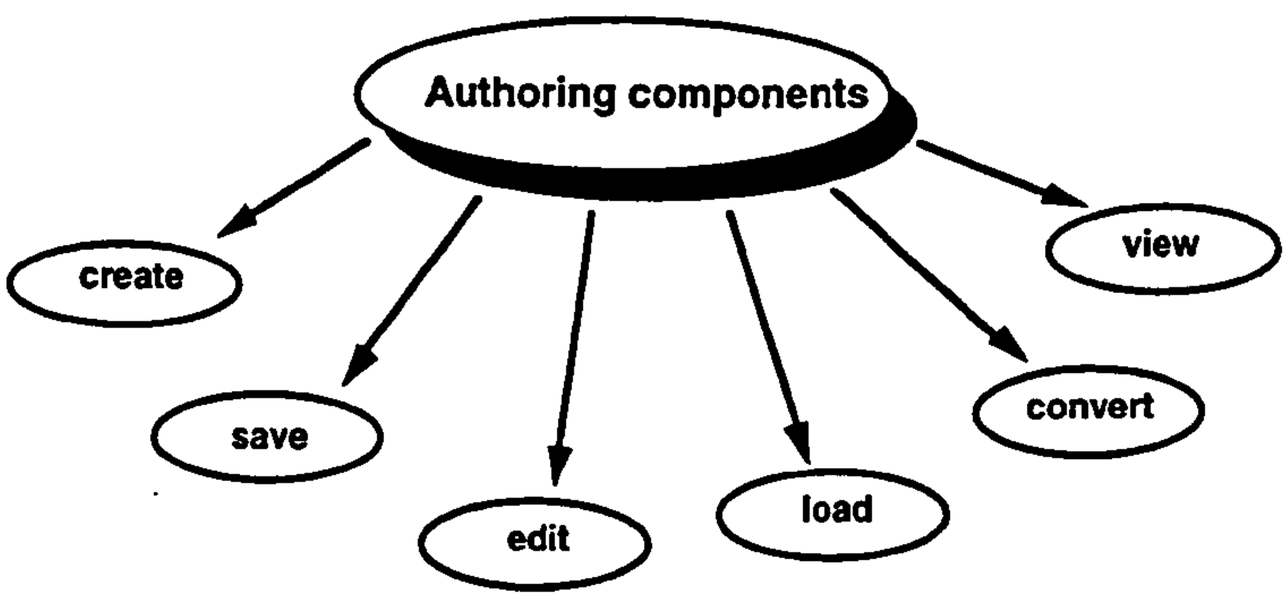


Figure 7.5. Authoring components of HyperAT

§7.3.3.1 describes how nodes and node relationships are defined in order to implement the automated structuring of nodes. §7.3.3.2 describes two authoring aids implemented to help designers. §7.3.3.3 describes the implementation of some well-established web design principles in the web documents produced by HyperAT. Since they are automatically inserted, designers can be certain that the final web documents would adhere to web design principles.

7.3.3.1 Automated structuring to reduce complexity in authoring

Despite its broad appeal, one of the limitations of the web is that there are no information structuring facilities beyond hyperlinks (Maurer, 1996). As discussed in §5.3.3, since hierarchies are easily understood and used by both hypertext designers and end-users (Smith and Newman, 1996; Berners-Lee, 1995; Lynch, 1995), incorporated into HyperAT are quasi-hierarchical structures as framework for capturing node information. To some, this may be forcing structure too early in the design process, which is not desirable (Halasz, 1987). However, a counter-argument is that divergence can be prevented in hypertext, normally the reason for end-users being LIH, and limitation of the hypertext structure is a good way to do that (Am, 1994).

Before discussing how the hierarchical framework is implemented, this thesis will define firstly, the types of data that are captured in the nodes and secondly, how relationships between nodes are represented.

- **Node**

  In HyperAT, a node refers to a fundamental unit of information representing a web page. Each web page has the following attributes:

  i. *Hypertext name* refers to the name of a web document to be created.

  ii. *Parent node name* refers to the parent of a web page.

  iii. *Node name* refers to the name of a web page.

  iv. *Window title* refers to the title of a web page.

  iv. *Icon file* refers to any icon file associated with a web page.

  v. *Node text* refers to the body of text which may/may not contain hotspots or cross-referenced links in a web page.

These attributes are captured in HyperAT via a form-like screen shown in figure 7.6. They are then written into as Lisp data structures and stored into a working file.

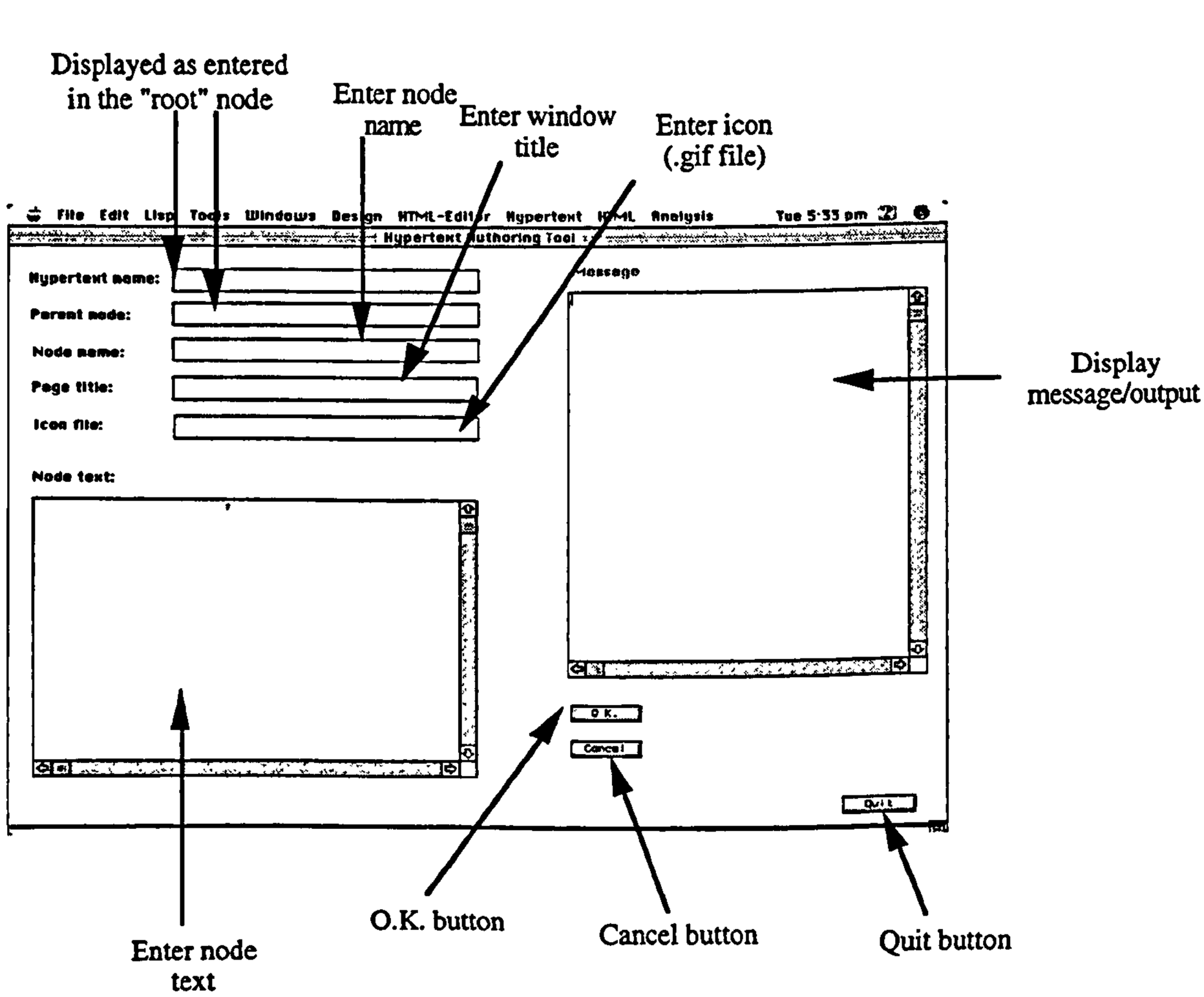


Figure 7.6. Entry screen for creating nodes in HyperAT

Two other attributes of a web page are automatically assigned by HyperAT based on designers' inputs. They include "children nodes" which refer to web pages hierarchically related to a web page, and "neighbour nodes" which refer to web pages referenced by cross-referenced links in the node text.

- **Node relationships**

  HyperAT captures node relationships using a simple parent-child analogy. This simple way of representing node relationships is not only "intuitive" to designers but powerful in constructing data structures in Lisp. Node relationships are expressed in terms of the associations between nodes. There are two common kinds of associations between nodes and they are represented in terms of hierarchical and cross-referenced links. Figure 7.7 shows a simple example of a hypertext with 4 nodes related to each other by hierarchical or cross-referenced links.

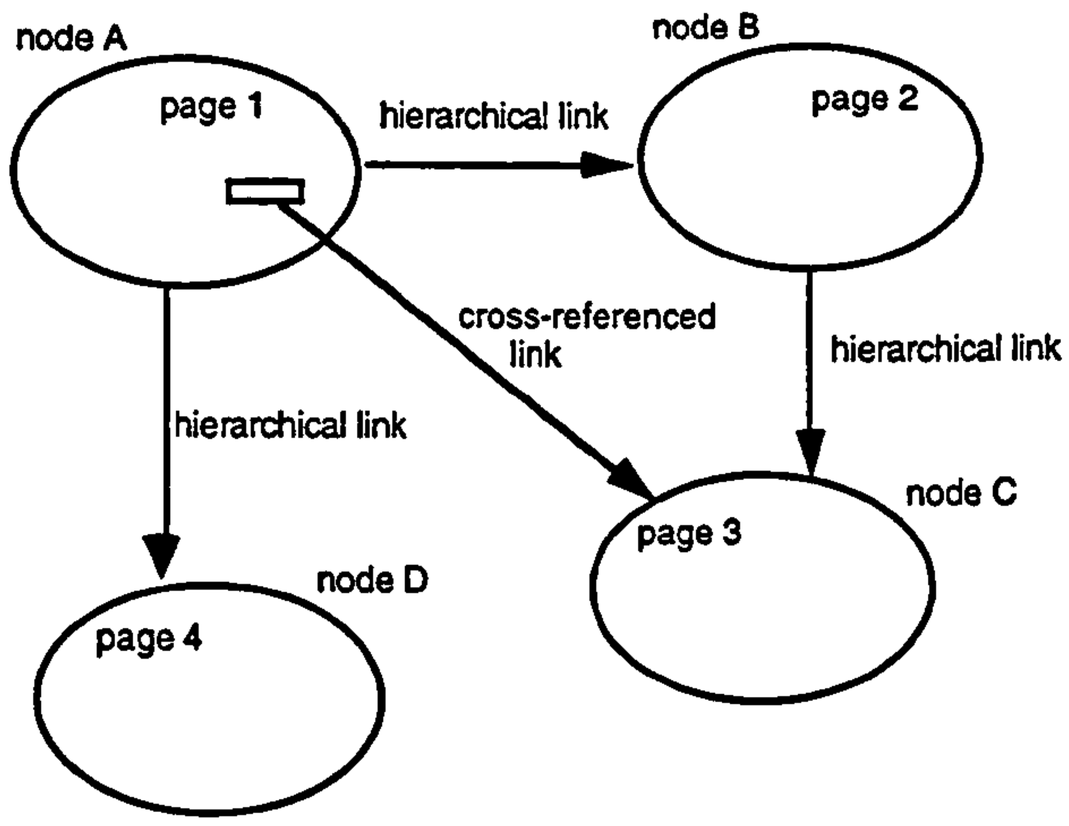


Figure 7.7. Relationship between nodes A, B, C and D

Representing these links in HyperAT is simple. Using figure 7.7 as an example, the relationships among nodes A, B, C and D are described by these facts:

- *Fact 1*: "node A is hierarchically linked to node B"
- *Fact 2*: "node A is referentially linked to node C"
- *Fact 3*: "node B is hierarchically linked to node C"
- *Fact 4*: "node A is hierarchically linked to node D"

To begin, a form-like screen (see figure 7.6) is used to input information on node A. To represent the relationship between node A and node C, a HTML <a href="c.html"> tag to indicate a cross-referenced link from node A to node C is entered into the node text. When complete, press "O.K." to save the information about node A. A prompt then appears to ask if another node is to be created. If "yes", a dialog box appears for the name of the parent node of the new node to be entered. In this case, type in "A" since the next node to be created is node B. Once done, HyperAT captures this information and generates a new input screen, into which information about node B is entered. The process repeats for node C and the

name of the parent node is "B", since node C is hierarchically linked to node B. This same process is carried out for node D. Figure 7.8 shows the various input screens recording how these four facts are captured in HyperAT.

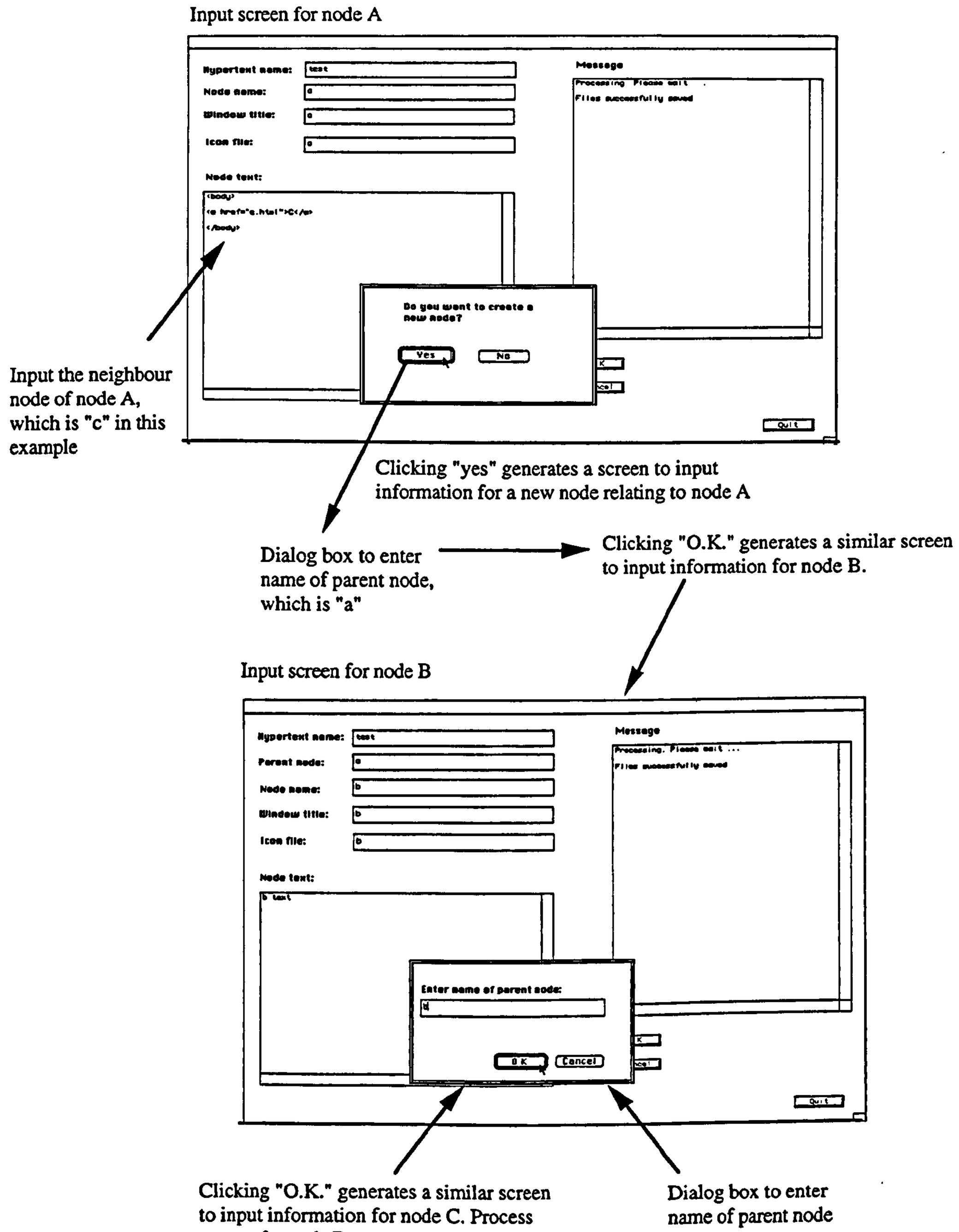


Figure 7.8. Some screen shots to input information on nodes A, B, C and D

HyperAT then writes into the data structures of nodes A, B, C and D these facts into a file (see figure 7.9):

- *Node A*: "node A has a child named node B", "node A has a neighbour named node C".
- *Node B*: "node B has a parent named node A", "node B has a child named node C".
- *Node C*: "node C has a parent named node B"
- *Node D:* "node D has a parent named node A" (not shown in figure 7.9)

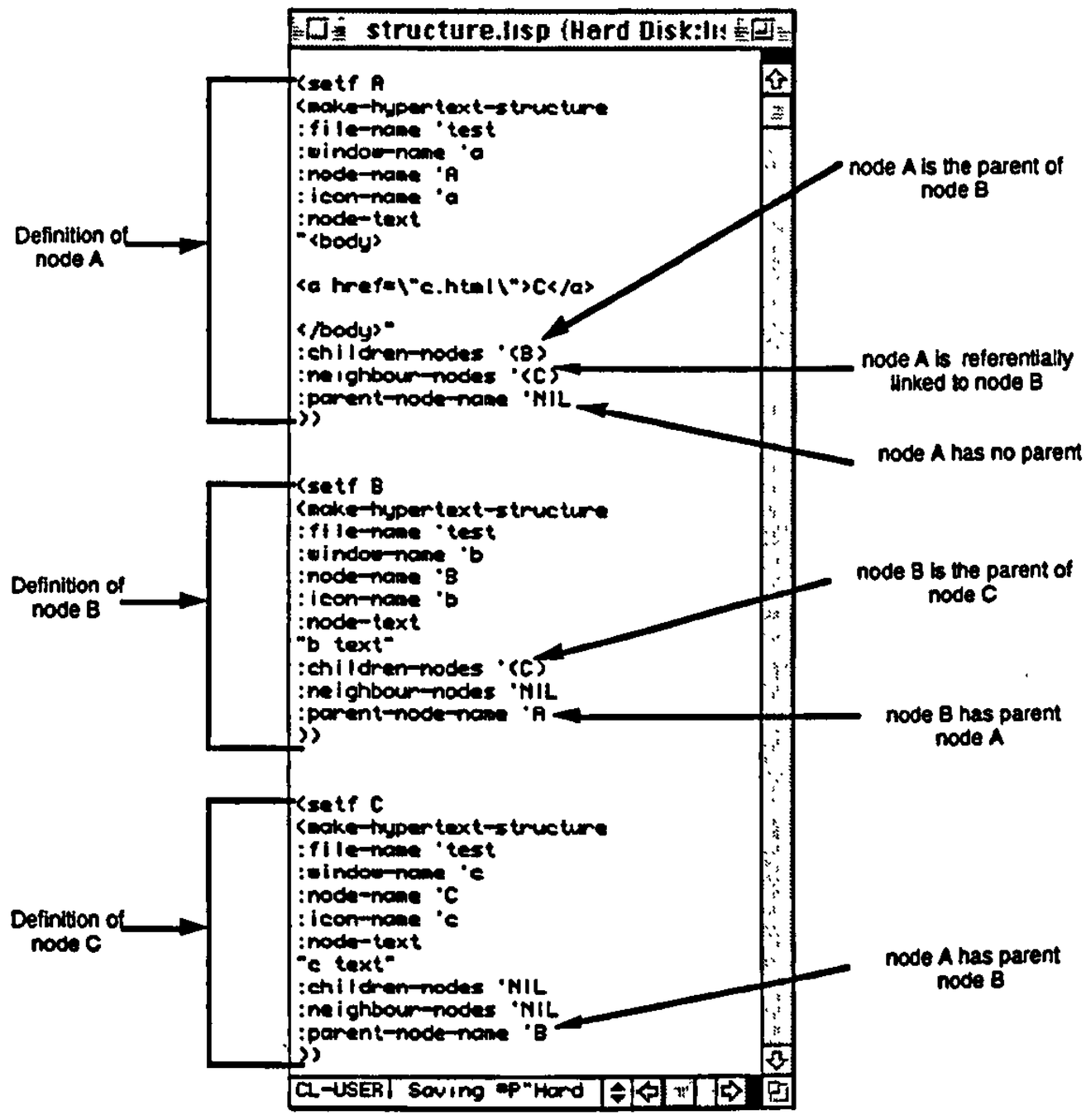


Figure 7.9. File containing data structures of nodes A, B, C and D

Note that designers do not need to write into the structures of nodes their relationships with other nodes, it is automatically taken care of by HyperAT. Because of the way the attributes of nodes A, B, C and D are captured and represented, modifying and updating of nodes can be easily carried out. This is important since developing a hyperdocument is an iterative process in which decisions need to be revised as the hyperdocument is being produced. Supporting incremental modification of the hyperdocument increases flexibility and minimises the detrimental effects of such changes.

- **Generated hierarchical structure**

  As mentioned previously, the data about the nodes are stored into Lisp data structures. To display a hyperdocument consisting of these four nodes, data is converted into the HTML format (see appendix F2 for the Lisp program to perform the automatic conversion of data to HTML format). During the conversion process, a hierarchical structure of the web document is generated. This structure is then incorporated into the HTML codes to provide end-users with global and local views of the structure of the web document, which will be

described in more detail in §7.3.3.3. Figure 7.10 shows part of the web document displayed using Netscape.

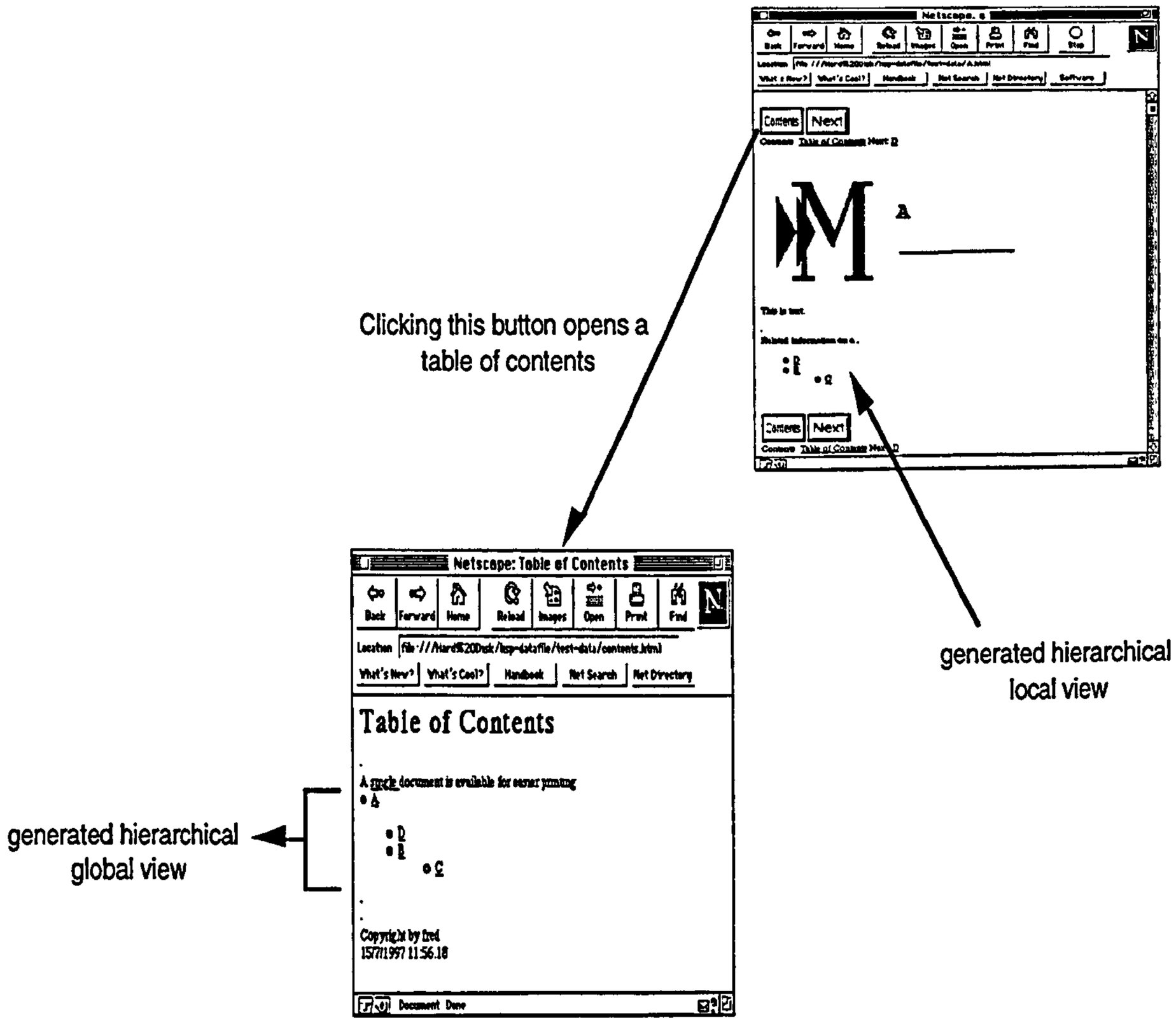


Figure 7.10. Web pages showing generated hierarchical global and local views

7.3.3.2 Authoring aids

Besides providing within HyperAT an automated structuring feature to represent node relationships, other authoring aids were also implemented to help designers. Owing to time and resources constraints, two aids were selected since they have great potential in helping designers manage the complexity of the design process without them getting "lost".

- **Generated trace of created nodes**

  Designers are provided with a generated trace of created nodes since the start of the HyperAT session. Figure 7.11 shows a sample trace displayed in the Display Window of an input screen for the inputting a new node, in this example, node D. The trace of created nodes is generated dynamically for every new input screen. These traces provide useful memory jots for designers, who may be interrupted during the HyperAT session or are simply confused over the nodes created.

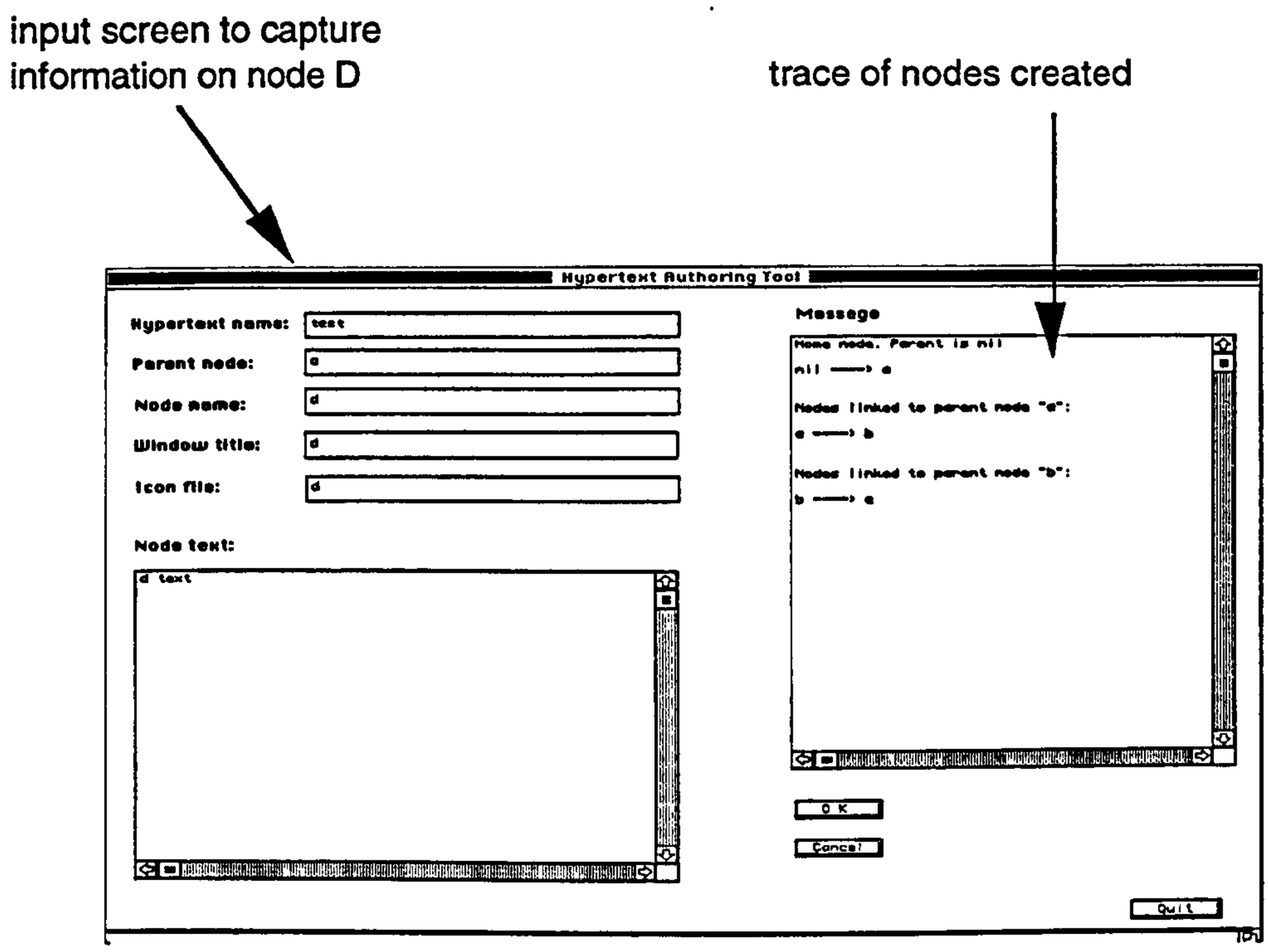


Figure 7.11. Input screen with trace file displayed in the Display Window

- **Generated map**

  Designers can request for a generated global map of the structure of the hyperdocument showing its constituent nodes. Clicking onto each node will bring up another map, with that node as the root node, providing designers with a local view cancelling off other details not related to it (see figure 7.12).

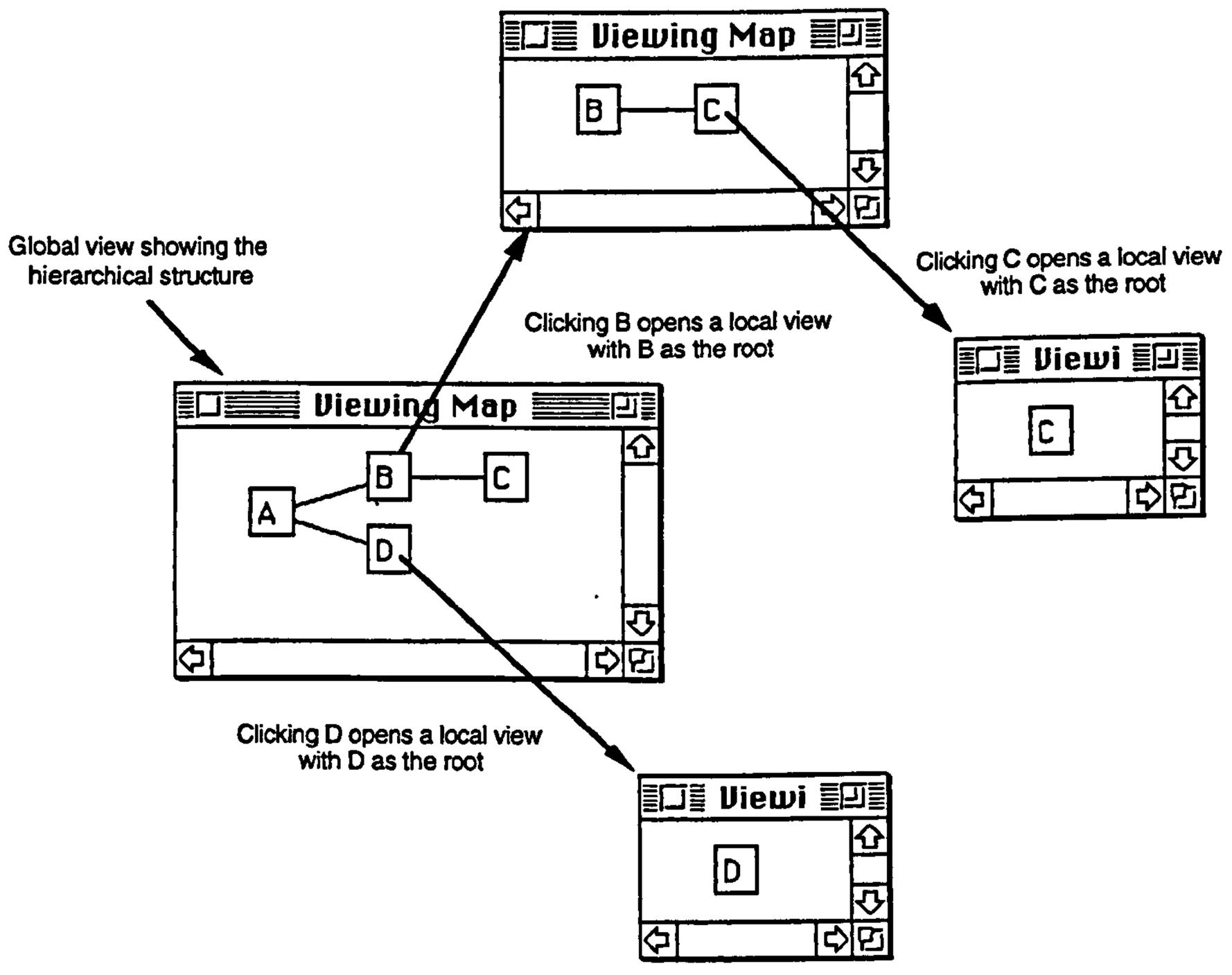


Figure 7.12. Global and local maps showing structure of a sample hyperdocument

7.3.3.3 Web design principles

Designing, structuring and maintaining websites is difficult. Not only should designers ensure that websites are structurally sound to prevent end-users from getting LIH while surfing the web, they have to create websites that are aesthetic enough to attract end-users. Proper web page design is largely a matter of balancing the structure and relationship of menu or home pages and individual content pages or other linked graphics and documents. The goal is to build a hierarchy of menus and pages that is natural and well-structured to the end-user, and does not interfere with the use of the web pages or mislead them (Lynch, 1995). Given that there are potential difficulties in creating web pages that are both easy to use and full of complex content, Tilton (1996) proposes that the best strategy is to consistently apply a few basic document design principles in every single web page designers create.

HyperAT implemented a few established web design principles and showed that they could be automated in any authoring tool without designers having to worry about their implementation. Figure 7.13a – 7.13b show sample web pages generated by HyperAT incorporating some of the suggestions made in table 7.2 (as discussed previously in §7.2.4) such as automatically providing a document header, text-labelled buttons, local view of related pages and page footer identifying the origin, authorship and author contact information.

A sample web page

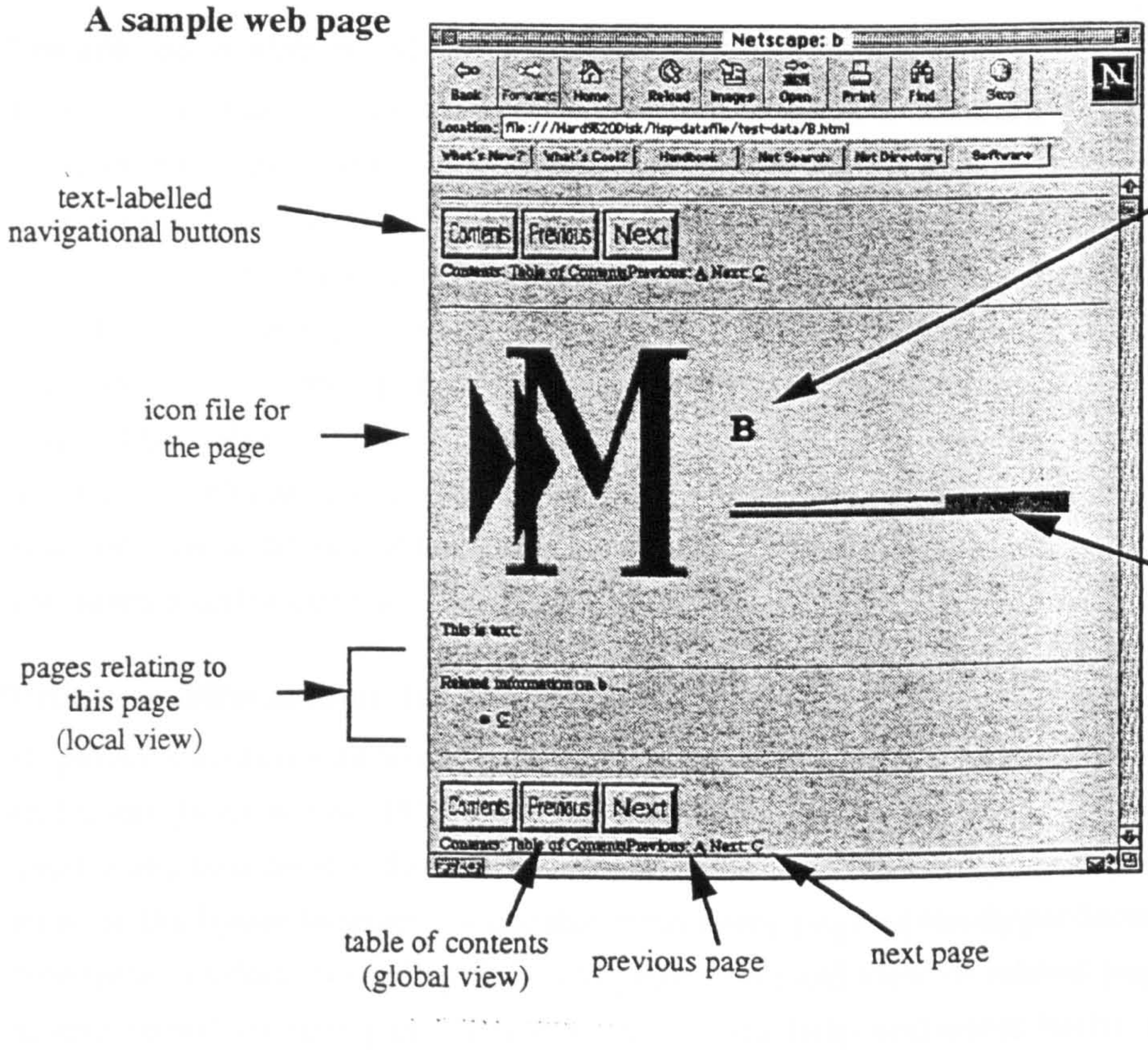


Figure 7.13a. Sample web page generated by HyperAT

"Table of contents" page

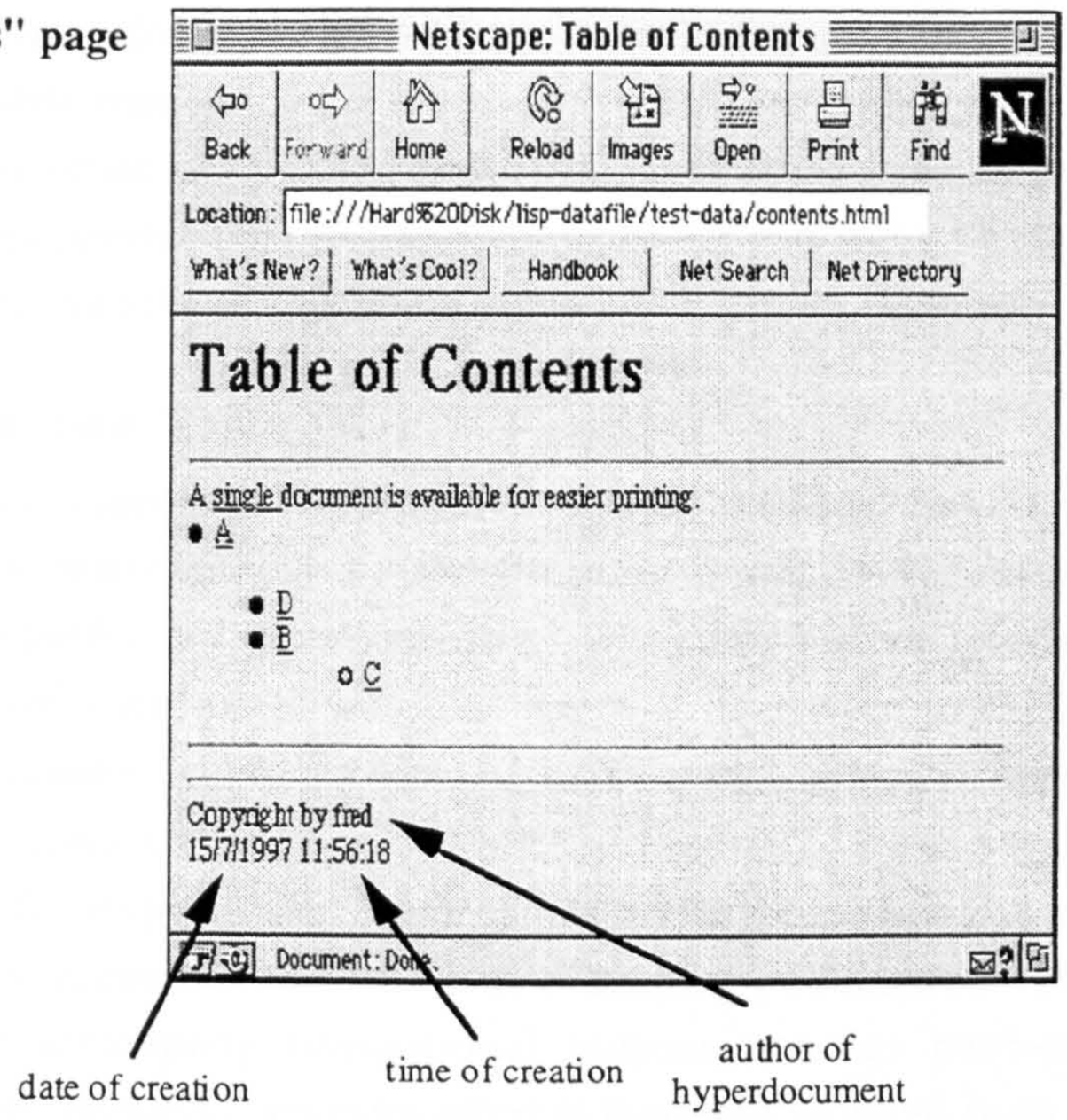


Figure 7.13b. Sample "table of contents" page generated by HyperAT

- **Ensure consistency of presentation**

  To ensure consistency of presentation, every page has a standard "look and feel" with navigational buttons at the top and bottom of a web page. On each web page, text-labelled navigational buttons which represent fixed links that allow end-users to move to "content" page, "previous" page, or "next" page are provided. Links to other related web pages giving a local view of the structure in relation to the end-users' current page are also generated for each web page so that end-users know how to navigate to relevant materials easily. The "table of contents" page contains a hierarchical representation providing end-users with a global view of the structure of the web document. The "table of contents" page also contains important information such as the author, date and time of creation, which are automatically generated by HyperAT.

- **Structure information for the end-users**

  HyperAT structures information using hierarchies because they are easily understood by end-users (Berners-Lee, 1995; Lynch, 1995). These hierarchies are made known to the end-users using two design ideas. One idea is to generate a table of contents providing a global view of the hyperdocument accessible from every page of the hyperdocument, using the "contents" button. The other idea is to provide a local view of related pages with respect to end-users' current page. These views would help end-users better understand the structure of the hyperdocument in relation to where they are, thus ameliorating the LIH phenomenon.

- **Provide status of the hyperdocument**

  The bottom of the table of contents web page shows the page footer with generated information on the authority of the document. Information such as author, date and time of

creation are provided. This information is useful to both designers and end-users. For designers, it would enable them to keep track of the different versions of the web documents produced, and would also provide useful insights into understanding the design process involved in the development of the web documents. For end-users, this information provides them with the relevance and timeliness of the information presented.

- **Provide a prospective view**

  Web browsers provide end-users with a prospective view by showing the URL with path and filename in the footer *before* end-users make the jump (Jones, 1996; Nielsen, 1995b). Adopting this idea, HyperAT provides end-users with prospective information by generating the title of pages end-users would move to if they were to click onto the navigational buttons, thereby helping them to make more effective navigational decisions. For example, "previous" and "next" buttons in figure 7.13 indicate moving to pages named "A" and "C" respectively. These names are more meaningful since they reflect titles of web pages, in contrast to URL's way of naming pathnames. Because these prospective views that accompany navigational buttons are not hard-coded but automatically generated by HyperAT, no extra effort is therefore required from designers to ensure the inclusion and maintenance of this feature.

So far, this thesis has shown how the authoring features in HyperAT are designed to enable designers to manage the design process to ensure proper structuring and presentation of information.

The next section describes the usability components implemented in HyperAT whose main purpose is to ensure that usable web documents are created.

7.3.4 Usability components

In the "early" years of website design, efforts had been focused on hacking HTML as the main requirement for creating a website, and user interface design is often an afterthought. Nielsen (1996) predicts that web-surfing is dead, with only a few websites visited repeatedly by a substantial number of end-users. Owing to a change in relationship to web design, there is an increasing need to treat end-users as individuals rather than a nestful of hungry GET-request end-users. Usable web pages that subscribe to end-users' needs should be developed. However, this is not a simple task.

Designers need authoring aids to help them understand end-users' needs and browsing behaviour. Berners-Lee (1995) suggests carrying out testing on the web documents produced even though testing takes time. However, the decision of how much testing designers do depends on the quality of the web documents designers wish to provide. Presently, designers are not carrying out sufficient testing on the web documents created. This may be due to the lack of tool support making testing cumbersome (as discussed previously in chapter 6), demanding too much time and effort from designers. Hence, apart from the basic editing facilities of create, edit and save, embodied within HyperAT is an experimental, authoring testbed which allows hypertext designers to carry out different modes of usability testing on

the web documents created by HyperAT, all within the authoring environment of HyperAT (see figure 7.14): (1) *structural analysis* which analyses formally the structure of the web documents; and (2) *real user evaluation* which analyses end-users' browsing behaviour based on real users' transaction logs. The ability to toggle between different modes makes testing less cumbersome, and hence more convenient for designers, thereby increasing the chance of creating more usable web documents. The first and second modes of usability testing have been implemented in HyperAT. This thesis recommends exploring the potential of using *executable user modelling* or non-human user testing as the third mode of usability testing as work for future research (see chapter 9).

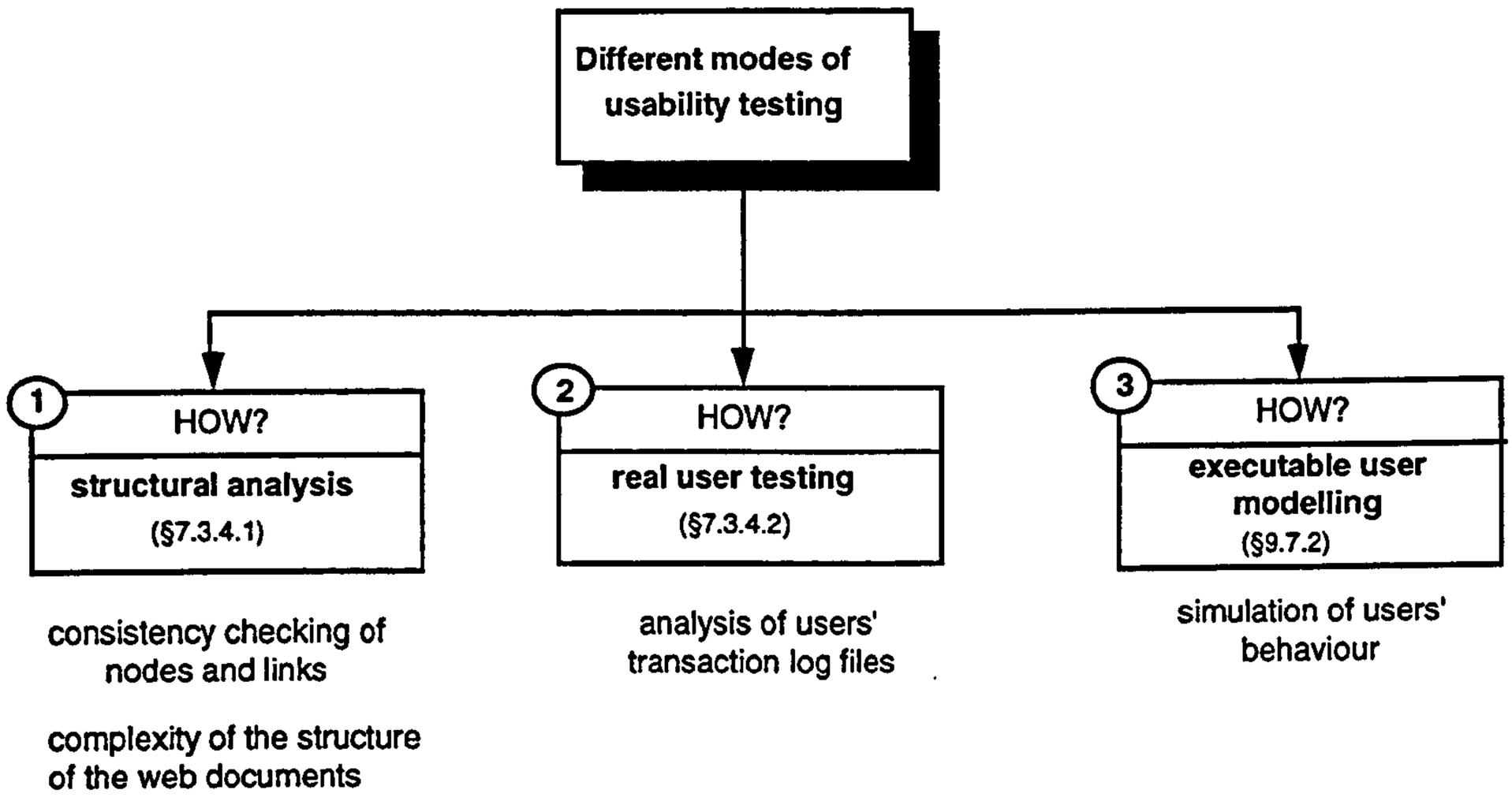


Figure 7.14. Three modes of usability testing within HyperAT's authoring environment

7.3.4.1 Structural analysis

In HyperAT, not only should designers be helped to structure information, this thesis also wants to help designers to avoid structural inconsistencies and mistakes. By treating end-users like computers, a formal way of analysing the structure of the web documents is implemented. HyperAT allows designers to analyse the structure by, firstly performing integrity checks on the nodes and links, and, secondly measuring the complexity of the structure of web documents. This first-cut evaluation alerts designers to take corrective measures as early as possible in the design process before it is too late.

- **Check structural inconsistencies**

  Because of the way information about the hyperdocument is input and captured into HyperAT, missing nodes and links can be identified quite easily. Figure 7.15 shows the results of two analyses, one displayed in the Display Window on the left and the other in a Bug Report file on the right. The analysis on the left compares the number of nodes that are created with the number of cross-referenced links in a hyperdocument. The Bug Report on the right is generated after parsing a website of related hyperdocuments. What the report highlights is a list of .gif files not found in the server. This could be due to missing

or inconsistently-named files/nodes. In this particular case, it is the latter. This form of analysis brings to designers' attention "silly" mistakes that can be easily rectified.

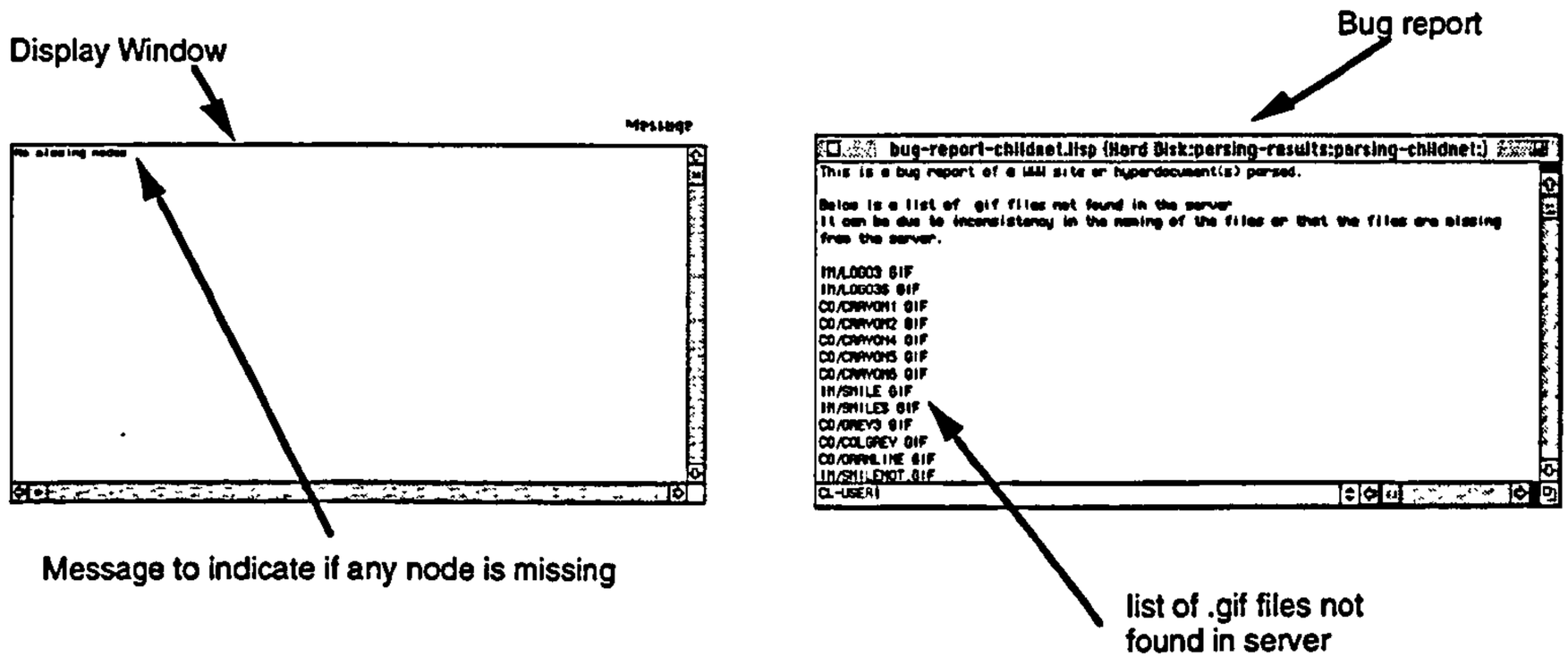


Figure 7.15. Information on missing or inconsistently-named files/nodes

- **Check structural complexity**

  The more complex the structure of the hyperdocument is, the more easily end-users may feel confused and "lost". With this in mind, simple metrics are implemented to measure the complexity of the structure of the hyperdocuments:

  - *No. of nodes*. HyperAT calculates the total number of nodes in the hyperdocuments, together with a listing of all the nodes present. If the number goes beyond a certain value, 10 000 nodes for example, then the structure may be too complex for some tasks. Designers may have to decide whether it would be wise to re-organise the nodes. Figure 7.16 shows the Display Window providing information about the number of nodes, and a listing of nodes in the hyperdocument.

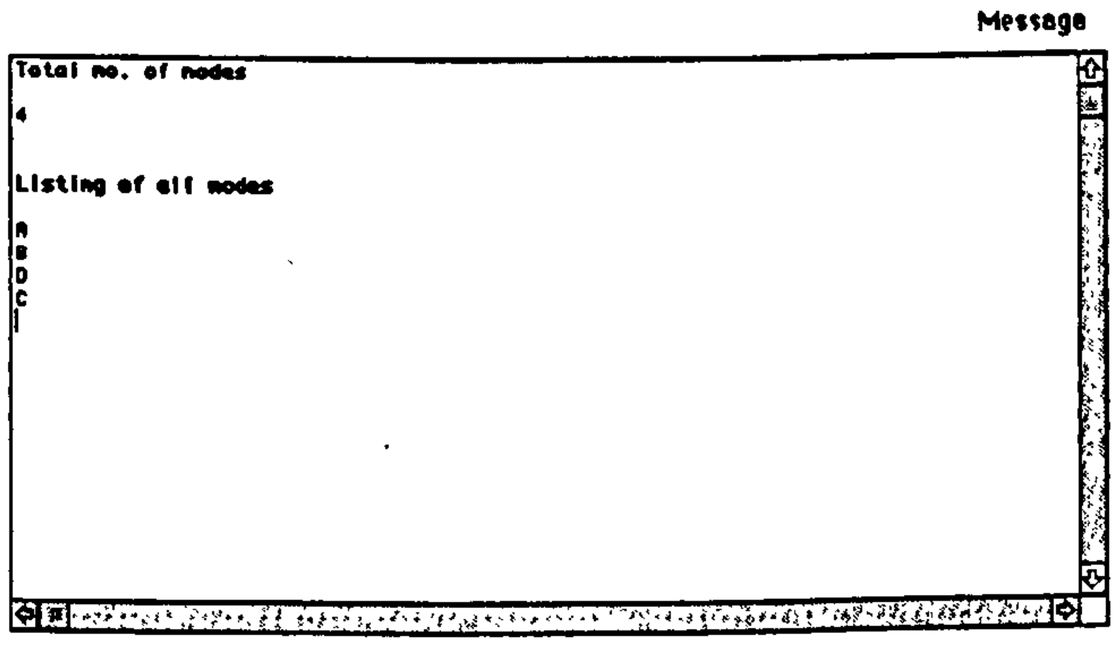


Figure 7.16. Information on number of nodes

  - *No. of links per node*. This metric indicates how "busy" the nodes are in terms of the number of links per node, both in-coming links and out-going links (also known as in-degree and out-degree respectively). If a node has too many links, then designers might infer that it contains too much information. Perhaps the design decision is to split it into simpler nodes, since good design principle suggests that nodes should be kept simple. Figure 7.17 shows a report providing information on the number of links

per node, using a hyperdocument with four nodes A, B, C and D as an example (see also figure 7.7 for the relationships among the nodes).

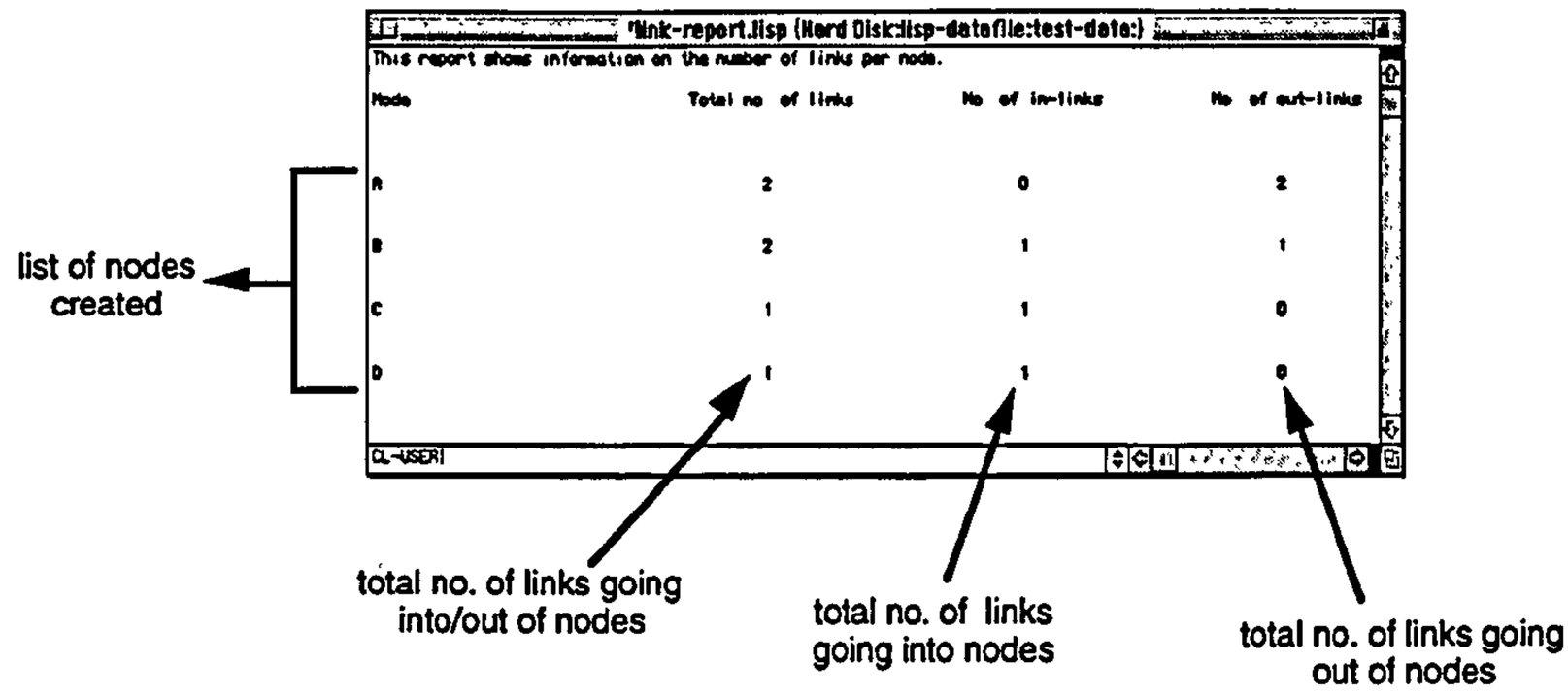


Figure 7.17. Information on no. of links per node

- *All possible paths from a given node.* Designers can query this information by selecting from a list of nodes. This information provides designers with all possible paths from a given node to all the leaf nodes in the hyperdocument (see figure 7.18). Designers can find out from this information the number of nodes that have to be taken to reach a leaf node. If the number is too high, it would imply that the structure is too complex. Designers can perhaps make use of this information to provide directed navigational help to end-users.

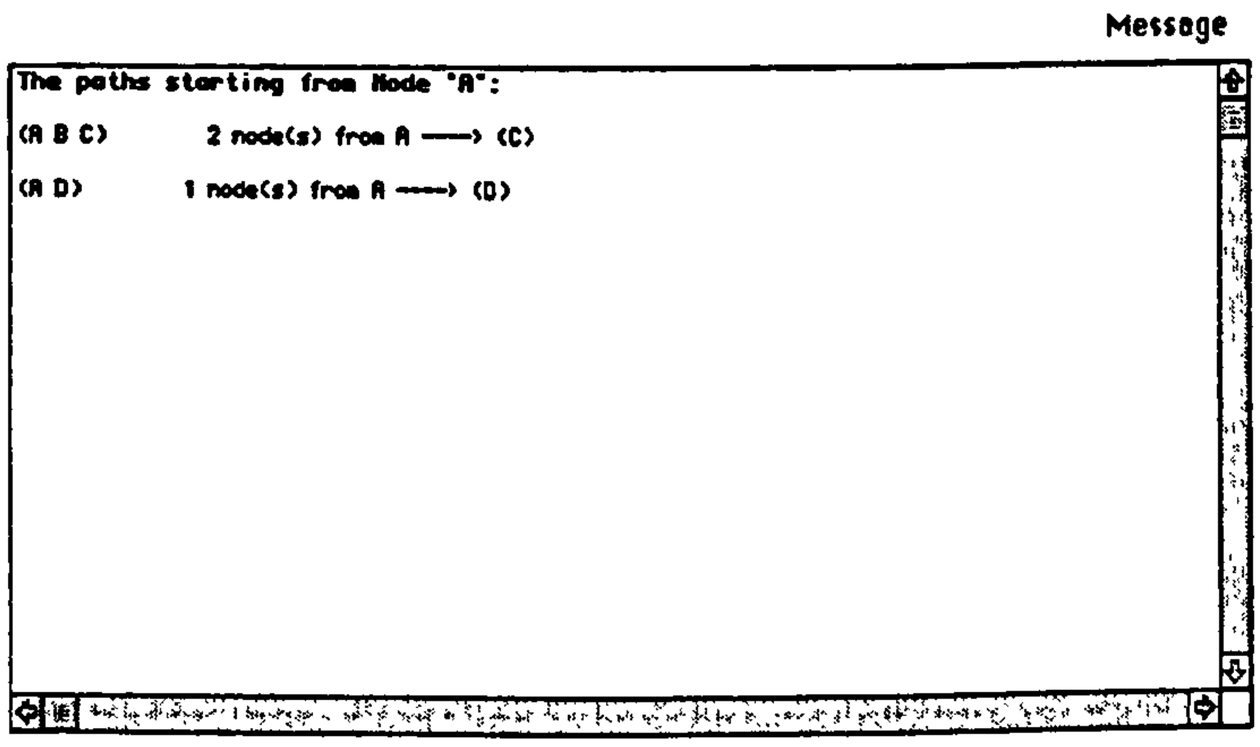


Figure 7.18. Information on all possible paths starting from node "A"

- *Depth of a structure.* This metric shows the number of levels away from the root node that is present in the hyperdocument. Accompanying this information is a global map of the structure. Clicking onto a node will open up another map which is a local map of that particular node (see figure 7.19).

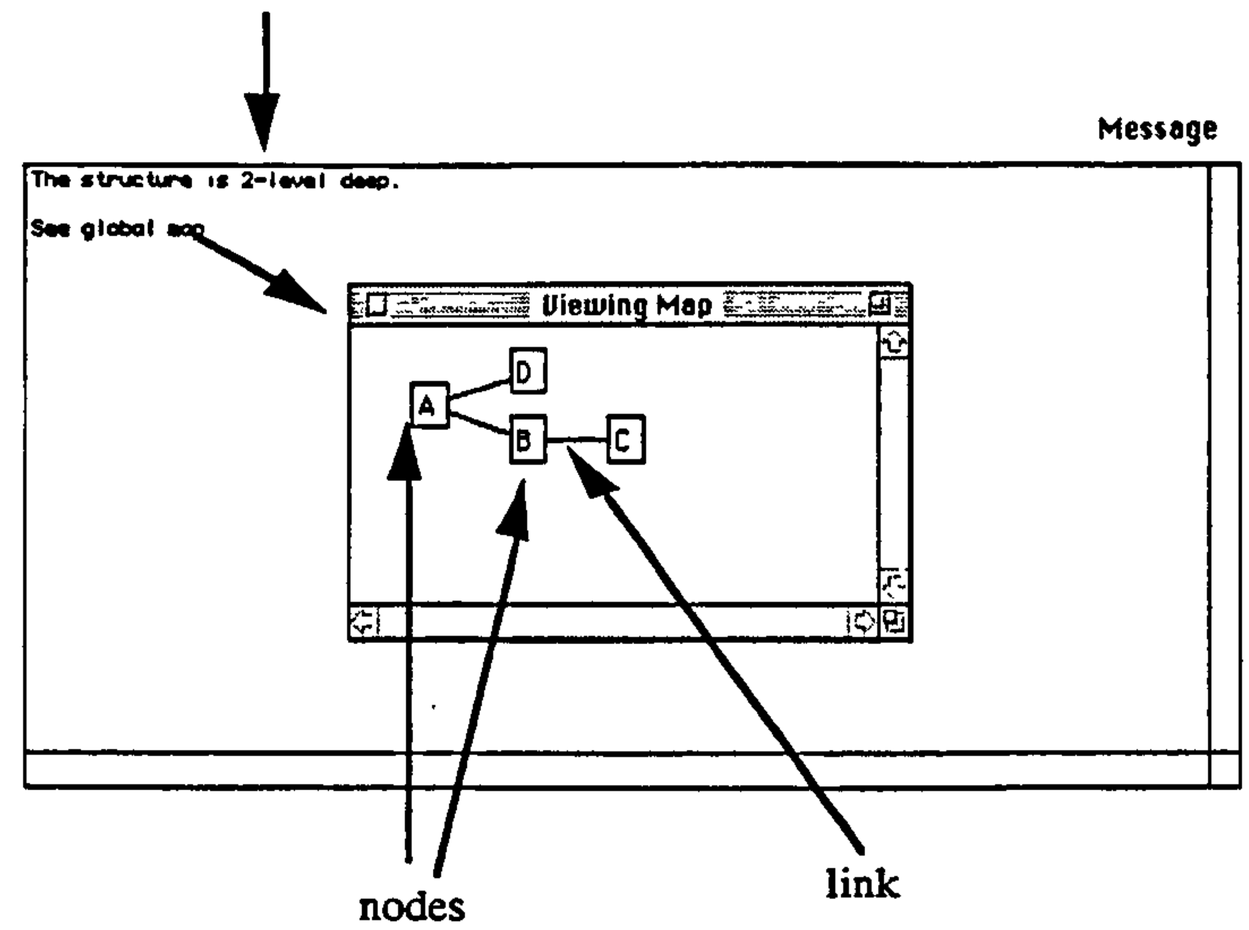


Figure 7.19. Information on structure of hyperdocument

- *No. of successors*. This metric isolates those nodes with less than three successors from those with three or more successors. Though the number three may be arbitrary and can be decided by designers for a specific domain, it reflects important design guideline to keep structure simple. Figure 7.20 shows the result of the analysis of a hyperdocument.

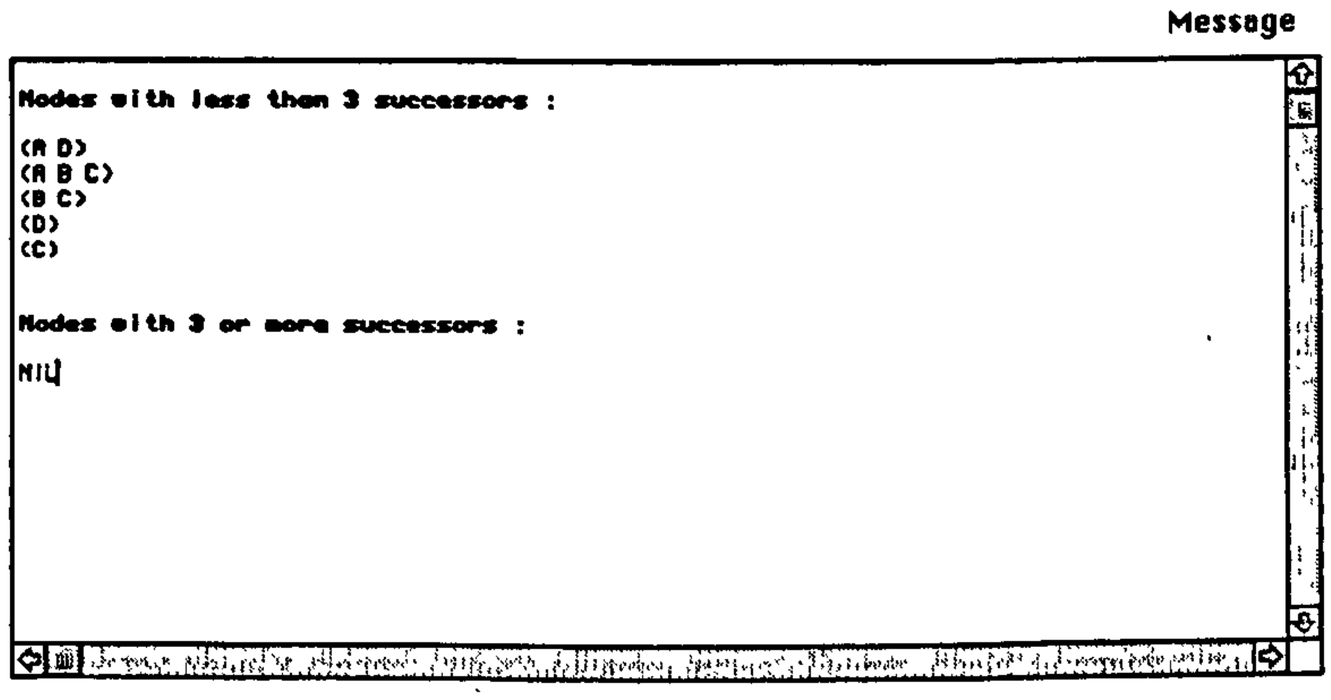


Figure 7.20. Information on number of successors of nodes

7.3.4.2 Real user evaluation

Real user evaluation is important because hyperdocuments are designed for end-users and not just what designers think or feel are important. In order to discern whether a certain web page is attracting its desired target audience, it is necessary to find out who is visiting the site. Understanding which pages receive the most hits can inform designers whether the intended messages are being viewed, and can offer some clues to either revise the structure or a site map to lure visitors to top-priority pages (Busch, 1997). Because the web servers log information about each transaction such as page accessed, date/time, visitors' profiles based on the URL from which the visitors accessed the page to the kind of browsers used, errors encountered, *etc.*, server log files are a source of very useful information.

By analysing the transactions that have been logged by server log files, real users can be effectively employed to evaluate web documents, a view also shared by Berners-Lee (1995) and Shneiderman (1997). Different metrics can be used to measure end-user activities such as counting the number of page hits, analysing the time end-users spent looking at a page, *etc.* Various factors, however, can affect the reliability of the metrics. For example, differences in web browsers can determine the way end-users perceive content and navigational alternatives or poorly designed structure and content of the documents may inhibit end-users from finding what they are looking for. Therefore, metrics chosen to measure end-user activities should be carefully considered, otherwise they may provide designers with an inaccurate estimate of end-user interest and motivation. Berners-Lee (1995) also cautions that analysing server log files takes time, if designers have to do that manually.

When work began in end 1996 on HyperAT, there was only a slight interest in site management tools. Over this past year, from the end of 1996 to the time of writing up this thesis end of 1997, there has been a growing increase in the number of site management tools flooding the market. Developers of these tools hope to cash in on the growing demand for good website analysis software to transform the raw transaction log data into meaningful information. Busch (1997) tested seven website analysis tools such as Bazaar Analyzer Pro 1.0, NetTracker 2.0, WebTrends 3.0, Market Focus 3, MKStats 2.2 and Net.Analysis Desktop 2.0, NetIntellect 2.1, and recommended WebTrends 3.0 to have the best combination of features at a reasonably low cost. While these tools share certain features, for example, the ability to process log files from a range of servers, analysing and converting these data into tables, charts and graphs, they differ dramatically in how they are used and in their ability to analyse files. Some of these tools are intended for Unix servers only and others are available for multiple platforms. To select the most appropriate tool, apart from the cost, the most important criteria is to find out whether the log file analyser is capable of handling a reasonably-sized log files (over 50Mb), whether it provides the kinds of reports needed, and the ease of use of the tool.

Yet, most of these website analysis tools are often sold as "accessories", and they are not integrated within the authoring environment. Perhaps, because the developers of authoring tools and website management tools are different, it is difficult to co-ordinate efforts to incorporate the authoring and analysis features into an integrated authoring environment. This results in a range of authoring and site management tools designed for multiple platforms being developed. It may perhaps be good to separate authoring features from analysis features, so that users of these tools would need only to buy the necessary "accessories" if required. But as needs get more sophisticated with the growing awareness of the importance to analyse end-users' behaviour, it is crucial that tools should provide an environment to handle both the authoring and analysis requirements.

This thesis, therefore, wants to explore the possibility of incorporating the analysis of server log files within HyperAT's authoring environment, and to demonstrate that this may be a viable solution. A couple of metrics have been implemented to illustrate how simple this can be done. Extending HyperAT for more metrics is not difficult. However, it is not within the scope of this thesis to be concerned with the validity of specific results.

Before discussing how the metrics are implemented in HyperAT, this thesis will discuss briefly the format of server log file since this will have consequences on the results of the analysis. Formats of server log files differ from server to server depending on how they have been configured, and on what platform they are running. In fact, according to Busch (1997), there are over thirty different formats for storing log files with most servers adhering to the NCSA/CERN-compatible common log format. For the purpose of illustration, a transaction log file of the Microsoft Internet Information Server was chosen since this was the only kind of server found in the School of Computing Science at the time when the development of HyperAT was carried out. The transaction log file from this server has been configured to display information such as date and time of the transaction, IP address of server, IP address of client machine and the web pages visited as shown in figure 7.21. This version of server does not, however, provide analysis of the log files to give useful information on document trails, referrers, *etc.*, which can be done fairly easily with more recent servers such as Netscape Enterprise Server 3.

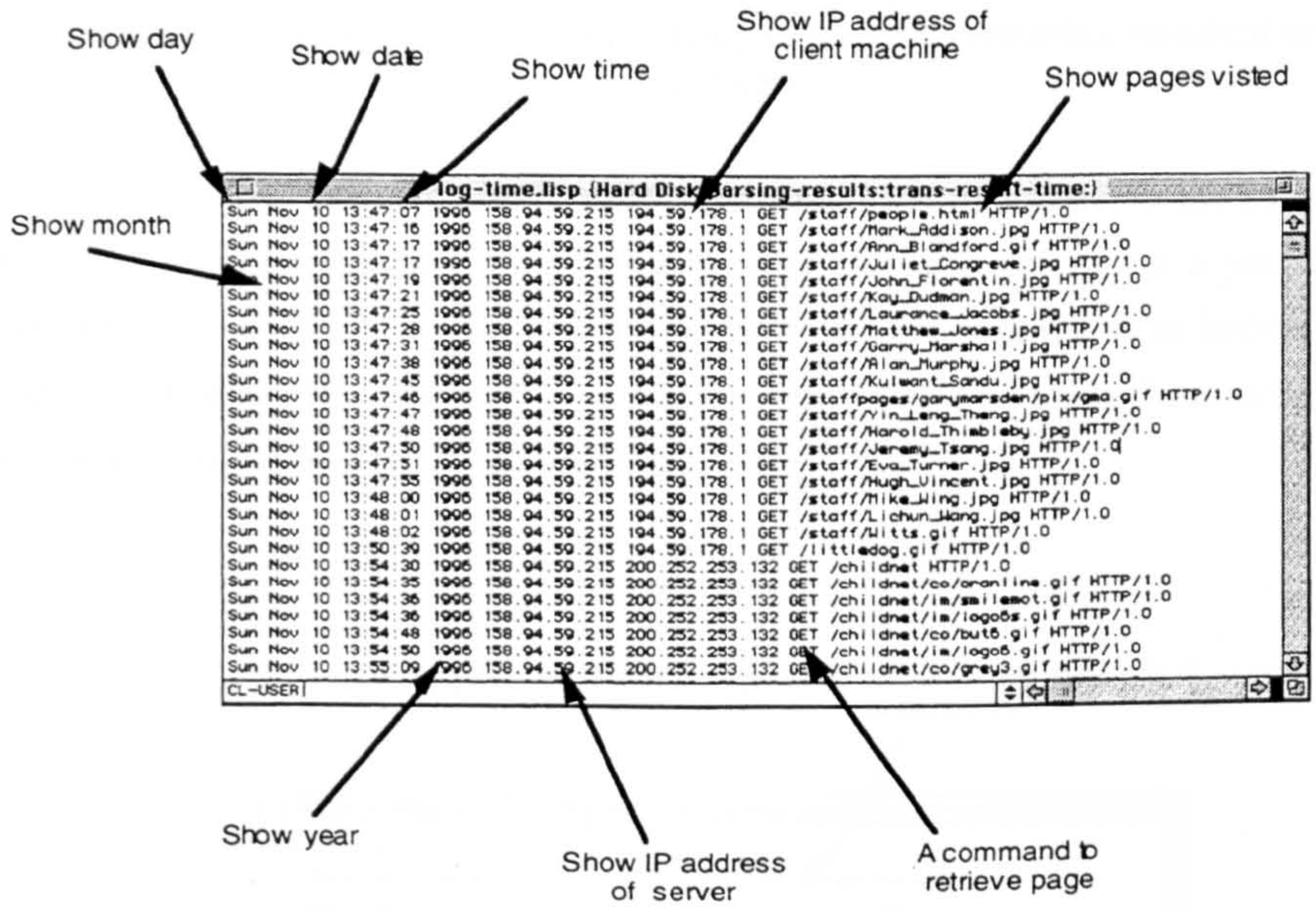


Figure 7.21. A sample of a transaction log file

HyperAT has been programmed to read this particular format of a transaction log file. Meaningful information that can be derived from analysing the log files includes frequency of visits, most popular pages, visitor demographics, most popular files downloaded, missing files, *etc.* Since most website management tools are capable of providing these data, this thesis will neither elaborate on them nor implement them fully in HyperAT. Instead, as an exploratory, research tool, HyperAT will concentrate on one aspect of helping designers to better understand end-user interest and motivations in terms of their browsing paths and goals. This is an area which has not been explored in website management tools. The following are three reports produced by HyperAT to achieve this objective:

- **Analyse end-users' browsing path**
  - *Frequency of visits.* This metric provides designers with information on the most visited pages to spur them to think of reasons why certain pages are more frequently visited than others. Could it be a case of design flaws? The whole idea about giving

designers this kind of information is to bring to their attention and set them thinking about their designs. Figure 7.22 shows the report on the frequency of visits to websites resident on a web server.

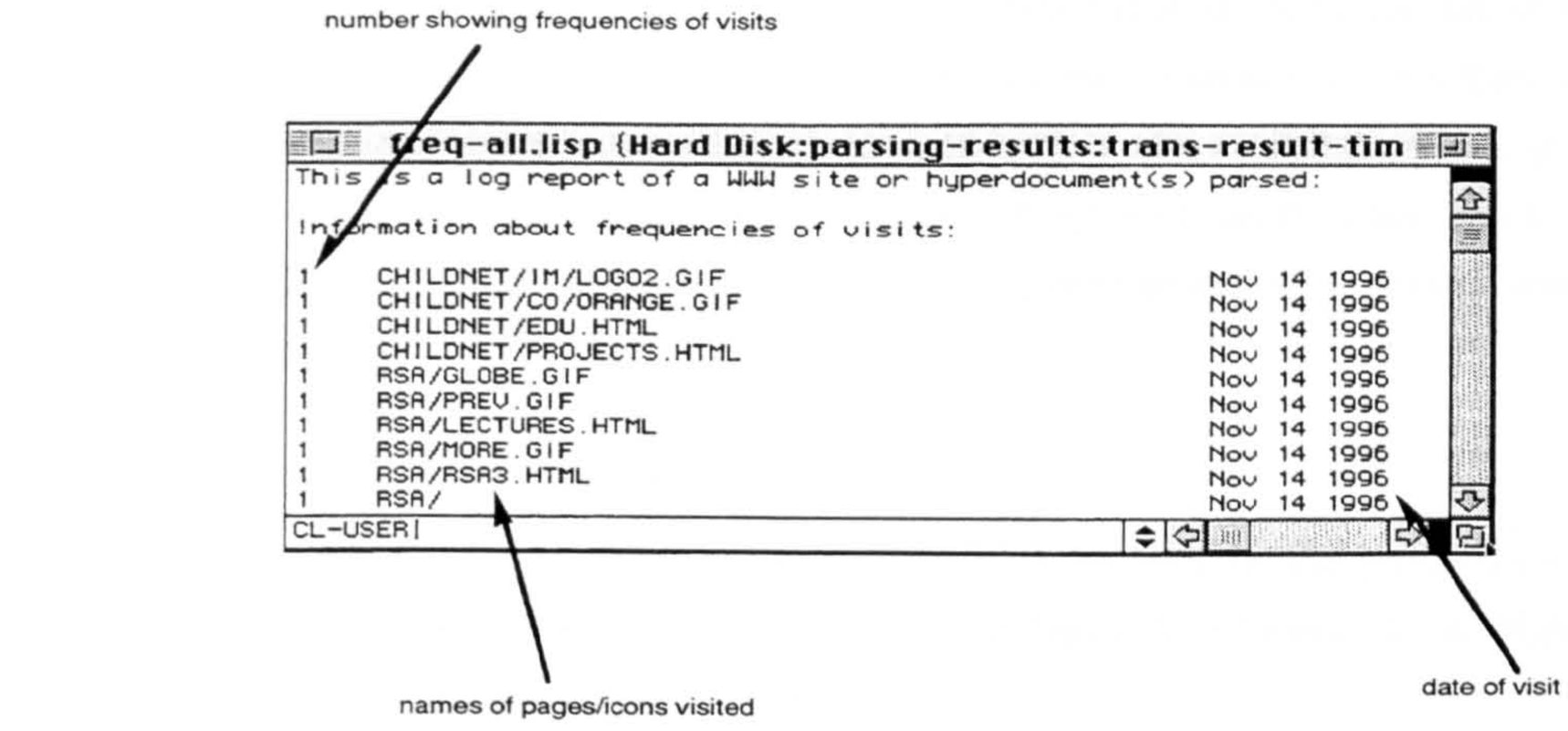


Figure 7.22. Report showing the frequency of visits to websites resident on a web server

- *Clients' browsing information*. This report provides designers with browsing paths of all clients who visited the websites held in the web server for a particular period (see figure 7.23). This means that designers can get real users to browse a website, and then analyse their browsing patterns, which will provide them with some clues on the structural design of the website.

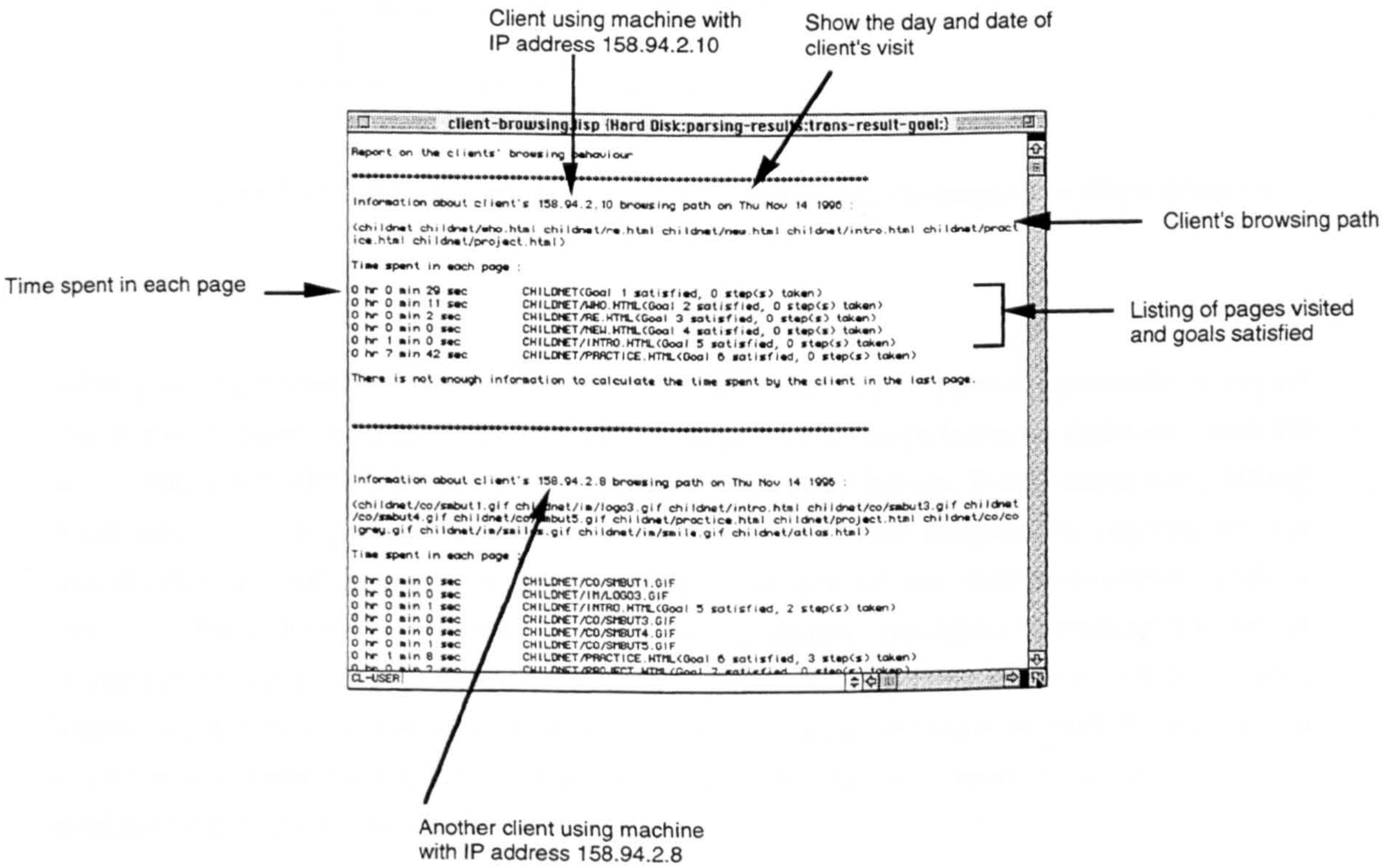


Figure 7.23. Report showing clients' browsing information

In this report, not only is HyperAT able to capture the pages visited, it also provides information on the goals satisfied. The methodology described here and its interpretation of the analysis may be debatable depending on the learning and cognitive theories used. It is, however, not the intention of this thesis to investigate that. What HyperAT wishes to do is to demonstrate that by parsing and analysing server log files containing otherwise untapped end-users' data, useful reports can be generated to analyse end-users' goals, pinpoint usability problems and guide design decisions.

- **Analyse end-users' goals**

  HyperAT provides an option to allow designers to specify the goals they would like to investigate and the respective number of steps taken to achieve them. Figure 7.24 shows the window to capture designers' inputs.

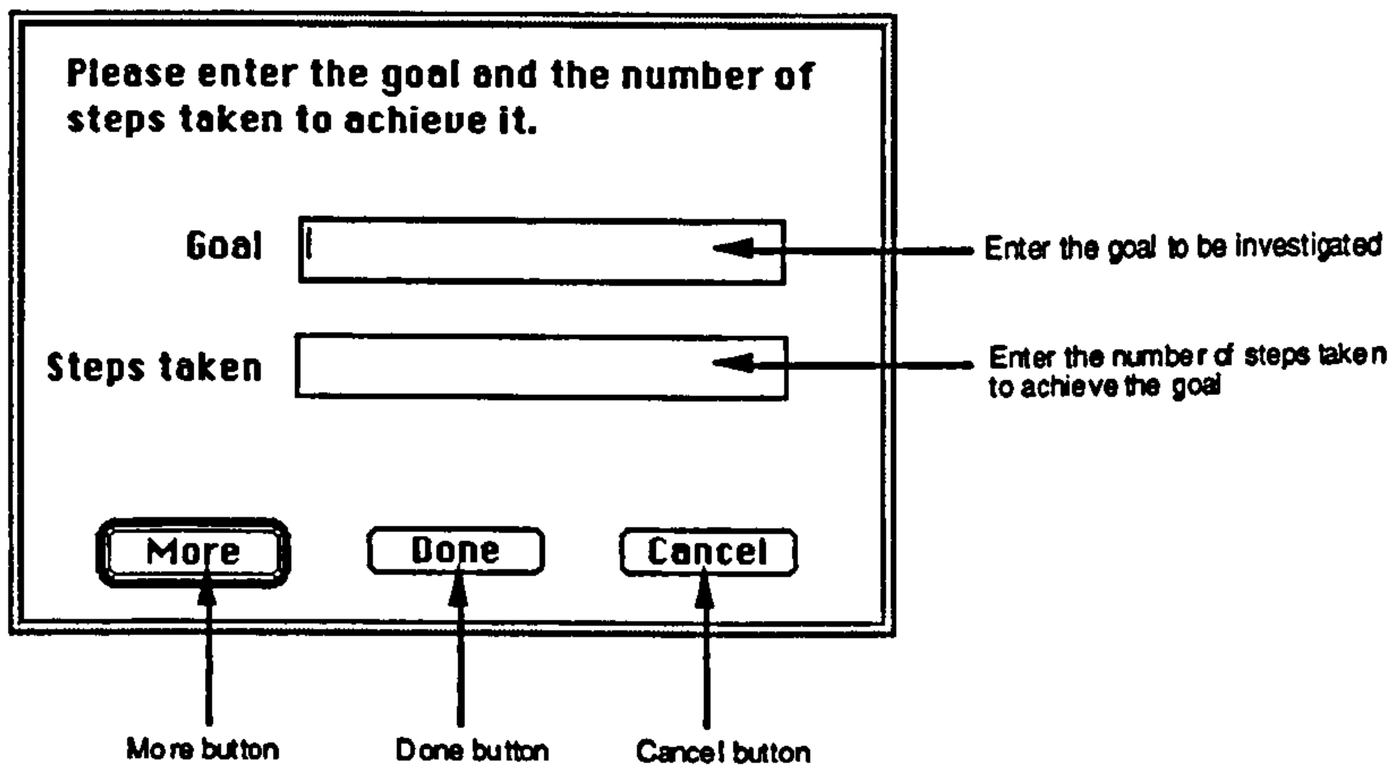


Figure 7.24. Input screen to capture the goal(s) and the respective steps taken to achieve them.

After parsing transaction log files, say for a particular day, HyperAT generates a report like the one shown in figure 7.25. This report compares the steps taken by end-users and the actual steps that should be taken according to designers' inputs. If end-users were taking more steps than expected, then designers might want to investigate the reasons for the discrepancies, and perhaps re-examine the structure of the web document, and/or interview the end-users concerned. The underlying assumption taken in reading this report is that performance of end-users is measured in terms of the number of steps or mouse clicks logged by the transaction files, rather than the actual time taken so that differences in thinking and learning time do not confound results. Different metrics could be used to evaluate different tasks, *etc.*

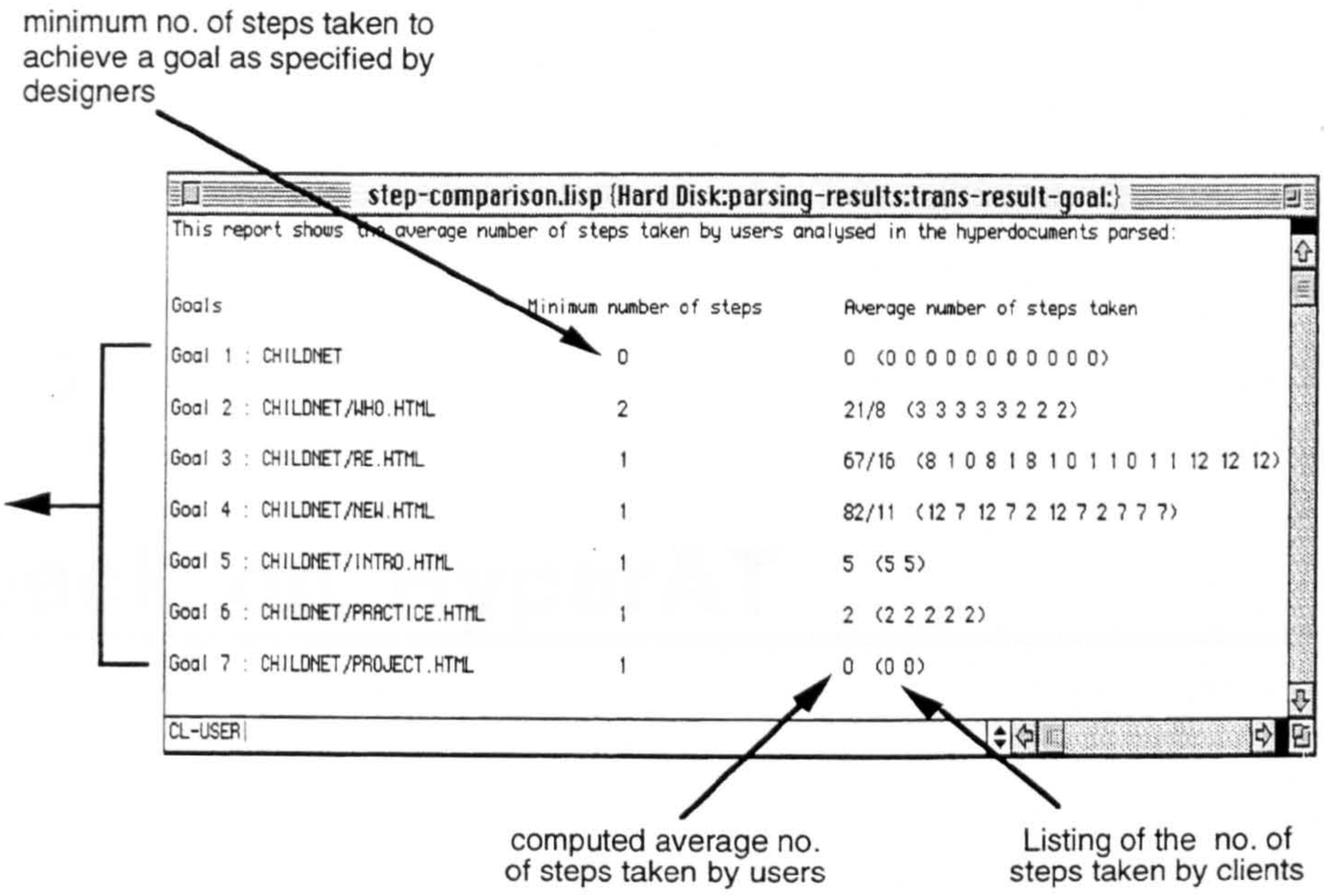


Figure 7.25. Report comparing steps taken by end-users and actual steps that should be taken

7.4 Conclusion

The chapter started off explaining what the web is, and reviewed solutions to address the LIH problem on the web. Solutions involve either looking at ways to improve the performance of the web or searching for alternatives to the web. A solution in the form of a web authoring tool called HyperAT was proposed to help designers manage the design process without feeling "lost". HyperAT was described in terms of its overall architecture and how the underlying concepts were implemented in HyperAT. The approach taken in HyperAT is novel in that established hypertext and HCI elements were integrated and implemented to ensure proper structuring and presentation of web documents. Two modes of usability evaluation of the web documents within HyperAT were implemented: (i) formal, mathematical methods to analyse the structure of the hyperdocuments; and (ii) transaction log files analysed to understand end-users' goals and browsing paths.

Incorporated within HyperAT were also design aids which are intended to help designers make informed design decisions to keep the website simple. In addition, HyperAT also generates navigation aids which are intended to help end-users navigate hyperdocuments produced by HyperAT more effectively, thus ameliorating the LIH phenomenon. Based on the investigations into design principles and guidelines (chapter 3), task analysis and end-user modelling techniques (chapter 4), hypertext structures (chapter 5) and designer tools (chapter 6), this thesis believes that HyperAT has the potential to eliminate or ameliorate the LIH problem on the web.

In the next chapter, an informal evaluation on HyperAT will be carried out to obtain feedback on the usefulness and usability of HyperAT as a web authoring tool in addressing the LIH problem.

Chapter 8

Feedback on HyperAT

Chapter objectives:

This chapter gathers qualitative results and impressions on the usefulness and usability of HyperAT as a web authoring tool. The activities carried out in this chapter include:

- using HyperAT to re-construct an existing web document
- comparing the web document created by HyperAT with a similar web document (in content) produced by a standard HTML editor
- getting feedback from hypertext designers
- comparing HyperAT with some web authoring and management tools

8.1 Introduction

In chapter 7, the web was selected as a hypertext example to demonstrate how the ideas generated from the investigations into various disciplines (as described in chapters 3 – 6) to address the LIH problem on the web were implemented into a practical web authoring tool called HyperAT. A general architecture of HyperAT was described, explaining how the synthesis of ideas drawn from human-computer interaction, hypertext, cognitive psychology and software engineering were used as underlying concepts in the design of HyperAT. Built into HyperAT are various forms of facilities and authoring aids to help hypertext designers manage the complexity of the design process without themselves getting "lost". In addition, established design and usability elements were integrated and implemented into HyperAT to ensure that usable web documents are produced, thus addressing the LIH problem. The conception of HyperAT is to promote greater research and discussion surrounding the LIH problem.

This chapter is concerned with getting qualitative results and impressions on the usefulness and usability of HyperAT as a web authoring tool. It is worthwhile to emphasise at this stage that since HyperAT is a research tool, the evaluation of HyperAT is therefore not concerned with whether it provides a full range of editing facilities with attractive interface, but whether HyperAT incorporates and implements novel and interesting ideas for future hypertext authoring tools to eliminate or ameliorate the problem of designers getting "lost" in the design process. Activities were selectively designed to achieve this objective.

§8.2 begins with describing HyperAT in use. However, this chapter will not be describing all the features implemented in HyperAT. Instead this section only focuses on the features that are involved in the creation, analysis and maintenance of a web document. §8.2.1 shows how to get started with HyperAT. §8.2.2 describes how an existing web document is re-constructed using HyperAT. §8.2.3 explains how analysis of the web document is carried out using HyperAT. §8.2.4 describes how the web document created by HyperAT is modified and maintained. §8.2.5 explains how to save the web document created and end a HyperAT session.

§8.3 discusses how an informal evaluation of HyperAT was carried out. §8.3.1 compares the web document created by HyperAT with a similar web document (in content) produced by a standard HTML editor. §8.3.2 obtains feedback from hypertext designers. §8.3.3 compares HyperAT with some web authoring and management tools.

8.2 HyperAT in use

8.2.1 Starting a HyperAT session

To start HyperAT, simply type in the Listener

>**(hyperat)** and press return.

Figure 8.1 shows a welcoming window to HyperAT. Click on "Click here to start" to invoke the main HyperAT window (see figure 8.2).

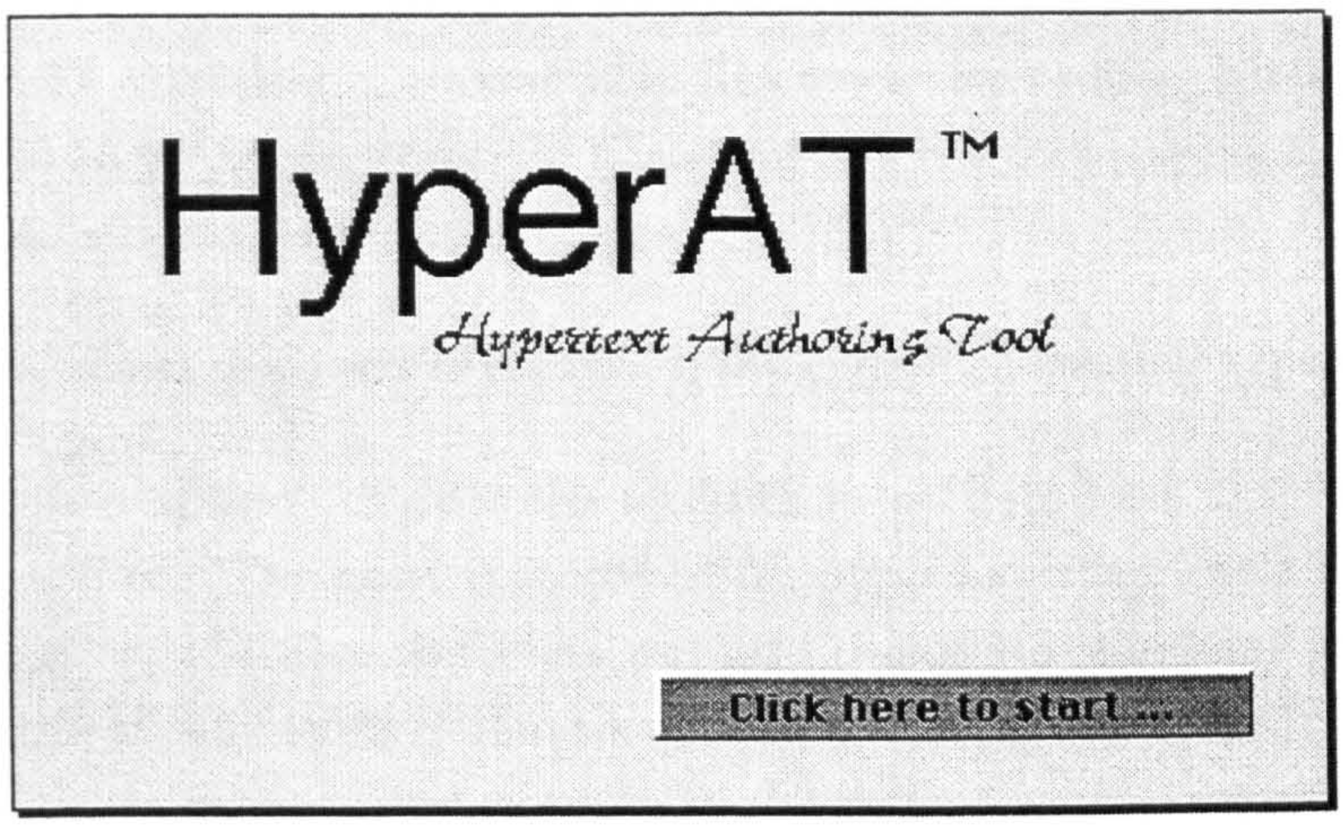


Figure 8.1. Opening screen of HyperAT

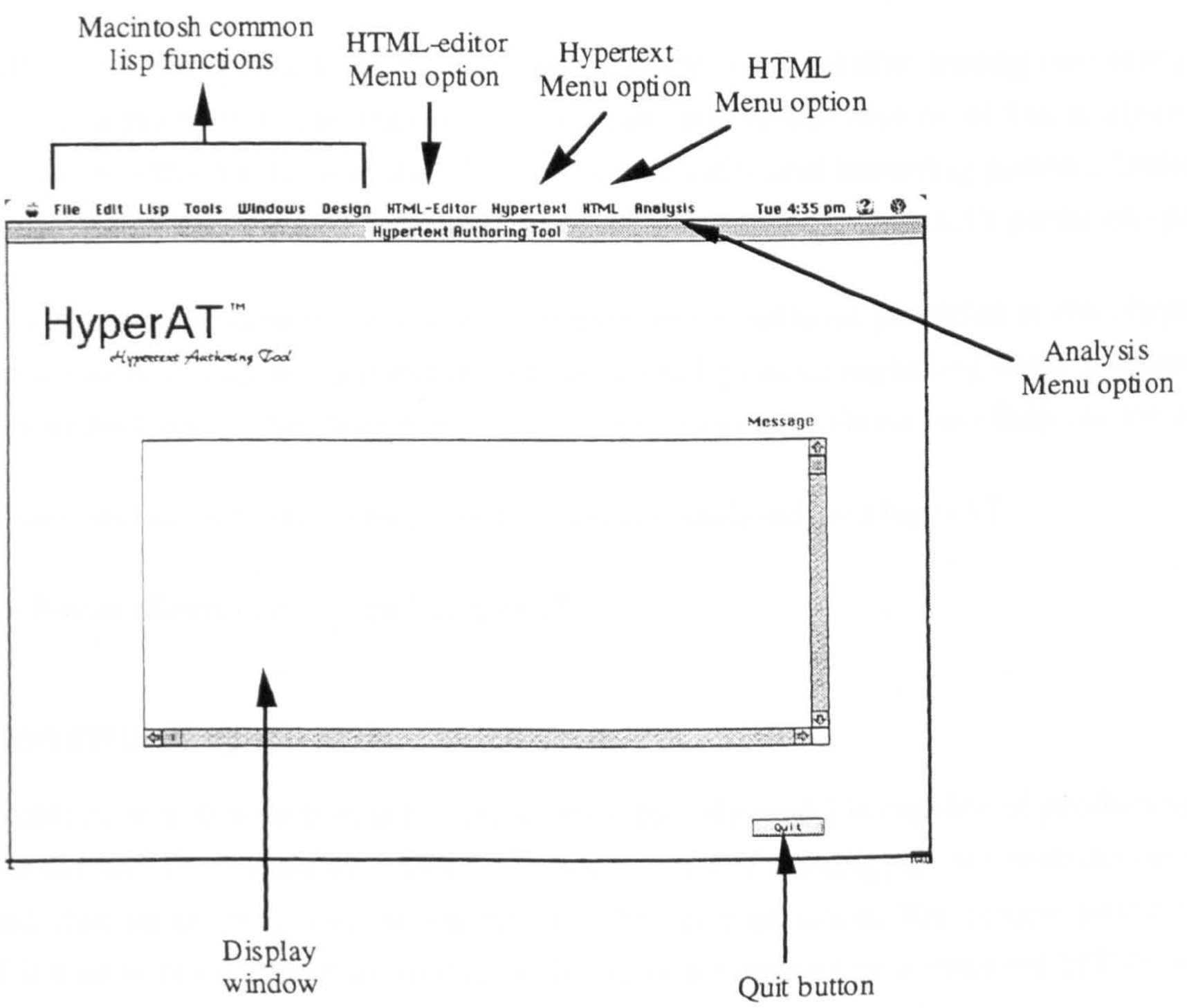


Figure 8.2. Main HyperAT window

HyperAT's facilities are accessed using a graphical, user-based interface. The main HyperAT window consists of:

- *Title bar* contains two parts. The first belongs to the Macintosh Common Lisp environment with the standard functions such as File, Edit, Lisp, Tools, Windows and Design (see the Macintosh Common Lisp Reference[1] for further details). The second belongs to the HyperAT environment. It has four different menu options: HTML-Editor, Hypertext, HTML and Analysis.

  HTML-Editor allows designers to call out HTML tags when creating hyperdocuments.

  Hypertext Menu option provides the facilities to perform basic operations such as create, save and edit. Designers can convert the hyperdocument into a predetermined HTML format. In addition, designers can also perform a statistical analysis of the hyperdocument to obtain information about its structure, pages and links.

  HTML Menu option allows designers to convert a HTML document into a structure recognisable by HyperAT. Using this structure, designers would be able to perform analysis of the hyperdocument which provides useful information about the pages and links, as well as be informed of discrepancies detected by HyperAT.

  Analysis Menu option allows designers to carry out usability testing involving real users. Designers can read transaction log files and obtain reports of the analysis that provide information on end-users' frequency of visits and browsing pattern. Designers can also query about a particular user's browsing behaviour, or about a particular page.

  As mentioned previously, this section focuses on the features provided in the Hypertext Menu option. A step-by-step guide on how to use HyperAT regarding these features will be described. As for the other menu options, appendix A explains how they can be used.

- *Display window* outputs message and/or results analysed by HyperAT.

- *Quit button* allows users to quit HyperAT.

8.2.2 Constructing a web document

The main objective of this section is to demonstrate that HyperAT is capable of producing web documents like any other standard HTML editors. Instead of creating just any web document, it is intended that an existing web document is to be re-constructed. The reason being that if HyperAT is able to re-construct an existing web document created by a standard HTML editor, this thesis would have shown that HyperAT possesses basic authoring features capable of creating web documents like any standard HTML editor.

---

[1]Produced by Apple Computer, Inc. for Macintosh Common Lisp (version 3.9).

For this purpose, the "Childnet International" website was chosen because it is reasonably small, and therefore discussion and analysis of it could be more complete. In addition, this website is maintained in the server at the School of Computing Science at Middlesex University, and therefore running analyses to compare both the original and created websites would be much easier and more convenient. (Middlesex University has no connection with the original designers of "Childnet International".)

- **About the "Childnet International" website**

  The "Childnet International" website is originally designed and built by AltasWWW, a consultancy company that specialises in web designs. This website promotes the interests of children in international communications, and protects them from any negative influences. It is organised under six different headings:

  - *Research*. This area discusses the organisation's research plans and developments.

  - *Positive projects*. This area describes some positive projects undertaken to link children across the world using the Internet.

  - *Promoting good practice in the industry*. This area encourages positive developments to adopt positive initiatives and prevents undesirable content being seen by children.

  - *Educating and informing*. This area highlights the need to network with child welfare and educational groups, governments and international agencies to provide information on how children can benefit from as well as be protected when using the Internet.

  - *Who we are*. This area introduces the organisations and the persons in charge.

  - *What's new*. This area gives up-to-date information on related events.

Figure 8.3 shows the structural representation of the website consisting of 13 nodes or web pages (as at September 1996), drawn using square boxes. "Ext" represents external web pages. To simplify the structure, not all the links are drawn. For example, links from web pages 3 – 10, 11 & 12 to other web pages 2 – 8 are not reflected in the diagram.

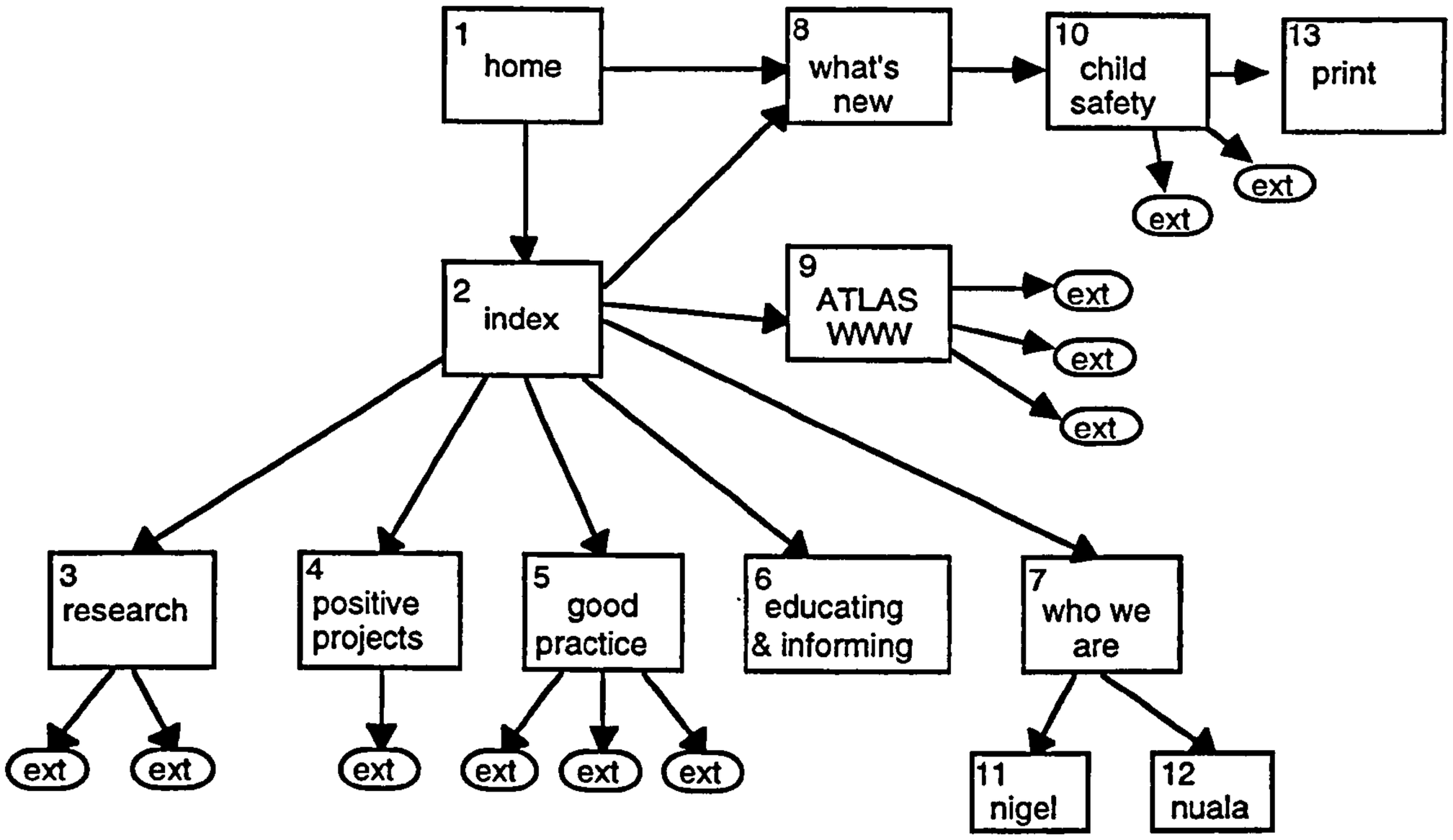


Figure 8.3. Simplified structure of "Childnet International" website

- **A sequence of screens for the creation of web pages**

  To begin, highlight and select **New** in the Hypertext Menu as shown in figure 8.4. To assist designers in writing the HTML codes, a HTML-editor is incorporated in HyperAT. This HTML-editor, written by Bill St Clair from Apple Computer, Inc. and adapted for HyperAT, provides designers with a menu to write HTML codes without having to memorise the syntax (see appendix A3). It is to be noted that at the start of a session, only the submenus **New** and **Load** are activated. A form-like screen (see figure 8.5, shown previously in figure 7.6) was then displayed into which information for the root node, that is, the Home Page, was entered.

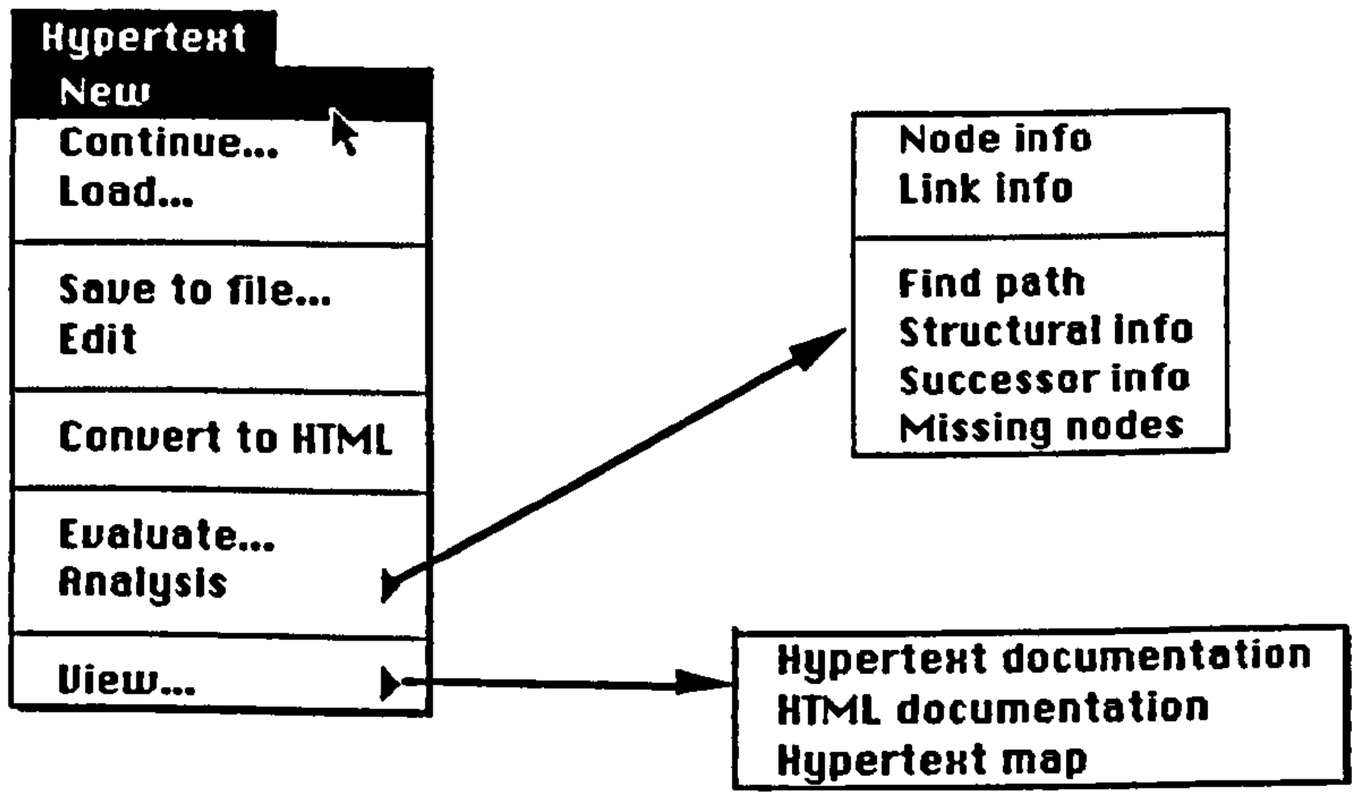


Figure 8.4. Hypertext Menu and associated submenus

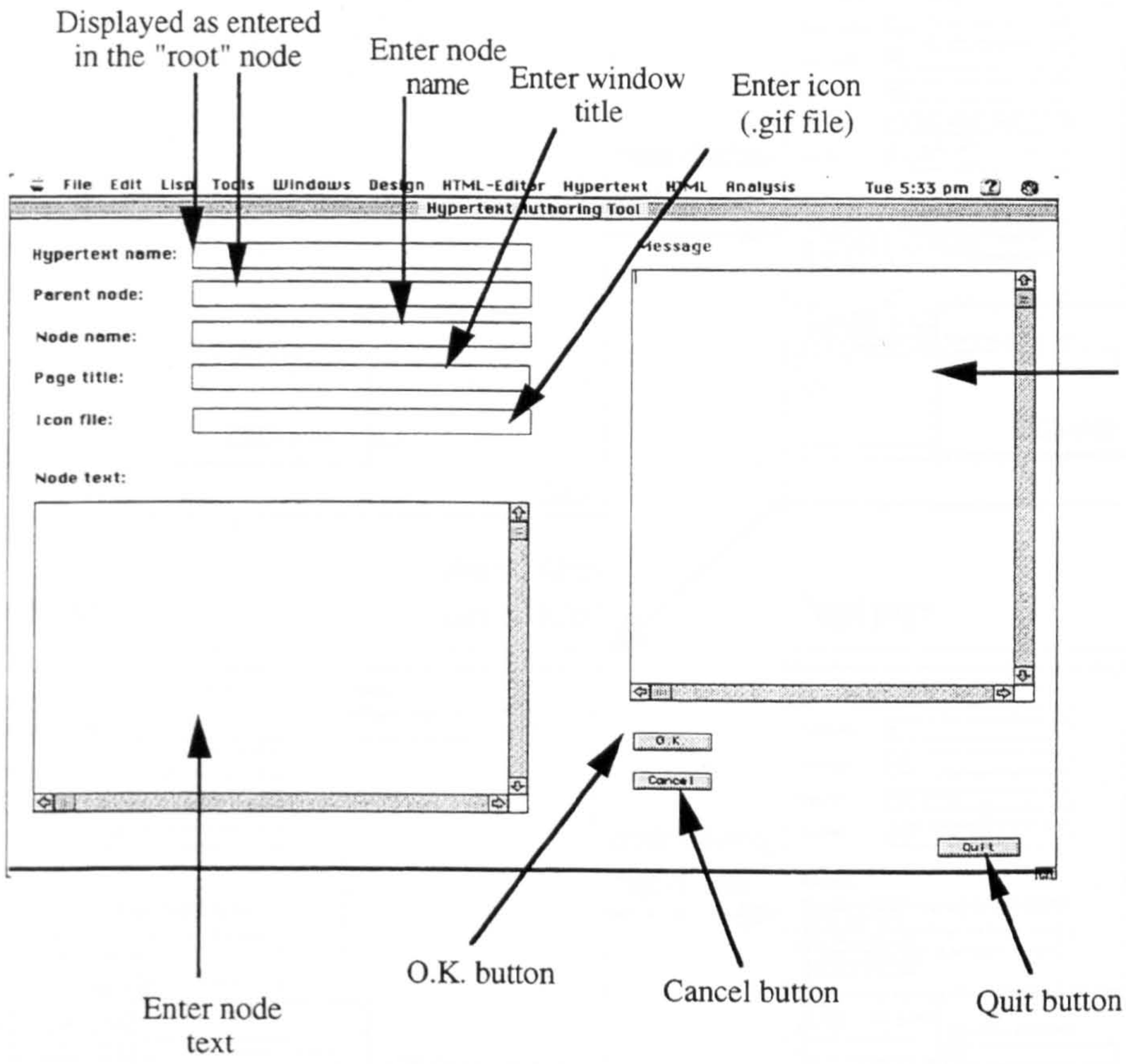


Figure 8.5. A form-like screen for node entry

For the purpose of illustration, figure 8.6 shows the sequence of screen inputs for the creation of only part of the website involving web pages 1, 2, 7, 8, 11 and 12. The home page was first created. When all the information had been entered, a window appeared to prompt if another page was to be created. Clicking **yes** would bring up another window to ask for the name of the parent of the new web page to be created. In this case, **home** and clicked **O.K.** This would open up a new screen to input data for "child-page" of **home**, which in this case is the **index** page. This created a parent-child relationship between **home** and **index**. The rest of the pages **who**, **nigel**, **nuala** and **new** were created in the similar manner by firstly indicating the parent page, and then entering the information for the children pages.

To save all the six web pages created, select **no** in the window for the creation of a new node (see last window in figure 8.6). HyperAT would then write the data structures of all the web pages created so far into a file (see appendix G). When this was complete, submenus **Evaluate** , **Convert to HTML** , **Save to file** and **Edit** were activated.

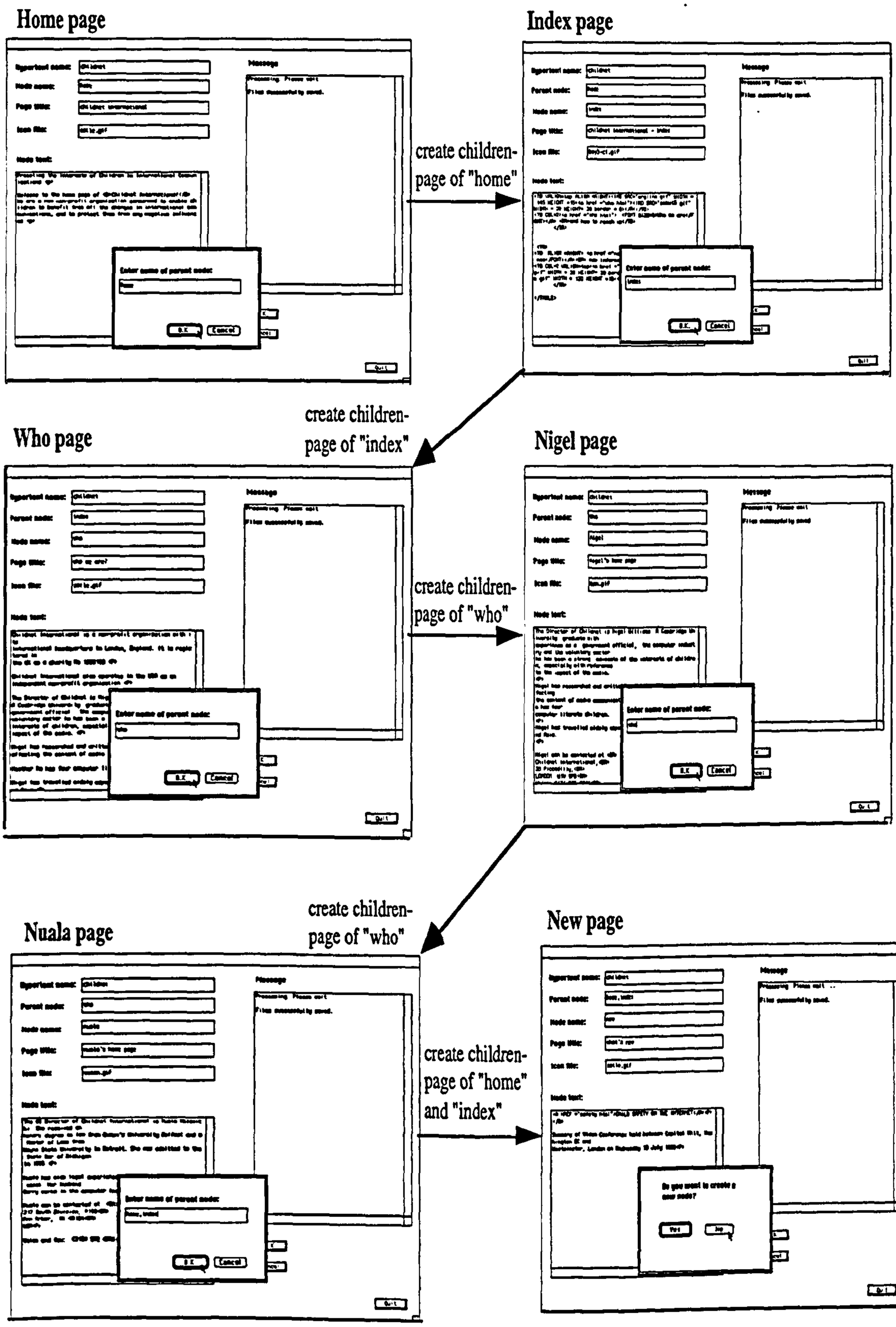


Figure 8.6. Sequence of screens to capture the for the creation of web pages 1, 2, 7, 8, 11 and 12

- **Generated map showing structure**

  A map showing the structure of the website can be obtained by selecting the submenu **Evaluate** before selecting **Hypertext map** under **View** submenu (see figure 8.4). Figure 8.7 shows the graph generated by HyperAT. The algorithm for generating the graph begins by drawing from a node and subsequently the children, grandchildren, *etc.* of that particular node. In this example, the graph was generated from the node **home** and the children nodes include **new**, **index**, **who**, **nigel** and **nuala**. Owing to the way in which the graph algorithm was implemented, the node **new** was duplicated. The algorithm could be modified to remove the duplicate (indicated by the cross) and a line could be drawn from **index** to **new** (indicated by a thick line). However, it would make it difficult for designers to determine whether **index** is the parent or **new** is the parent, since the direction of the link is not indicated, therefore making the graph ambiguous. It is debatable, but without even considering the technical implementation of a modified graph generator, it is advantageous to retain the duplicates since the number of duplicate nodes is also an indication of the complexity of a web document.

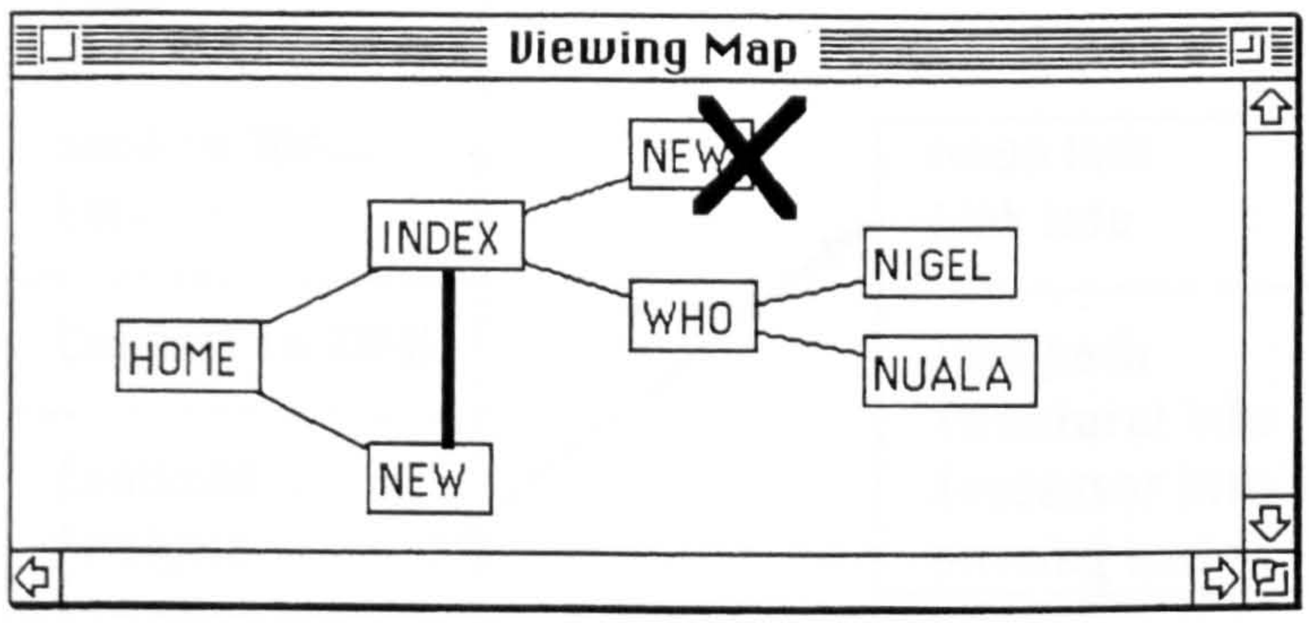


Figure 8.7. Map showing the structure of the six nodes

- **Convert to HTML format**

  To convert the six web pages into HTML format, the submenu **Convert to HTML** option under Hypertext Menu was chosen. A pop-up window (see figure 8.8) appeared which prompted the name of the creator of the hyperdocument to be entered. This name will be displayed at the bottom of the web page for the "table of contents". Time and date of creation will also be automatically generated and displayed. The **O.K.** button confirms the operation and the **Cancel** button abandons the operation. Appendix H shows the generated HTML files for the six web pages.

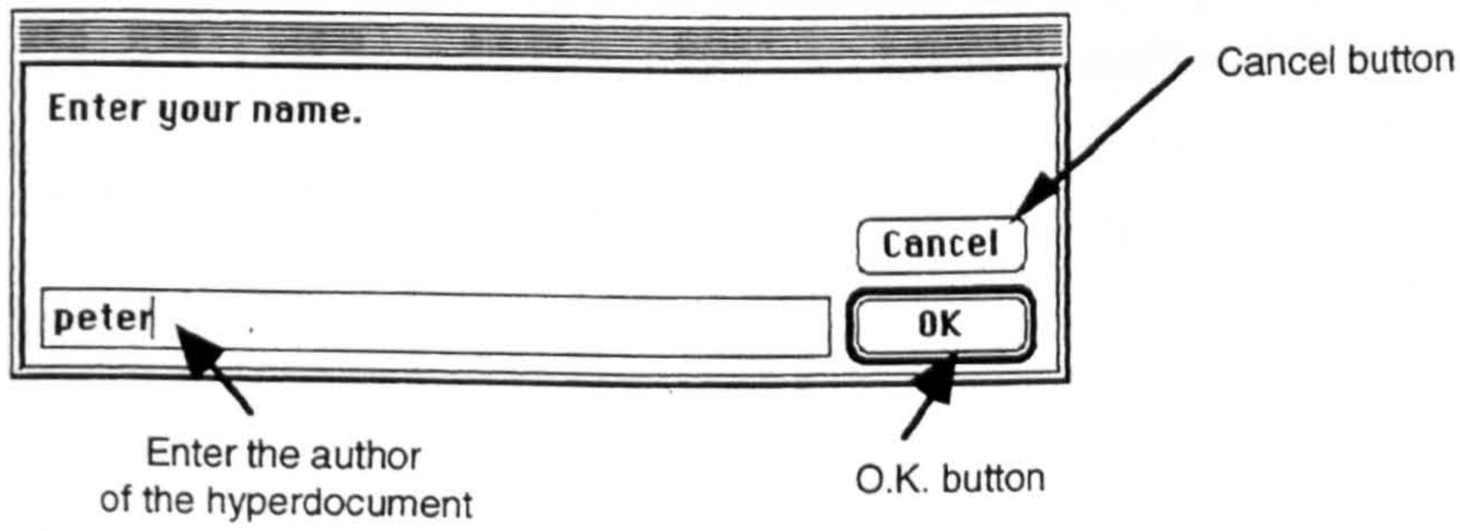


Figure 8.8. Window to capture the creator of the hyperdocument

8.2.3 Analysing a web document

HyperAT generates navigational aids to aid user navigation. In addition, as described previously in §7.3.4.1, embedded within HyperAT is a testbed which allows a *first-cut, formal evaluation* to be carried out early in the design process by running analysis on the entire "Childnet International" website to obtain information regarding: (i) number of nodes; (ii) number of links per node; (iii) structure of the website; (iv) nodes with 3 or more successors; and (v) missing nodes. The analysis can be invoked by selecting the **Evaluate** option in the Hypertext Menu (see figure 8.9). As an illustration, only (ii) - (iv) were selected to show how analyses by HyperAT could provide designers with feedback on complexity of the "Childnet International" website, thus allowing designers to make informed design decisions to make changes, if necessary.

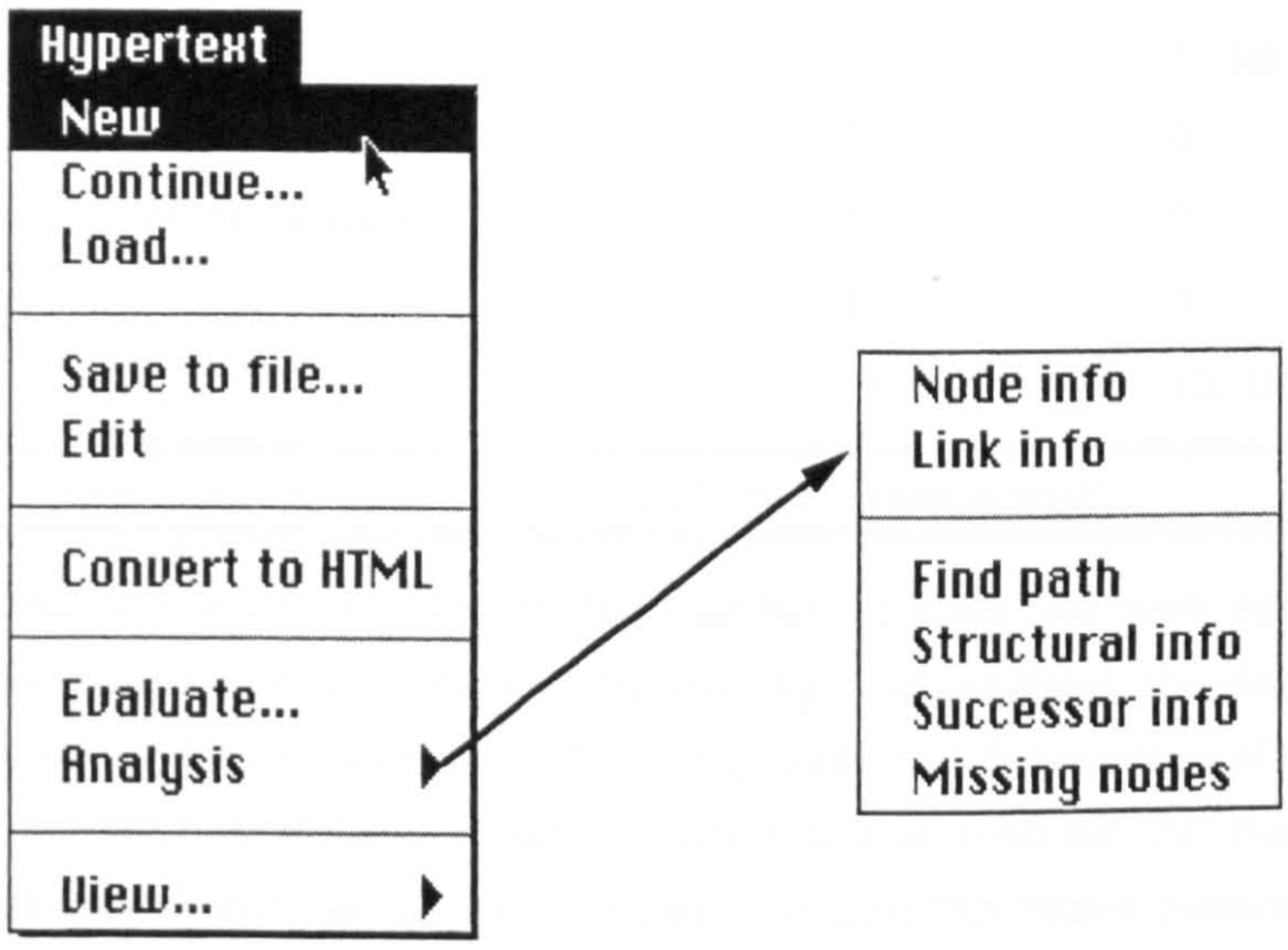


Figure 8.9. Hypertext Menu and the Analysis submenu

- **Number of links per web page**

  Table 8.1 tabulates the number of in-links and out-links manually calculated for each of the web pages of the "Childnet International" website (see figure 8.3), which is invoked by selecting the submenu **Link info** under **Analysis** menu. The calculation, however, does not include links provided by the navigational buttons. The column under in-links refers to the number of incoming links to a web page, and the column under out-links refers to the number of out-going links from a web page. These out-going links can either point to links within "Childnet International" or to links outside "Childnet International" which is denoted by (e) in table 8.1.

Table 8.1. Breakdown of number of links per web page in "Childnet International"

Web page	In-links	Out-links
1. home	0	2
2. index (table of contents)	1	7
3. research	1	2(e)
4. positive projects	1	1(e)
5. good practice	1	3(e)
6. educating & informing	1	0
7. who we are	1	2
8. what's new	2	2
9. AtlasWWW (creator of this site)	1	3(e)
10. child safety	1	1, 2(e)
11. nigel (contact person)	1	0
12. nuala (contact person)	1	0
13. print	1	0
total	**13**	**12, 11 (e)**

Key: (e) means link to pages not in "Childnet International"

Figure 8.10 shows the information on the number of links per web page generated by HyperAT. The number of in-links is computed by reading from the data structures the number of parent nodes of a web page. The computation of the number of out-links is more complicated. It involves writing a parsing algorithm to read off the number of ".html" links in a web page which points to web pages within "Childnet International" and the number of external links denoted by the prefix "http://www". The number of out-links is then computed by adding number of ".html" and "http://www" to the number of children nodes of that web page.

The number of links for the web pages as shown in figure 8.10 matched the manually calculated figures, except for the number of out-links for the **index** and **new** pages. The reason being that both **index** and **new** pages contain ".html" links which are also the children nodes. The algorithm could be modified in the future to check for duplicates. The benefits of an automatically generated table containing the number of in-links and out-links for each web page are obvious, since there is little effort on the part of the designers to get a rough estimation of the "busyness" of each web page compared to the time taken to calculate it manually. For example, the results reveal the **index** page is the "busiest", hence designers may want to make it less "busy" by splitting the page into two perhaps. The design decision to make changes or not is left to the discretion and experience of the designers. What HyperAT does by generating this table is to provide designers with data to make informed design decisions to make the "Childnet International" less complex, and hence more comprehensible to the end-users.

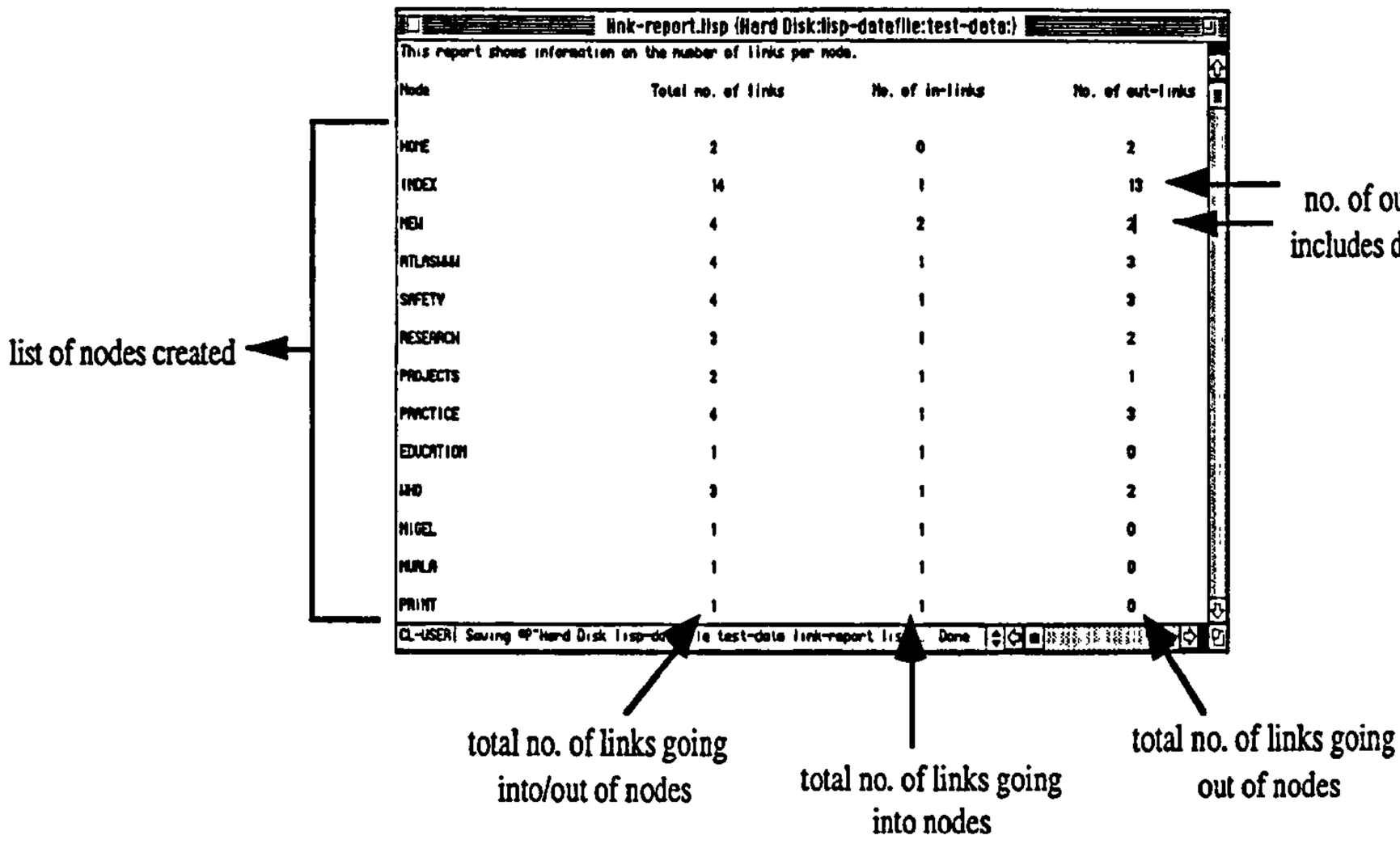


Figure 8.10. Information on number of links per web page

- **Structure of the website**

  Designers can get a feel of the complexity of the website by selecting the submenu **Structural info** option under the **Analysis** submenu to obtain information about the depth of the structure as well as the global map of the "Childnet International" website (see figure 8.9). The graph of this website is 4-level deep, with the "home" page defined at Level 0 (see figure 8.11). For a simple website like "Childnet International", the metric may not be significantly useful. However, most hyperdocuments are large and automatic generation of this metric by HyperAT would save designers much time and effort, if they had needed to calculate manually. The graphical representation of the structure presents to designers not only the relationships between the web pages, but also the complexity of the entire website. The graph generator incorporated in HyperAT also allows designers to obtain a local view of any web page, which is extremely useful when the website gets too big and complicated.

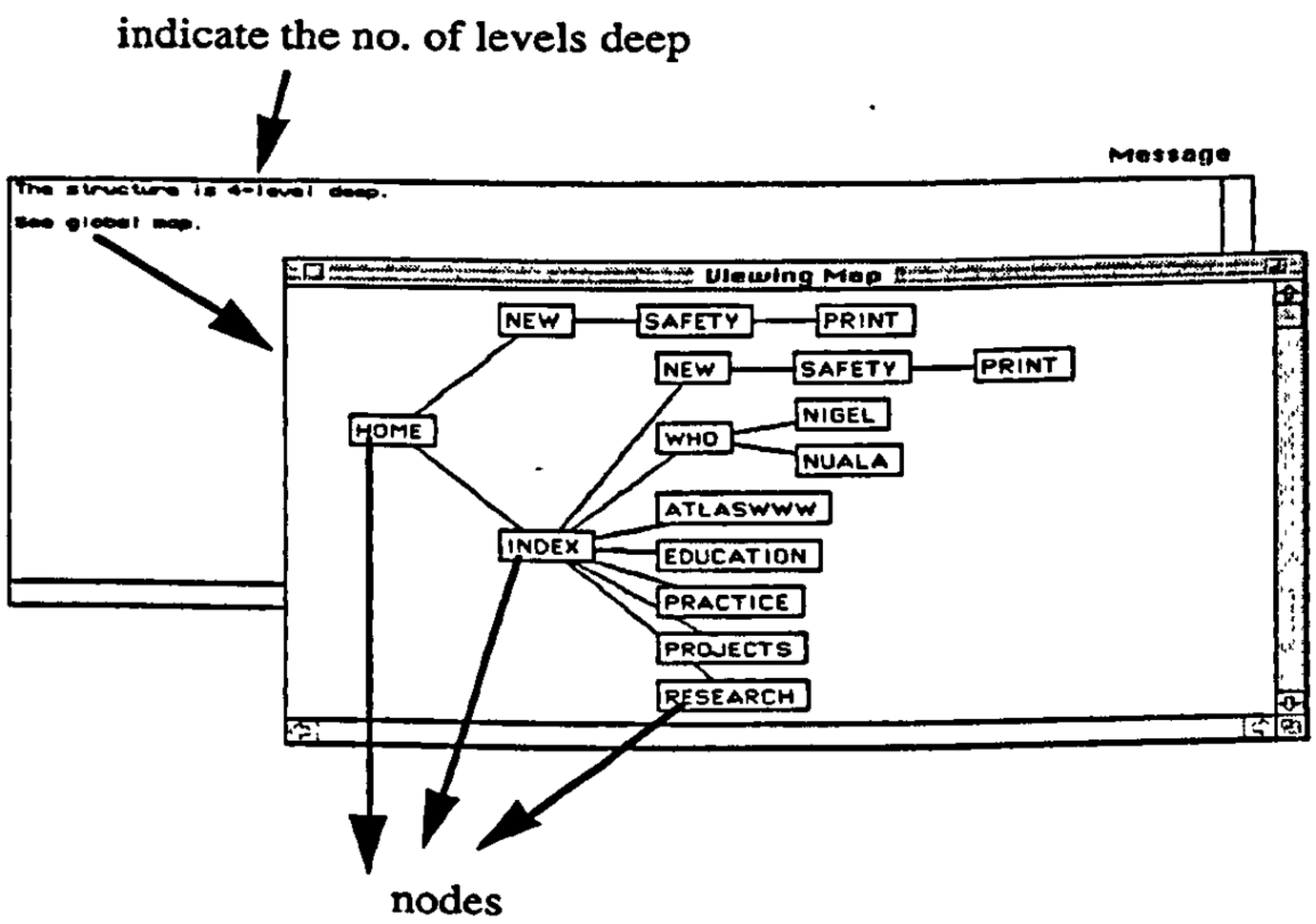


Figure 8.11. Information on the structure of the entire "Childnet International" website

- **Web pages with three or more successors**

  By selecting the **Successor info** option under the submenu **Analysis**, designers are provided with a list of web pages singled out by HyperAT with more than three successors. In this website, both **home** and **index** pages have more than three successors (see figure 8.12). Since important design principle suggests that structure should be kept simple with no more than three successors (Addison and Dudman, 1995), designers may want to re-organise the hyperdocuments to adhere to this principle. This would be a good design aid to remind designers to keep always keep structure simple.

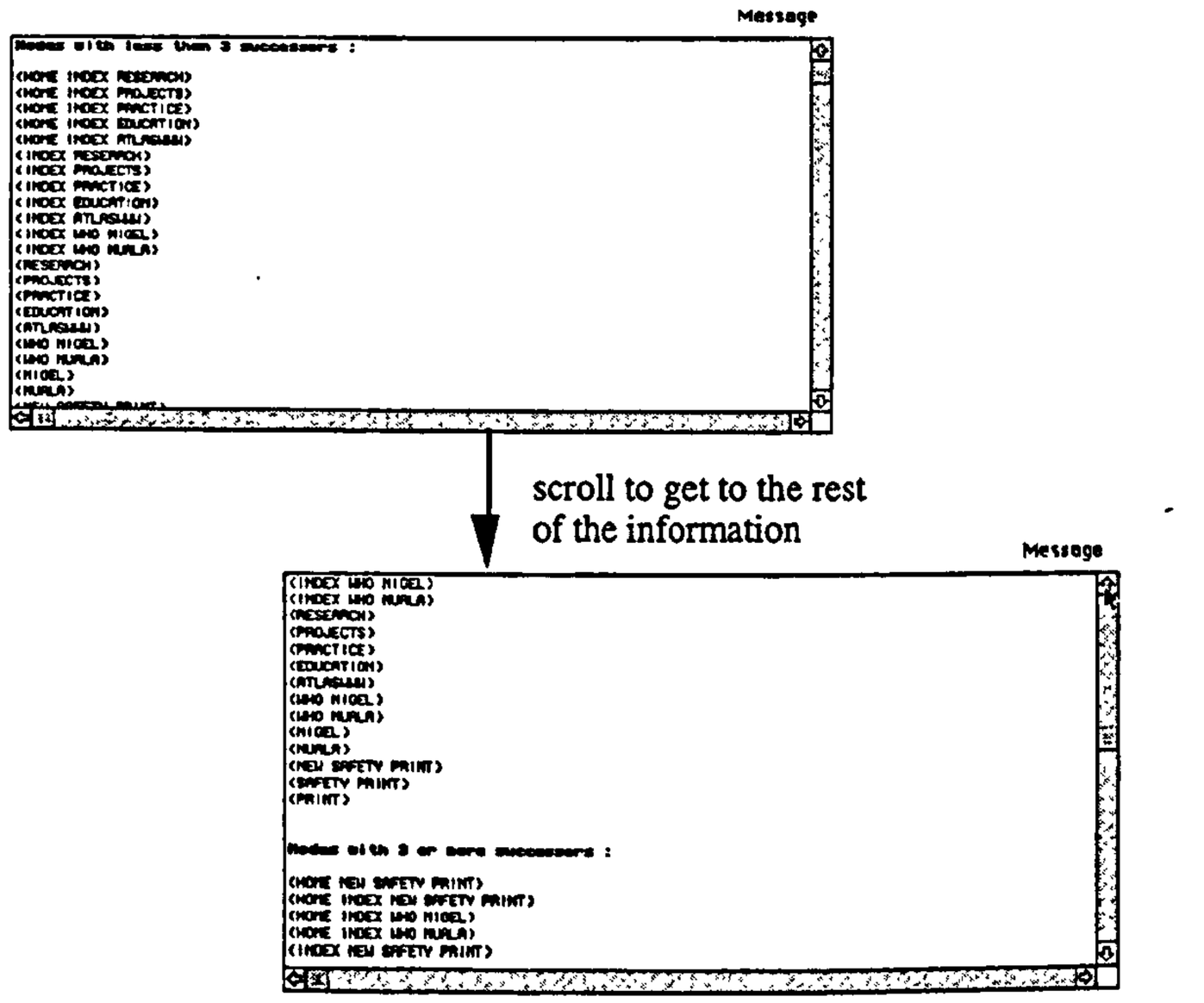


Figure 8.12. Information on web pages with 3 or more successors

8.2.4 Editing and maintenance of a web document

HyperAT provides editing facilities to correct and maintain a website. For example, if one wanted to edit the **index** page, click the **Edit** option in the **Hypertext Menu** option (see figure 8.9). Selecting the **Edit** option opens a window giving a list of all the web pages created. In this example, the window displays the names of all the six web pages. Select the node **index** to edit (see figure 8.13).

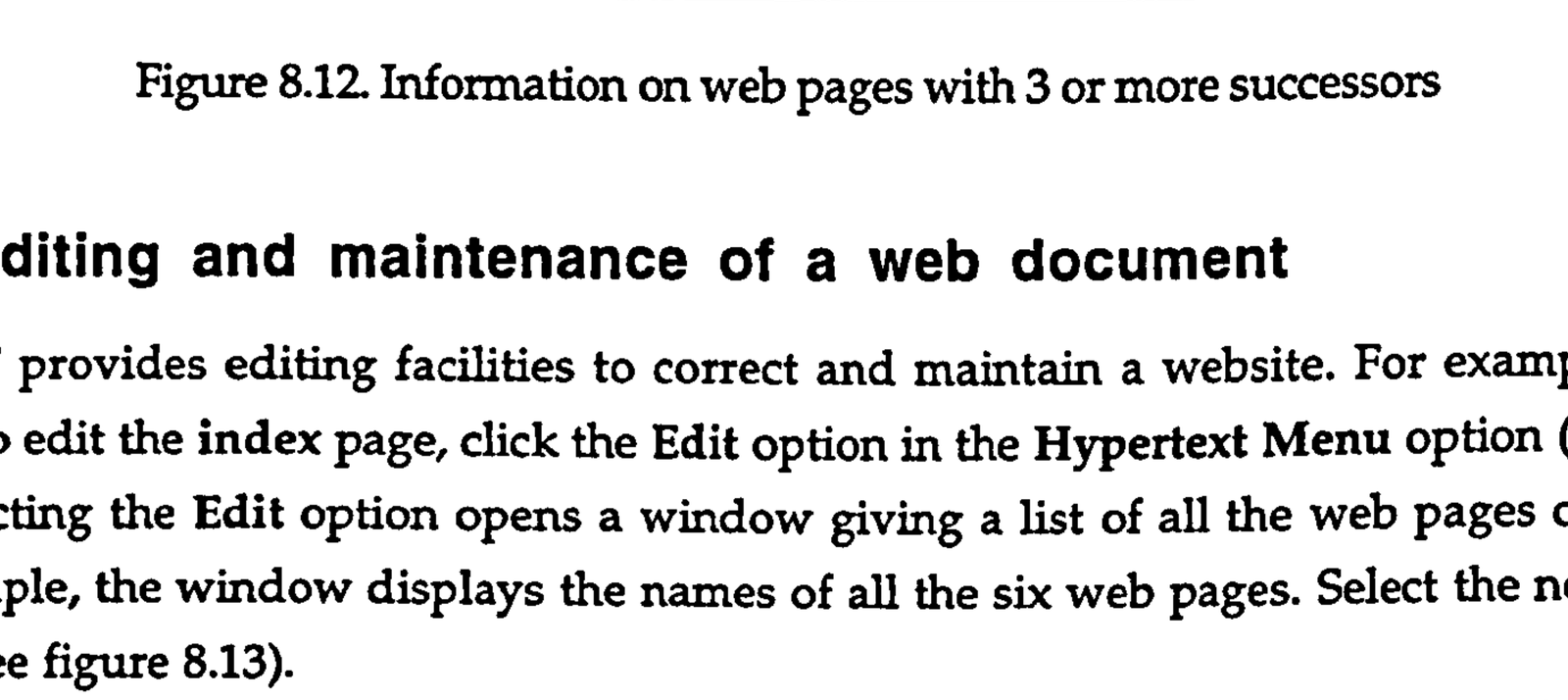


Figure 8.13. Window to select web page **index** to be edited

Pressing **Cancel** abandons the edit operation. If **O.K.** is pressed, HyperAT retrieves information about the web page **index** and displays it as shown in Figure 8.14. Pressing **Save** saves the updated version. Again, the **Cancel** button cancels the operation.

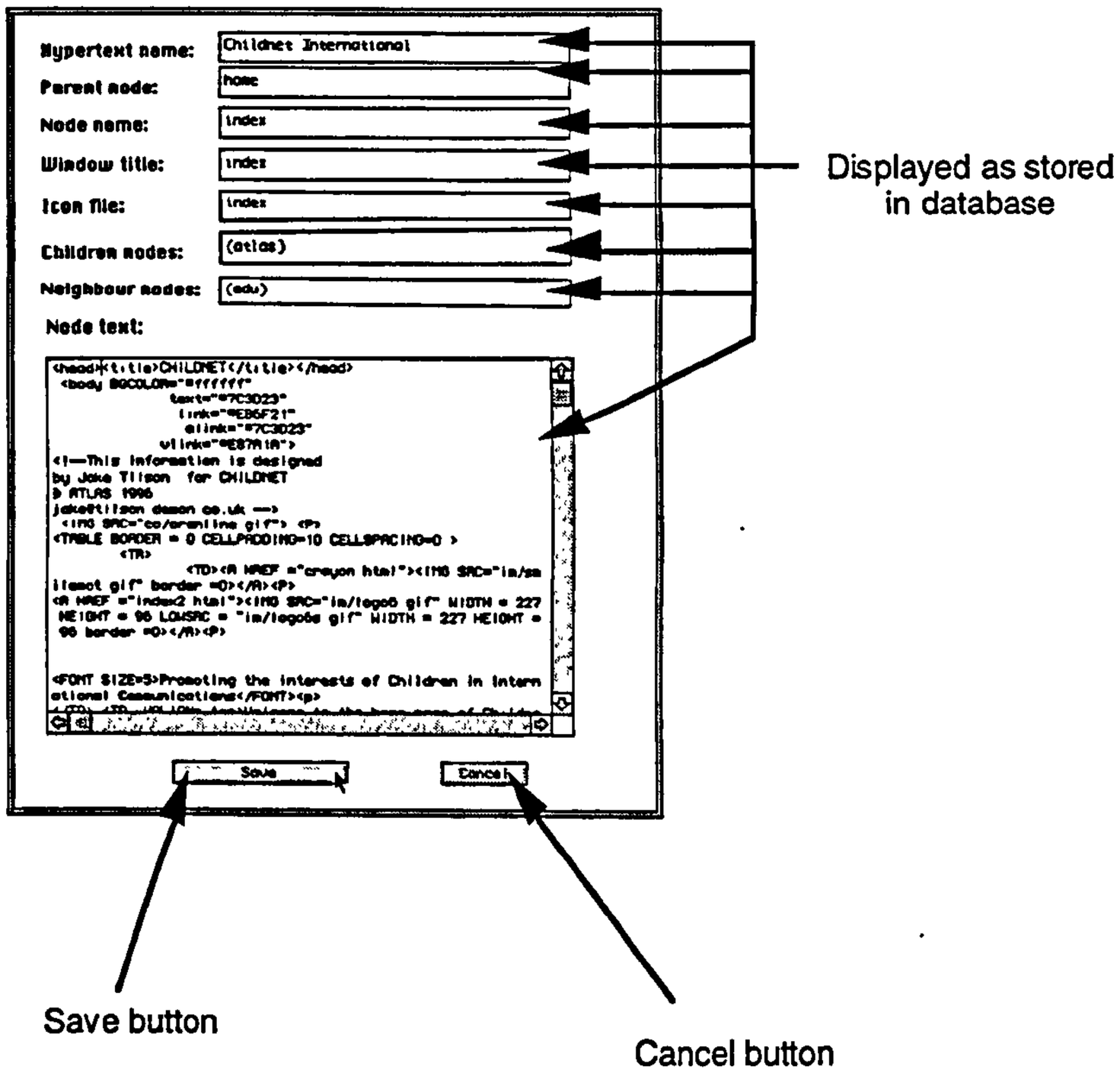


Figure 8.14. Window displaying information of the web page **home** to be edited

Designers can also obtain a hard copy of the web document structure and the corresponding HTML coding using the **View** submenu in the **Hypertext** menu as shown in figure 8.15.

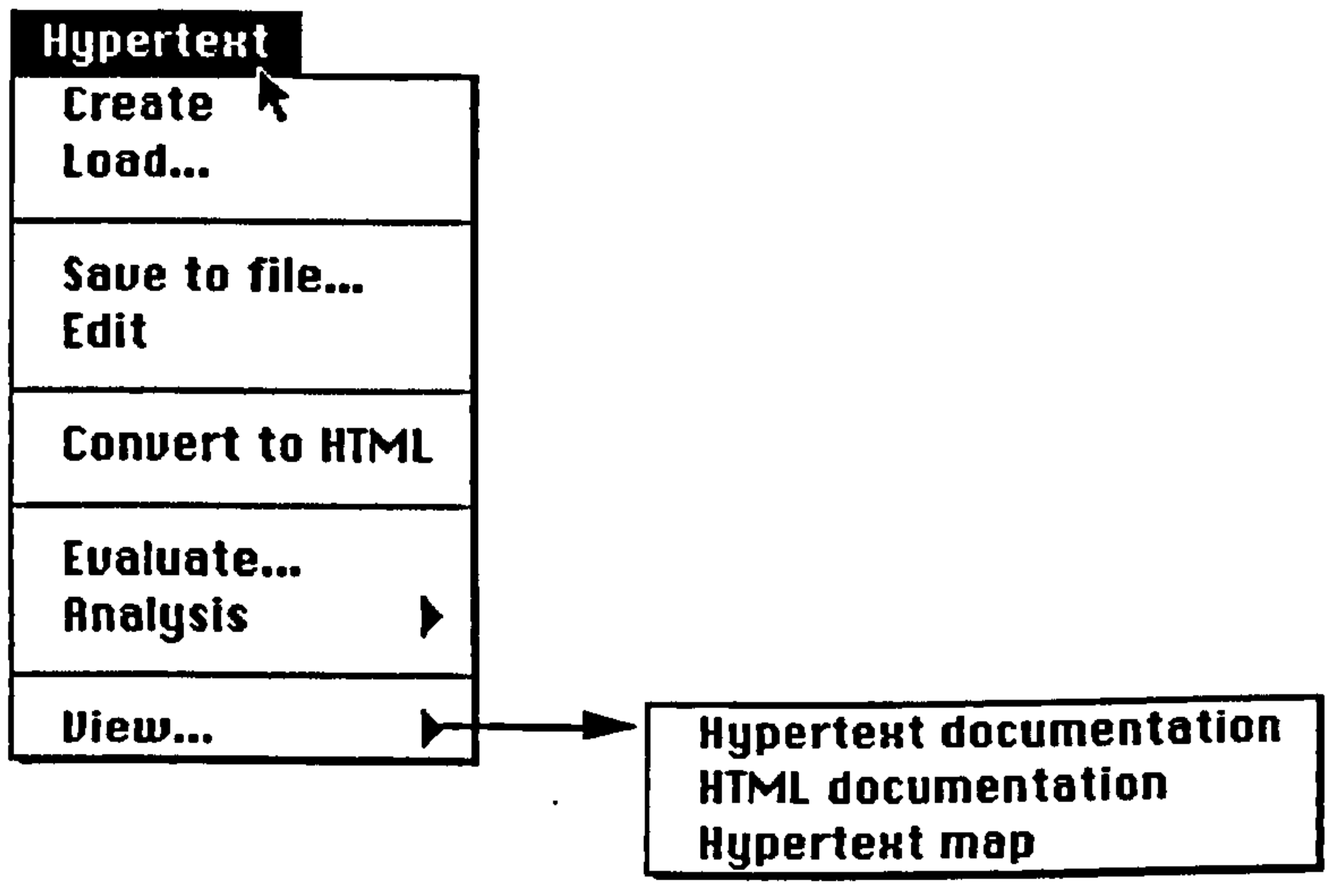


Figure 8.15. Hypertext menu with the View submenu

The **Hypertext documentation** option displays a window providing information on all the nodes that have been created since the start of the HyperAT session (see figure 8.16). This file can also be printed out for documentation purposes.

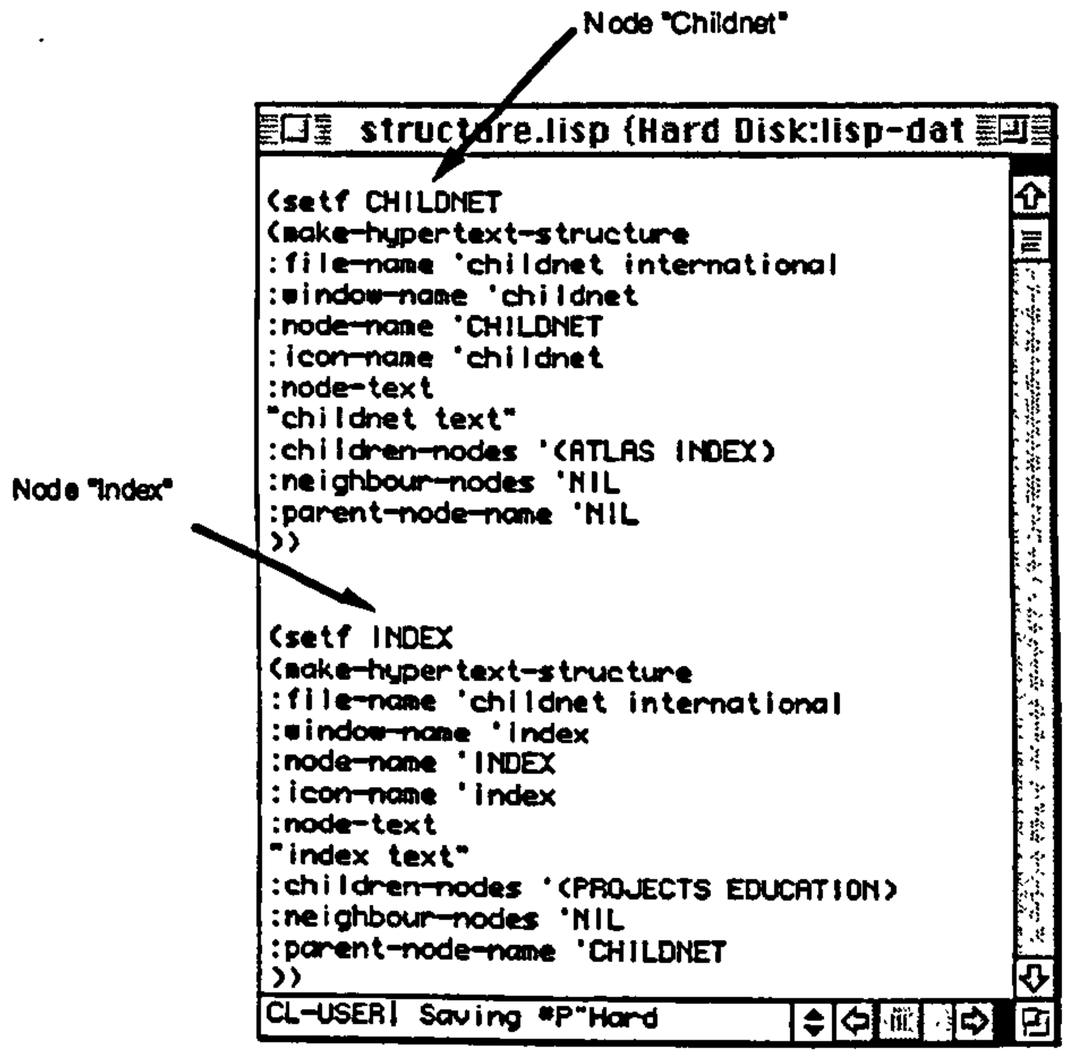


Figure 8.16. Documentation of hypertext structure

The **HTML documentation** option opens a file containing the HTML codes of the hyperdocument (see figure 8.17). This file can also be printed out for documentation purposes.

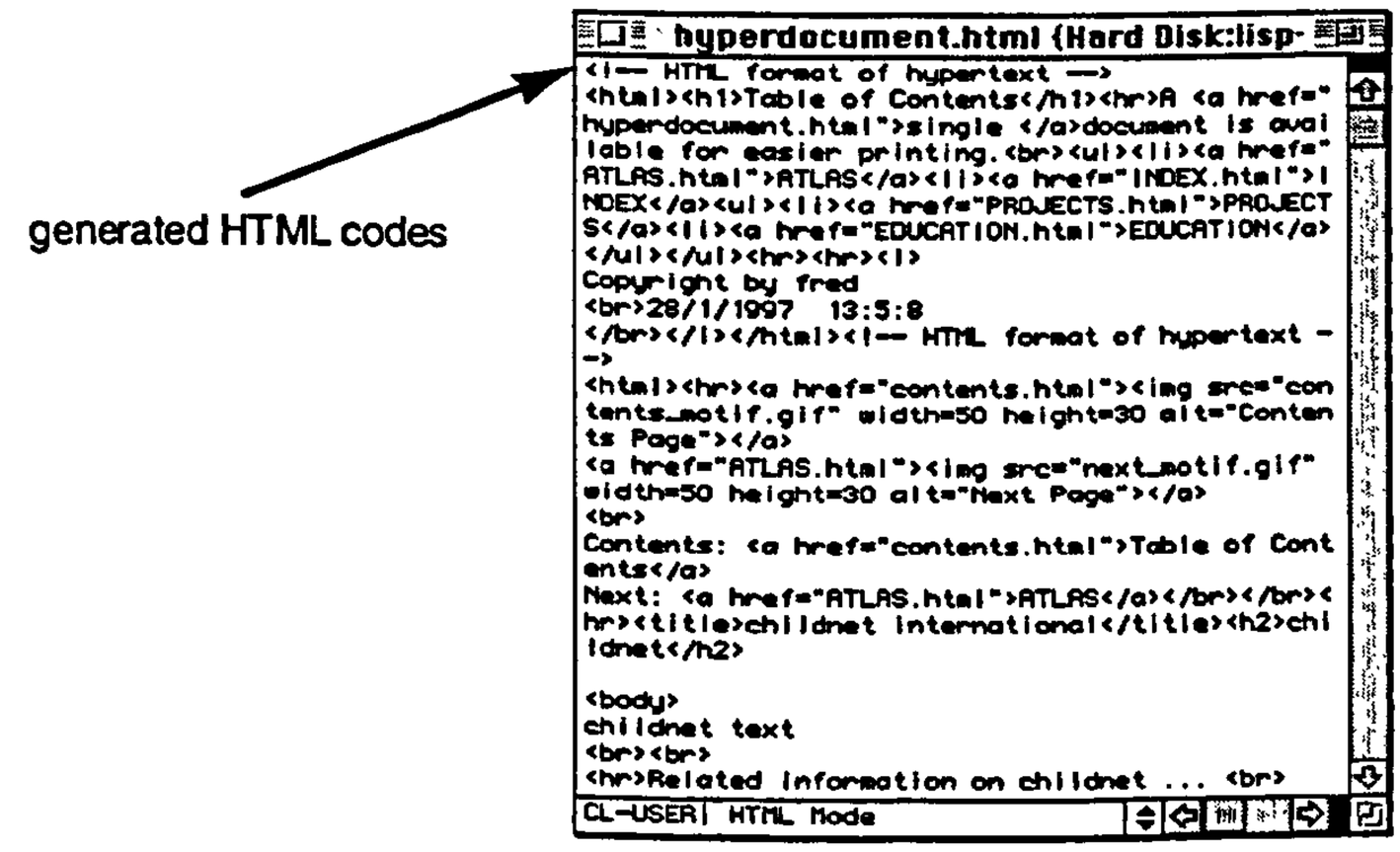


Figure 8.17. HTML coding of hypertext structure

8.2.5 Ending a HyperAT session

To end a HyperAT session, click the **Quit** button in the main HyperAT window. A window appears to ask the designers if they want to quit (see figure 8.18). Press **yes** to confirm ending the session. Another window is popped up to ask to save the three working files permanently. Press **no** not to save the four working files permanently. However, these working files would still be stored in their respective locations as long as the **Create** or **Load** options are not used.

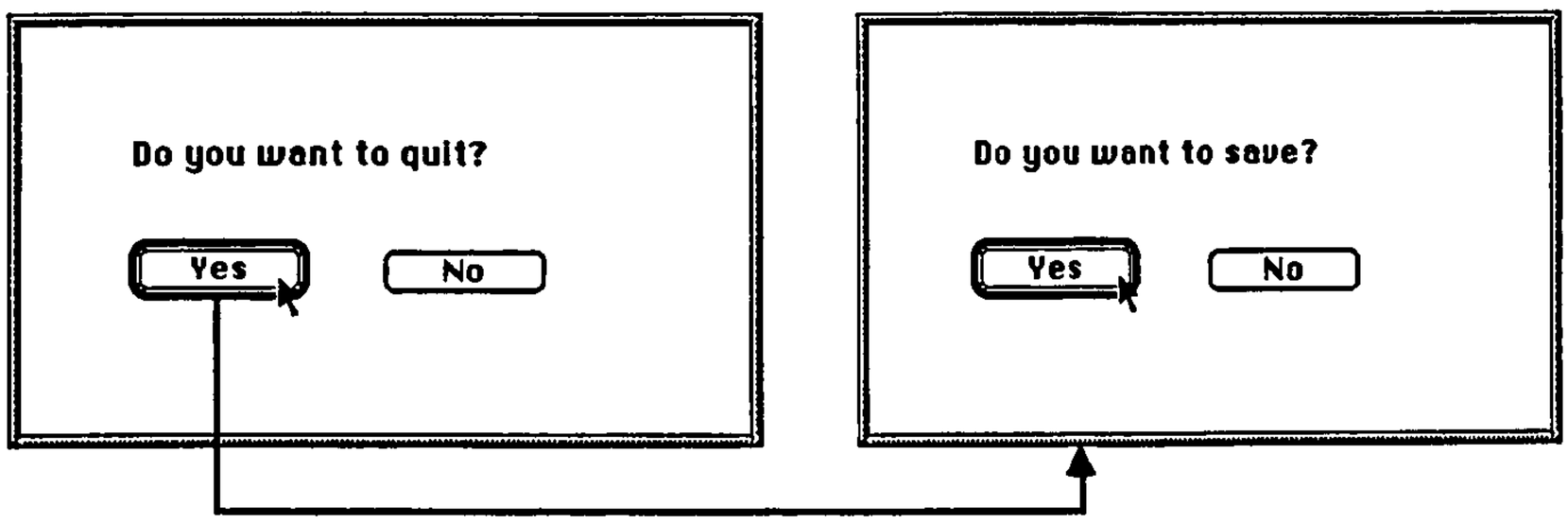


Figure 8.18. Windows showing "quit" sequence

If the **yes** option is selected in figure 8.18, then HyperAT would save the files to a directory "Hard disk:lisp-datafile:". The saving sequence is similar to the operation carried out by HyperAT when the **Save to file** option in the **Hypertext Menu** is selected. HyperAT will prompt the designers to save three files: (i) structure.lisp; (ii) datafile.lisp; and (iii) hyperdocument.lisp. "Structure.lisp" file captures the data structure of the nodes, "datafile.lisp" file captures the parent-child information and "hyperdocument.lisp" file captures the HTML format of the hyperdocument. Figure 8.19 shows the prompts provided by HyperAT to save a structure file. First, a dialog box to enter the name of the structure file to be saved appears. Pressing **O.K.** brings up another window to confirm the file to be saved. "Datafile.lisp" and "hyperdocument.lisp" files will be saved one after another in the similar manner.

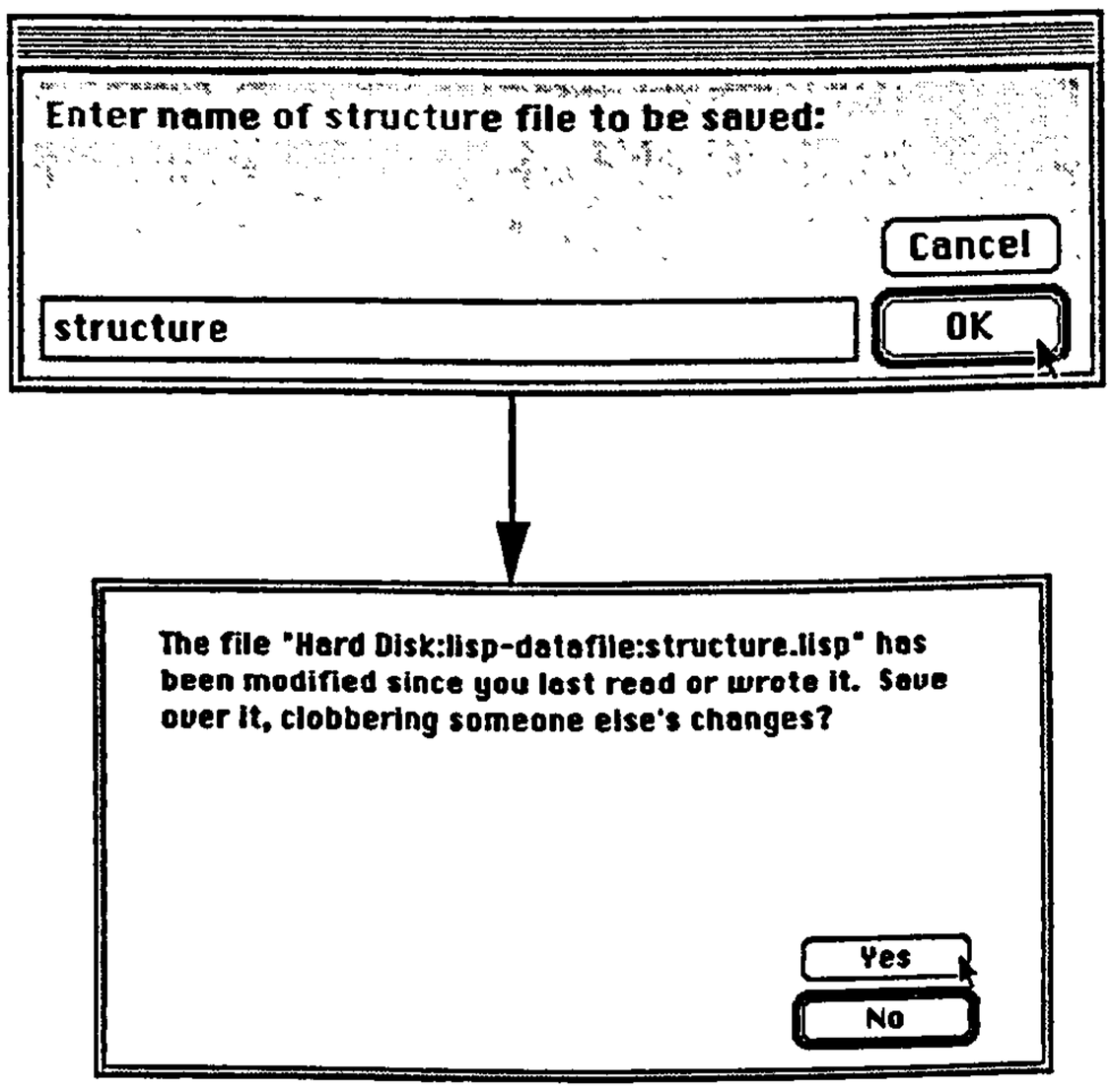


Figure 8.19. Prompts to save the "structure.lisp" file

8.3 Discussion on evaluating HyperAT

§8.2 described the use of HyperAT in constructing, analysing and maintenance of a web document. It has shown that HyperAT possesses basic editing facilities required for the construction and maintenance of a web document such as the "Childnet International", of which the original was created using a standard HTML editor. In addition, HyperAT is capable of carrying out a rough, first-cut evaluation of the "Childnet International" website.

This section wants to obtain qualitative results and impressions on HyperAT in terms of its usefulness and usability. Figure 8.20 shows how an informal evaluation of HyperAT was conducted. This section began by comparing websites produced by HyperAT and a standard HTML editor (see §8.3.1), getting feedback from experts (see §8.3.2) and comparing HyperAT with some web authoring and management tools (see §8.3.3). This thesis, however, will not undertake an extensive evaluation of HyperAT which would require more time and effort than the current scope of this thesis allows. This is not to suggest that the issue of extensive usability and usefulness testing is not important and therefore should be ignored. It is an important element in the design-development-testing iterative cycle of any development process. This thesis would certainly recommend more in-depth evaluation of HyperAT as work for further research.

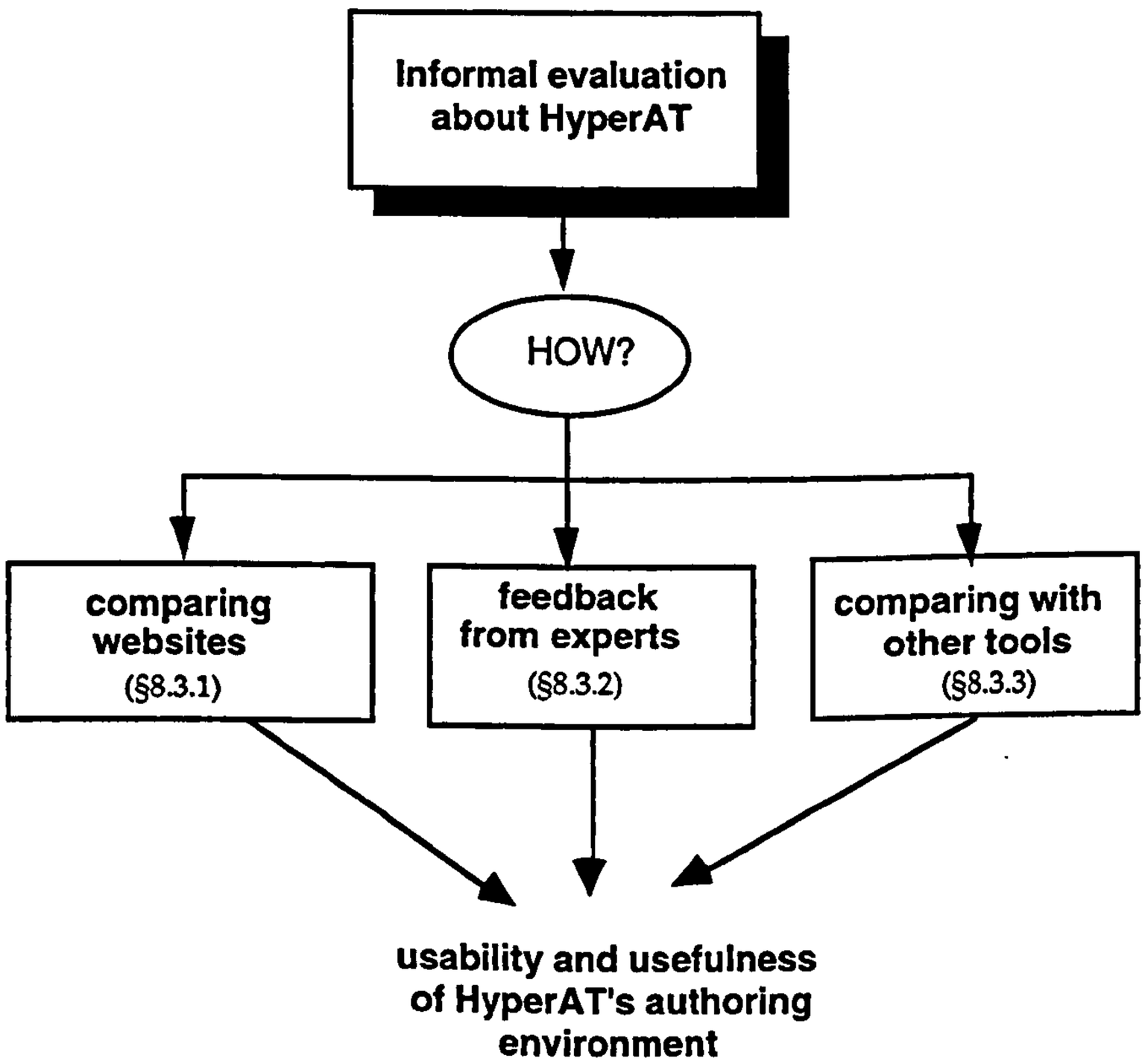


Figure 8.20. Informal evaluation of HyperAT

8.3.1 Comparing web documents

The main objective of this section is to compare the "Childnet International" website reconstructed by HyperAT as described in §8.2 with the original website produced by AtlasWWW, the original creator of the website. Figure 8.21 shows a sample web page created by AtlasWWW, the original creator of the "Childnet International" website, using a standard HTML editor.

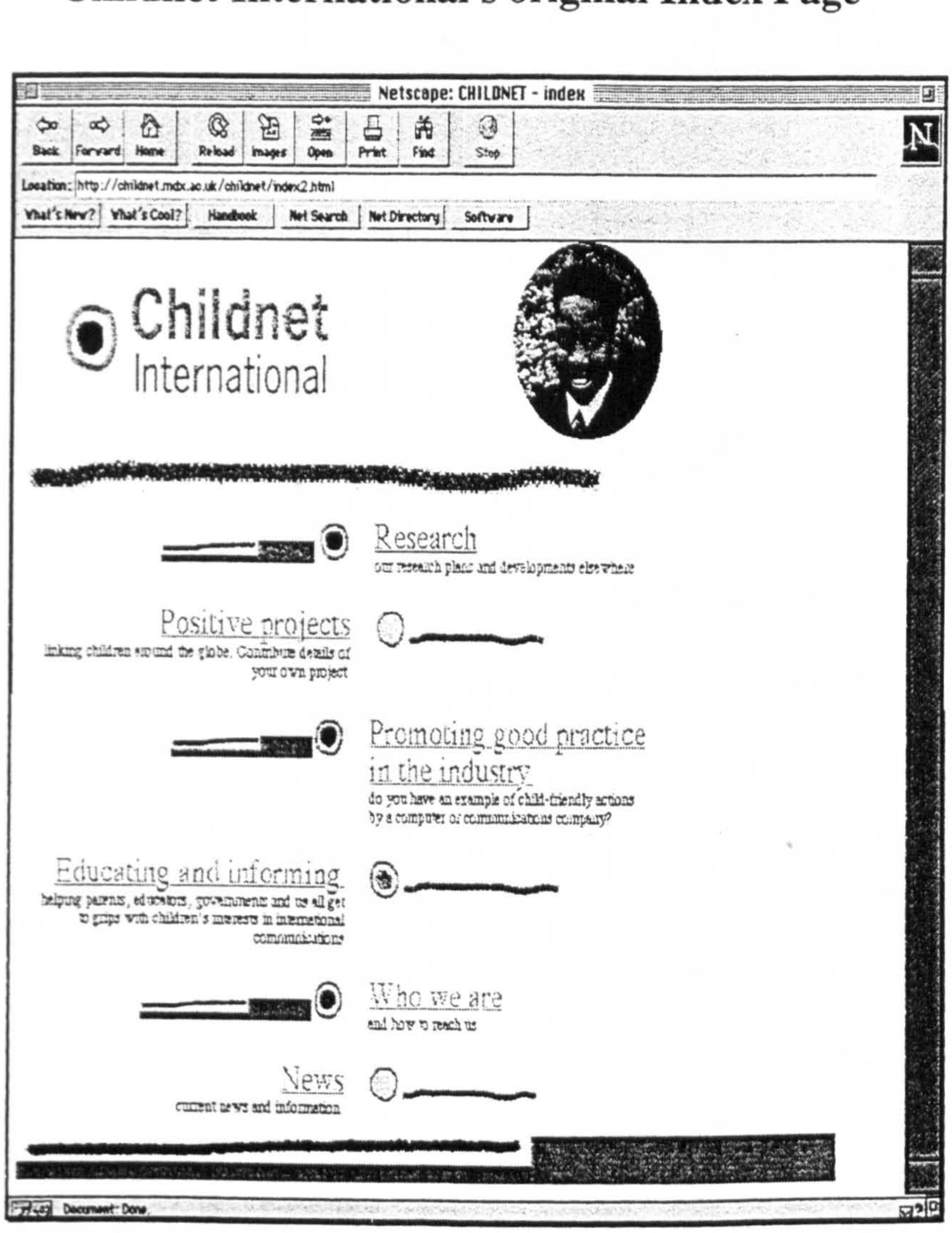


Figure 8.21. A sample web page from the original Childnet International website

Figure 8.22 shows the same web page created by HyperAT. It is to be noted that by using HyperAT, not only was the original "Childnet International" index page re-created but useful information for each web page to help user navigation were automatically generated. They include the automatic generation of text-labelled navigational buttons with prospective views, local views of related pages in relation to end-users' current page, table of contents providing a global view of the structure of the "Childnet International" website, as described previously in §7.3.3.3.

Childnet International's Index Page generated by HyperAT

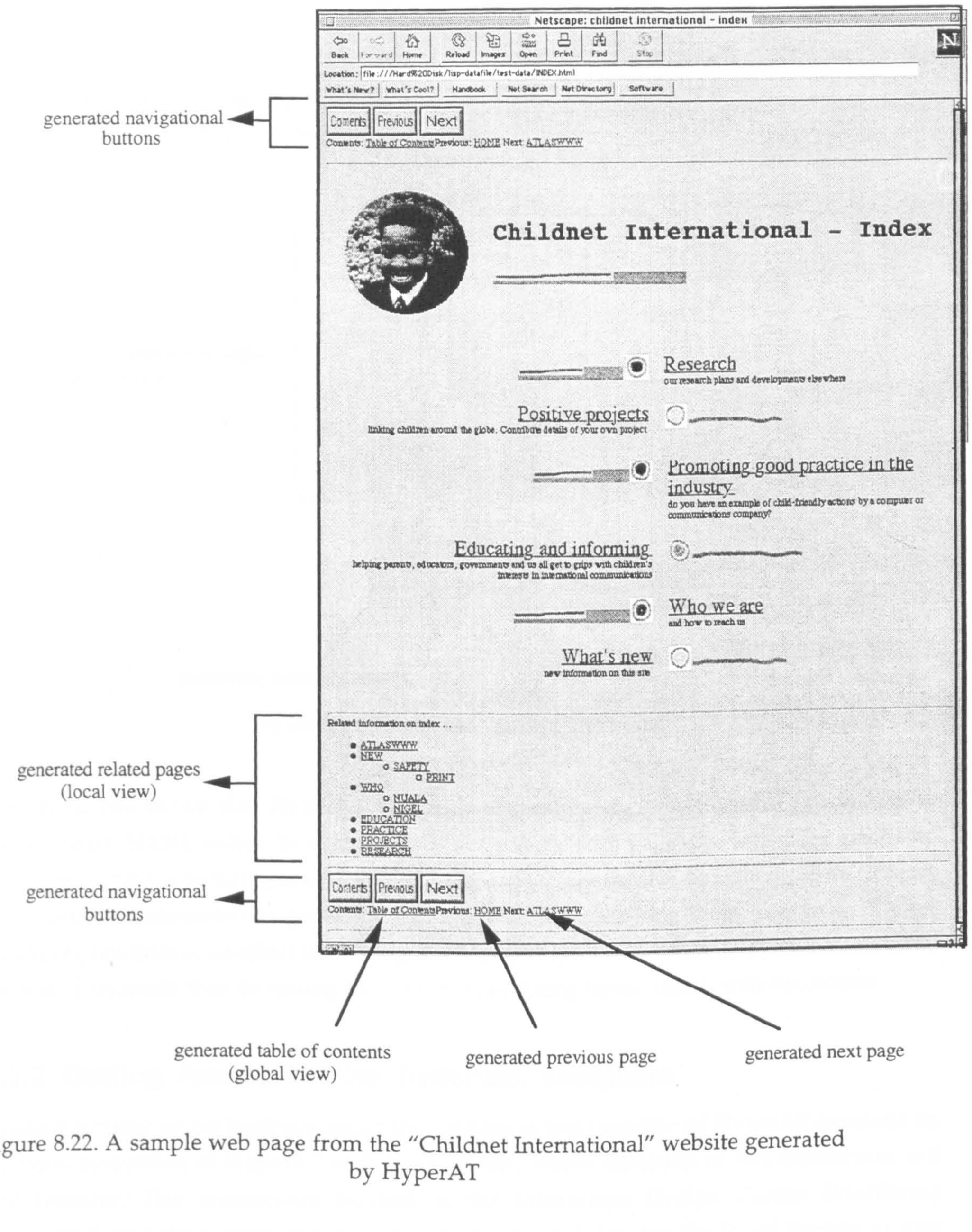


Figure 8.22. A sample web page from the "Childnet International" website generated by HyperAT

For easier printing, a feature to print the website as a single document is also incorporated in this web page. Other automatically-generated information such as the date and time of creation, and creator of the web document, are also displayed in the "table of contents" web page (see figure 8.23).

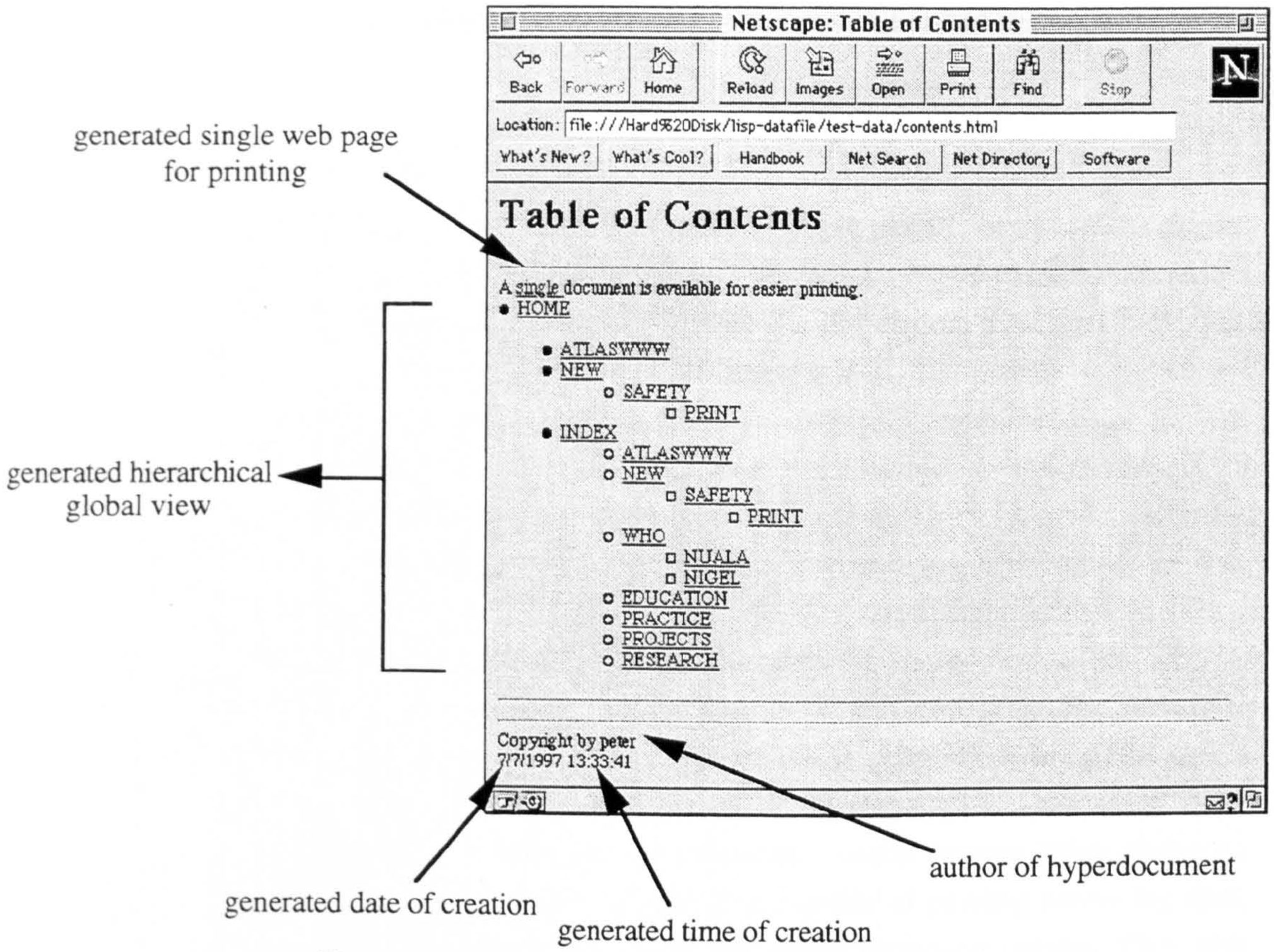


Figure 8.23. Generated "table of contents"

This thesis has shown that HyperAT is capable of creating any web document produced by any standard HTML editor. In contrast with the original web page, the web page generated by HyperAT contains navigational aids that are useful for effective user navigation. Because these aids are automatically generated by HyperAT, incorporating them into the web page would require little or no effort from designers. Designers can then concentrate on the content of the web documents, thus increasing the chance of producing better, usable web documents.

8.3.2 Getting feedback from hypertext designers

Another activity to get feedback about the usefulness and usability of HyperAT involved an informal evaluation of HyperAT with three subjects which consisted of two researchers and one lecturer. The researchers worked at the Interaction Design Centre (Middlesex University), and their research project was to design and develop the Royal Society of Arts web pages. They were chosen because they were experienced in web design. The lecturer was from the School of Computing Science (Middlesex University) teaching undergraduates web design. He was chosen to take on the role of a less experienced designer and thus able to highlight potential problems novice designers may encounter.

Each of the subjects was given a 15-minute guided tour of HyperAT covering both the authoring and usability components of HyperAT. They were then asked to give feedback on whether the authoring features implemented in HyperAT were useful: creation of nodes and

links; generation of map and structure; editing facilities; tracking and checking; testing and evaluation; and capturing end-users' browsing pattern. These have been identified in §6.2.2 as useful authoring aids for future designer tools. The subjects were also asked to comment on the usability of HyperAT.

- **Results and impressions**

  The feedback gathered from the subjects indicates that HyperAT is a useful tool for designing usable web documents. The subjects indicated that all the features implemented in HyperAT are useful in helping designers develop usable web documents. They found the built-in functions for the creation of nodes and links useful. The subjects found the hierarchical structuring simple and intuitive. Because HyperAT separates page content from structure, the subjects felt it would help them concentrate on the content. The automatic generation of overall map and structure was a good feature to provide designers with a graphical overview of the web document. The subjects were also impressed that many processes in HyperAT were automated. These include the automatic structuring of web pages, generated "table of contents" and generated trace of created nodes. The subjects liked the different levels of usability testing HyperAT can perform. For example, facilities to check the structural inconsistencies and complexity were seen as helpful features. The ability to switch from the author mode in the HyperAT environment to the user mode in the Netscape environment was perceived as a useful feature. What makes an impression on the subjects is the idea that HyperAT is capable of parsing server log files, and analysing them to provide information about end-users' browsing pattern. They also liked the idea of HyperAT parsing and converting an existing website into a structure recognised by HyperAT (see appendix A1).

  With regard to the usability of HyperAT, it was difficult for the subjects to make any conclusive remarks without actually using the tool. However, the impression they obtained from the guided tour was that HyperAT appeared to be fairly easy to use, once they got used to working with it.

  The subjects also made some suggestions to further improve HyperAT. These suggestions basically involve making HyperAT more sophisticated in order to handle even more complex designer needs. One comment against HyperAT is that though the hierarchical structuring process to create nodes and links is intuitive, it may be too restrictive. The suggestion is to provide designers with meta-templates to select the different structuring processes. It is also suggested that different templates could also be provided for designers to select the different visual styles of web pages. Another feature that needs further enhancement is to allow for easier modification of the structure based on a "drag-and-drop" feature, and that updating could be done in real-time. To make the analysis of the transaction log files more useful to designers, it is suggested that the map should show the most common paths based on analysis of the transaction log files.

  Since the aim in developing HyperAT was to identify requirements for web authoring tools to address the LIH problem, HyperAT seems to have served its purpose. It would certainly be worthwhile incorporating and implementing these design features into HyperAT.

However, it is to be pointed out that some of the suggestions given by the subjects could be tricky to implement owing to memory and technical constraints. In addition, it does not always imply that having more complex and sophisticated features would always make better tools. Landauer (1995) marshalls overwhelming evidence that even though developers of computers try to incorporate many features to make them more marketable, it merely increases cost and complexity but not usefulness and usability. Therefore, as part of future work for HyperAT, one would have to investigate whether these features, though useful in themselves, would contribute to the overall usefulness and usability of HyperAT.

8.3.3 Comparing with other tools

The task of comparing HyperAT with other tools, be it commercial or research, is not an easy one owing to a plethora of tools available. It is not the intention of this thesis to carry out an extensive comparison of HyperAT with other tools. This thesis will, however, provide general comments about commercial and research tools. Specifically, a comparison of HyperAT with a few selected tools to highlight the benefits and limitations of HyperAT will be carried out. Commercial tools often appear more superior in "look and feel" when compared to research tools. One criticism against research tools is that they are not colourfully, aesthetically developed since their main objective is to test ideas. In addition, research tools tend to be limited in functionality because they are focused to explore specific authoring issues. Since the primary objective of HyperAT is to explore authoring issues to address the LIH problem, HyperAT is more concerned with investigating how authors can be helped to produce usable web documents without feeling "lost". This thesis acknowledges that parts of HyperAT might be done better or done more aesthetically by commercial tools, which are often the result of a team of programmers working together compared to the author being the sole developer of HyperAT.

Compared with HyperAT, NetObjects Fusion is a more powerful page editor and site management tool, allowing designers to create, view and modify site structure graphically. Like HyperAT, it also gives semantics to the relationships between web pages. Although NetObjects Fusion provides analysis of the web documents, there is no attempt to integrate these analysis to make the necessary changes to the design of web documents.

Microsoft FrontPage is another popular tool for managing websites and editing individual pages. Similar to HyperAT, it automatically generates scheduled inserts, table of contents, *etc.* Unlike NetObjects Fusion, it does not allow for direct alteration of site structure. It provides more utilities over the whole website such as spell checking and external link verification, compared to HyperAT.

Yet, when commercial tools are examined in greater depth, they mostly appear to be superior only in limited ways (Thimbleby, 1977). Developers of commercial tools when trying to stay ahead of their competitors would often strive to introduce as many attractive features as possible. Aesthetic appeal is an important criterion, often to the detriment of the usefulness and usability of these tools. It is, therefore, not uncommon to hear complaints from users of

these tools that they only like certain features provided, but may be unhappy about the tools being either too complicated to use, or that the functionality provided is not all that helpful. For example, according to Colborne (1996), Microsoft FrontPage provides too many features making it too complicated to use.

On the other hand, because research tools tend to explore certain aspects of authoring issues, it may not be easy to establish the basis on which HyperAT can be compared. It may be debatable, but one thing is certain — developers of research tools hope their tools have something to contribute to further discussion and research in the areas on which they are working. Two research tools were selected to compare with HyperAT since like HyperAT, they adopt a similar direction and philosophy with the aim to make web authoring easier and more systematic.

WebWriter shares a similar view with HyperAT in that both are integrated systems for constructing web applications that support the creation of web pages. Crespo and Bier (1996) explains that inspired by the HyperCard system, WebWriter allows designers to build an application as a sequence of web pages, where each web page can contain text, images, HTML forms, and the content that is computed on the fly by WebWriter scripts. HyperAT in allowing users to construct an application through a simple form-like interface automatically creates parent-child relationships between pages, instead of linear relationships captured by WebWriter. This thesis believes that in providing a hierarchical structuring feature in HyperAT, it would make HyperAT simple to use since most users find hierarchical structures more intuitive (as discussed previously in chapter 5). However, WebWriter's direct manipulation feature makes editing easier and less cumbersome compared to HyperAT.

Thimbleby's Gentler (1997) is a web authoring tool developed to overcome authoring difficulties encountered by designers. To make web authoring more systematic, Gentler uses a database of pages and a page layout language, as well as reliable design features including hypertext linkage and navigation. Like HyperAT, Gentler aims to reduce the cognitive load of authors , particularly by separating content and design, and by supporting quality control. Compared to HyperAT, Gentler provides a more powerful, flexible support for mathematical analysis of the hypertext structure. However, HyperAT has the ability perform more than just formal analysis of the hypertext structure. It has a program parser to read and analyse end-users' log files in order to understand end-users' browsing behaviour and goals. It is believed that by providing another way of conducting usability testing within HyperAT, designers may be able to gain useful insights into hypertext structural designs relating to cognitive issues from different angles based not only on formal analysis but on real users' behaviour.

In summary, HyperAT does a useful job as a research tool in exploring authoring issues. Compared with other tools, a distinguishing factor of HyperAT is that besides providing the basic authoring facilities to produce web documents, HyperAT also delivers usability results to designers regarding any usability problems that might be detected during its analysis. Because both the authoring and usability components are found within HyperAT, it would be viable as a future enhancement to HyperAT to integrate the usability results into the

authoring process to allow designers to make necessary changes to the web documents more easily. So far, this chapter has described HyperAT's authoring ability in the creation of web documents. Though not fully explored, a feature to read existing web documents and to convert them to the structure recognisable by HyperAT has been implemented (see appendix A1). The result could be exciting and challenging. This means that given any existing web document, it is possible to perform usability testing on them, as well as change the layout of the web document. Presently, the web page layout provided in HyperAT is of one format. If more formats or templates for page design and display are incorporated into HyperAT, it would mean that using HyperAT any web document can be converted to a different page layout. One could also make comparisons of the different layouts by running analyses of them.

8.4 Conclusion

This chapter was concerned with obtaining qualitative results and impressions on the usefulness and usability of HyperAT as a web authoring tool. This thesis has shown that HyperAT is capable of reconstructing an existing website originally produced by a standard HTML editor, thus confirming that HyperAT does possess basic editing features found in standard HTML editors. Having established the fact that HyperAT is able to construct a web document, the next thing done was to compare a sample web page of the original web document with a similar web page generated by HyperAT. This thesis has also shown that with little or no effort from designers, the web page produced by HyperAT contains additional navigational aids automatically generated to help end-user navigation.

An informal evaluation of HyperAT was conducted with a handful of hypertext designers. On the whole, the feedback gathered from these experts indicates that HyperAT is a useful tool for designing usable web documents. Since these hypertext designers were only given a guided tour of HyperAT, they were unable to comment conclusively on the usability of HyperAT. More extensive evaluation of HyperAT involving both designers and end-users is recommended as work for further research. However, the impression obtained from preliminary evaluation was that HyperAT appeared simple, fairly easy to use.

Finally, HyperAT was compared with some web authoring tools, highlighting the benefits and limitations of HyperAT. Being a research tool, the author believes that HyperAT does a useful job exploring usability issues surrounding the LIH problem.

Although HyperAT has not undergone extensive evaluation, the results of the evaluation illustrate the potential of HyperAT as a useful and usable web authoring tool in addressing the LIH problem. Embodied within HyperAT are useful design aids to help hypertext designers to manage the complexity of the design and validation processes without themselves getting "lost". This thesis has also shown that the web documents produced by HyperAT contain useful navigational aids situated in well-established literature to help hypertext end-users navigate without getting "lost".

Section IV

Concluding remarks

Chapter 9

Summary and future work

Chapter objectives:

This chapter summarises the work done and opportunities arising from the work at the time of writing up this thesis. The chapter aims not only to highlight the technical aspects of the work but also to relate the author's experience in completing this PhD. By doing so, the author is able to share the complete experience of the work, which would be of use to anyone who may wish to continue the work or embark on a similar research project. Of course, the complete experience includes low level detail, but this is relevant to the overall results and may have more than incidental interest to researchers pursuing this work.

9.1 Summary of thesis

This research has been motivated by a desire to investigate ways to eliminate or at least, ameliorate the LIH problem in hypertext. Some conclusions have been reached, not just in the design and implementation of a prototype hypertext and a hypertext research authoring tool, but also with respect to disciplines in human-computer interaction, user modelling and software engineering.

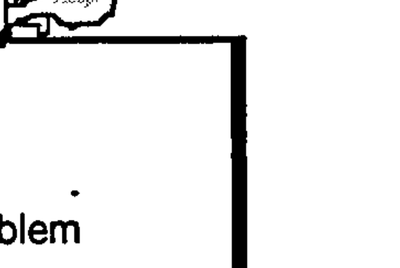
Figure 9.1 outlines the work carried out in this thesis to address the LIH problem. The thesis began with chapters 1 and 2 giving an introduction to the investigation and definition of the research question. By restructuring the LIH problem, this thesis set out the reasons for a *proactive, multi-disciplinary* approach taken to address the problem. Chapters 3 – 6 recorded investigations carried out on standalone hypertexts drawing insights from diverse fields in hypertext, human-computer interaction, cognitive psychology and software engineering. To demonstrate the ideas proposed to address the LIH problem, the web was chosen as a hypertext example. This has led to the design and development of HyperAT, a prototype hypertext authoring tool in chapter 7. Chapter 8 gathered qualitative results and impressions on the usefulness and usability of HyperAT as a web authoring tool.

Introduction and Problem Definition

Chapter 1	Introduction
Chapter 2	Hypertext and the "lost in hyperspace" problem

Hypertext Research

Chapter 3	Need for good design principles and guidelines
Chapter 4	Need for an engineering, task-based approach to understand users' needs
Chapter 5	Need for good hypertext structure
Chapter 6	Need for better support tools for authoring



Hypertext Example: The World Wide Web

Chapter 7	Multi-disciplinary approaches and web authoring
Chapter 8	Feedback on HyperAT

Figure 9.1. Outline of work carried out in this thesis to address the LIH problem

9.2 Methodology, research activities and feedback

Since the LIH is a complex problem involving both engineering as well as psychological issues, a productive problem-solving approach as discussed in §2.4.1 was adopted. As a result, many ideas were generated and the work reported in this thesis covered many "strands" of investigations into different disciplines. This would not have been possible if this thesis had focused on just one aspect of the LIH problem.

Because the intention of this thesis is to get qualitative results and impressions on the design as well as the usability issues surrounding the LIH problem, experiments were designed small and focused in order that frequent/common problems but not infrequent or minor ones were first identified.

Before outlining the contributions of this thesis and matching them against the objectives set out in §1.2, an overview of tools, systems, entities or methods employed to carry out the research activities discussed in chapters 3 – 8 is given. The research activities, as indicated in table 9.1, consisted of well-accepted stages in the software development lifecycle such as designing, building and evaluation. 'Designing' refers to "putting ideas together after gathering information and analysing end-users' needs". 'Building' refers to "demonstrating these ideas by constructing prototypes". 'Evaluation' includes "the means, procedures and standards for testing systems using a small sample of real users". 'Tool' refers to HyperAT, a hypertext research authoring tool, developed in the course of this thesis to build usable web documents. 'Systems' refer to hypertexts, which are divided into two groups: those developed by the author (for example, *Basic Computer Anatomy* in §3.3.3, *Childnet International* in §8.2.2) and those not developed by the author (for example, ACM's *Hypertext-on-hypertext* in §3.5.1 and *Childnet International* in §8.3.1). 'Methods' employed include questionnaire, interview and task analysis. 'Entities' include executable user models, real users and experts.

Table 9.1 also shows the respective sections within the thesis where these research activities were used, as well as indicates the work that had been presented at international and national conferences in hypertext (for example, IEE Colloquiums Sep'95 – Oct'95; Hypertext'97) and human-computer interaction (for example, HCI'95 – '97; APCHI'96). For example, table 9.1 shows that the design and building of HyperAT was discussed in §7.3.1 – §7.3.4 with the asterisk indicating that the ideas described in §7.3.1 – §7.3.4 had been presented at a conference.

Encouraging and positive comments have been received from referees as well as participants at these conferences regarding the approach taken to address the LIH problem. These comments include "the approach taken in addressing the LIH is plausible and logical", "it is an interesting angle, and a hot current debate", "[HyperAT] actually seems to provide the appropriate framework for the design method evaluation through the metrics used", "[The idea of incorporating] executable user modelling is interesting", *etc.* Comments here are by no means unanimous, indicating that the area is still controversial.

	RESEARCH ACTIVITIES		
	Designing	**Building**	**Evaluation**
Prototype authoring tool			
HyperAT	*§7.3.1, *§7.3.2	*§7.3.3, *§7.3.4	§8.1–8.4
Hypertext prototypes			
Basic Computer Anatomy	*§3.3.3	*§3.3.3	*§3.5.2
Childnet International	§8.2.2	§8.2.2	§8.3.1
Commercially-produced hypertext			
ACM's Hypertext-on-hypertext	n.a.	n.a.	*§3.5.1
Childnet International	n.a.	n.a.	§8.3.1
Methods			
questionnaire	§3.4	§3.4	§3.5
interview	§3.5	§3.5, *§4.2.4, §4.3.4 §8.3.2	n.a.
task analysis	*§4.2	*§4.2	*§4.2
Entities			
computerised users	n.t.	n.t.	*§4.3.4
real users	n.a.	n.a.	§3.5, *§4.2.4, §5.3.3, *§6.2
experts	n.a.	n.a.	§8.3.2

Key:
* : paper published,
n.a. : not applicable,
n.t. : not within scope of this thesis

Table 9.1. Tools, systems, entities or methods employed to carry out research activities in this thesis

9.3 Contributions of this thesis

The main contribution of this thesis is the *proactive, multi-disciplinary* approach taken to address the LIH problem, a different stance compared with most other research efforts in HCI. By adopting a "breadth-first investigation" approach into the various disciplines, this thesis has gained a wider insight and restructuring of the LIH problem, and therefore was able to see novel interpretations that might lead to viable solutions. It has also shown how bridges between various disciplines and perspectives were built and used to create opportunities for creative synergy to address the LIH problem. HyperAT built as a result of the investigation is an experiment to demonstrate this form of collaborative endeavour which is increasingly necessary to address complex problems, of which the LIH problem is one.

This thesis, however, would like to qualify that the approach taken represents only one possible way of systematising the design of usable, well-structured hypertexts drawing upon investigations into different disciplines. The thesis does not make any claim that this is the only way to address the LIH problem in hypertext, or the web in particular. It has, however, shown how designers could be helped to better understand the design and usability issues surrounding the problem, suggesting viable solutions to eliminate or at least reduce the impact of the LIH problem in hypertext.

In the course of searching for solutions to address the LIH problem, some conclusions have been made with regard to contributions made to human-computer interaction, cognitive psychology and software engineering. The remaining portions of this section detail contributions made in these disciplines.

9.3.1 Understanding the "lost in hyperspace" problem

In order for correct and appropriate solutions be sought to address the LIH problem, this thesis believes that it is important to understand the reasons why end-users feel "lost" when using hypertext. This is because by having a proper understanding of the problem space designers would stand a better chance of searching for appropriate solutions to address the LIH problem. Therefore, objective 1 (§1.2) was defined to re-visit the LIH problem. This thesis argues that the LIH problem experienced by end-users may be due to *conceptual disorientation*, "lostness" within the end-user's mind (§2.2.2), or *spatial disorientation*, "lostness" within hypertexts (§2.2.3). It concludes that "lostness" could be attributed to end-users' or designers' limitations.

Could this be one of the reasons why in spite of the fact that much research effort have been invested to address the LIH problem, solutions have only been marginal? This may not be surprising if wrong or inappropriate solutions are being sought, owing to inaccurate or wrong understanding of the LIH problem.

9.3.2 Human-computer interaction, design principles and guidelines

The intention of this thesis is to try to make the LIH problem dissolve by doing things well from the start, that is, by perhaps ensuring that good hypertext design principles and guidelines are incorporated in the design of hypertexts in the first place (see objective 2, §1.2). Based on the investigation carried out in this thesis and other independent studies, this thesis drew up some design principles and guidelines for hypertext.

In contrast with most research efforts which tend to provide a list of do's and don'ts of good design practice, and do not tell how they are derived, this thesis attempted to present a framework, a systematic approach to define design principles and guidelines for hypertext (chapter 3). In order to draw up a list of relevant design principles and guidelines for hypertext, an investigation of general design principles and guidelines for interactive systems was conducted. The design principles for hypertext authoring were examined, by assessing their relevance from a cognitive perspective, since the interaction between the end-users and hypertext is essentially cognitive. Assimilating findings from usability research, the design guidelines for hypertext were categorised into different areas of testing. The design guidelines were drawn up in the form of a questionnaire to test the design principles implemented in any hypertext.

Evaluation of hypertexts as reported in §3.5 indicates that the questionnaire (that is, design categories testing the design principles) is useful in providing useful insights into the design of the hypertext and in understanding the relationships among design principles, end-users' satisfaction and end-users' perception of the effectiveness of the hypertext.

This thesis has argued for and demonstrated a framework for drawing up design principles and guidelines for hypertext. It has shown how related disciplines could be used to underlie the formulation of design principles for hypertext, and for that matter interactive systems in

general. Design principles need not be vague, and this thesis has shown one way of making them testable and quantifiable through design categories formulated in the form of a user questionnaire. The ultimate use of design principles and guidelines is to provide reference and guidance for designers during the development process. The benefits are tremendous in that good principles and guidelines can help to shorten and reduce the number of iterations involved in the design-evaluate-redesign cycle, leading to savings made in terms of time, money and resources.

9.3.3 Task analysis and software engineering practice

To achieve objective 2 (§1.2) to better understand end-users' needs, this thesis turned to task analysis techniques and software engineering practice. Two weaknesses were identified as a result of the investigation (§4.1):

W1: there is a lack of a disciplined and systematic approach to capture end-users' needs (Carmel, Crawford and Chen, 1992)

W2: prototype is not thoroughly tested before developing into final system (Glushko, 1992)

To address weakness W1, a cognitive, task-based approach to understand end-users' behaviour and navigational issues when using hypertext was proposed (§4.2). The task *browsing* was used as an example to describe how end-users' common tasks can be broken down into simpler subtasks. From these subtasks, recommendations for the interface design of a hypertext were made.

Based on the design recommendations, a prototype hypertext built in §3.3.3 was modified to incorporate suggestions on how to carry out possible activities normally associated to browsing (see objective 4, §1.2). Although there was no significant difference in terms of the performance of the subjects who took part in the experiment, the subjects indicated they were more satisfied with the prototype that incorporated design recommendations obtained from the cognitive task diagrams. From this study, it can be shown that end-users' needs can be straightforward if they are systematically analysed using task analysis (§4.2).

To address weakness W2, evaluation methods and techniques to investigate the usability or potential usability of hypertext were examined (§4.3). If designers were to apply appropriate evaluation methods and techniques to the software development lifecycle, quality hypertexts can be guaranteed. There are many propositions. This thesis has argued for and presented an engineering, task-based approach to understand end-users' needs and browsing behaviour by giving them more structure. Consequently, an improved Conceptual Design stage built on top of the well-established software development lifecycle by incorporating the use of cognitive task diagrams and executable user models was proposed (§4.4). They could be used to pinpoint usability problems as early as possible in the design process so that remedial work could be minimised in the later stages of development.

This has significant implications. This thesis has described a way of improving software engineering practice by demonstrating how end-users' needs can be effectively represented,

analysed and evaluated as early as possible in the development lifecycle. Although not supported by empirical evidence, this thesis believes that this could lead to savings in terms of time, money and resources.

9.3.4 Hypertext structuring

To address the LIH problem, designers need to ensure that effective structuring is embraced in hypertext design (see objective 2, §1.2). An empirical study (§5.3) conducted to understand the relationships LIH has with hypertext structure and end-users' representation of the hypertext structure yielded some interesting findings. From the study, it is noted that non-linear structure in hypertexts makes it difficult for end-users to ascertain the layout of the hypertext network, leading to conceptual disorientation. However, if the interface design does not reflect accurately and clearly the information structure of hypertext, then spatial disorientation may take place. Therefore, to reduce the impact of the LIH problem, the structure of the hypertext should be more clearly and explicitly shown so that end-users do not need to guess the structure of the hypertext.

A preliminary study into the types of structures designers commonly used to represent hypertexts was conducted (§5.3.3). All the subjects produced structures that were somewhat hierarchical. Independent studies by Mendes and Hall (1997) and Smith and Newman (1996) confirm the results that hierarchies are most commonly used since they are "intuitive", and are easily understood by both hypertext designers as well as hypertext end-users (Garzotto *et al.*, 1991). Although hierarchical structures could usefully be applied in most hypertexts to address the LIH problem, it is to be noted that designers should ensure that the hierarchy does not get too deep as this would affect navigation which might lead to end-users feeling disoriented.

This thesis does not claim that the hierarchical structure is the best, and in some contexts alternative structures such as linear structures with loops or guided tours might do the job to avoid the LIH problem entirely. However, in most hypertext applications and especially in the types of hypertexts this thesis is interested, hierarchical structure is a good, intuitive way to organise information.

This thesis has demonstrated in chapter 7 how the knowledge about structures in hypertext could be effectively automated in the design process so that designers are free to concentrate on other design issues. Since designers' likely authoring structures are hierarchical or quasi-hierarchical, it seems plausible to incorporate a hierarchical authoring structure into an authoring tool, so that authoring becomes "intuitive", easy to follow.

This thesis has also demonstrated in chapter 7 how designers could be helped to avoid structural inconsistencies and mistakes. Two simple heuristics were proposed to analyse hypertext structure by firstly, treating the end-user as a computer, and secondly treating the hypertext as a graph. This thesis argues that a well-ordered graph of information is easier for both the authors to manipulate, and end-users to navigate. Consequently, a list of useful metrics was drawn up to allow designers to perform checks on network connectivity,

identification of dead ends and loops, *etc.* (§5.4). Some of these metrics have been implemented in HyperAT, which this thesis has shown that formal analysis of the hypertext structure could be performed easily and rapidly by automating the calculation of these metrics in an authoring tool.

9.3.5 Hypertext authoring

To address the LIH problem, objective 3 (§1.2) was defined to examine current popular authoring tools for standalone hypertexts. The results of a survey carried out to investigate authors' satisfaction with popular authoring tools showed that the designers were only partially satisfied, and that present authoring tools are not providing the essential and helpful features yearned for by hypertext designers (§6.2.1). Many problems encountered in authoring were highlighted, but the main problems could be attributed to poor support in: (i) integrity checking of nodes and links; (ii) capturing end-user requirements; (iii) analysing end-user interactions; and (iv) importing facilities.

A list of essential and helpful features for future authoring tools was compiled. For future authoring tools to be good design aids with a reduced tendency for the LIH problem to arise, this thesis calls for developers of authoring tools to consider seriously incorporating into or enhancing these features in their products. Developers of authoring tools should carry out more investigations into designers' authoring concerns in order that designers' needs are met, which inevitably would lead to more satisfied designers with a better chance of producing better quality hypertexts.

9.3.6 Experiment in collaborative efforts involving many disciplines

HyperAT demonstrates how research efforts need not be focused on a particular discipline to solve a problem. It opens up possible collaborative efforts involving many disciplines that may be necessary to solve complex problems, of which the LIH is an example.

The ideas generated from the multi-disciplinary approaches, as described from chapters 3 – 6, underlie the design of these components. As described in §7.3.3, the authoring components in HyperAT provide the basic authoring environment for creation, loading and modification of hyperdocuments. Figure 9.2 summarises some of the authoring features implemented in HyperAT.

HyperAT has demonstrated that design principles and guidelines could be automated without designers having to worry about their implementation, thus promoting the concept of distributed authoring. They include ensuring consistency of presentation, structuring information for the end-users, providing status of the hyperdocument, and providing a prospective view.

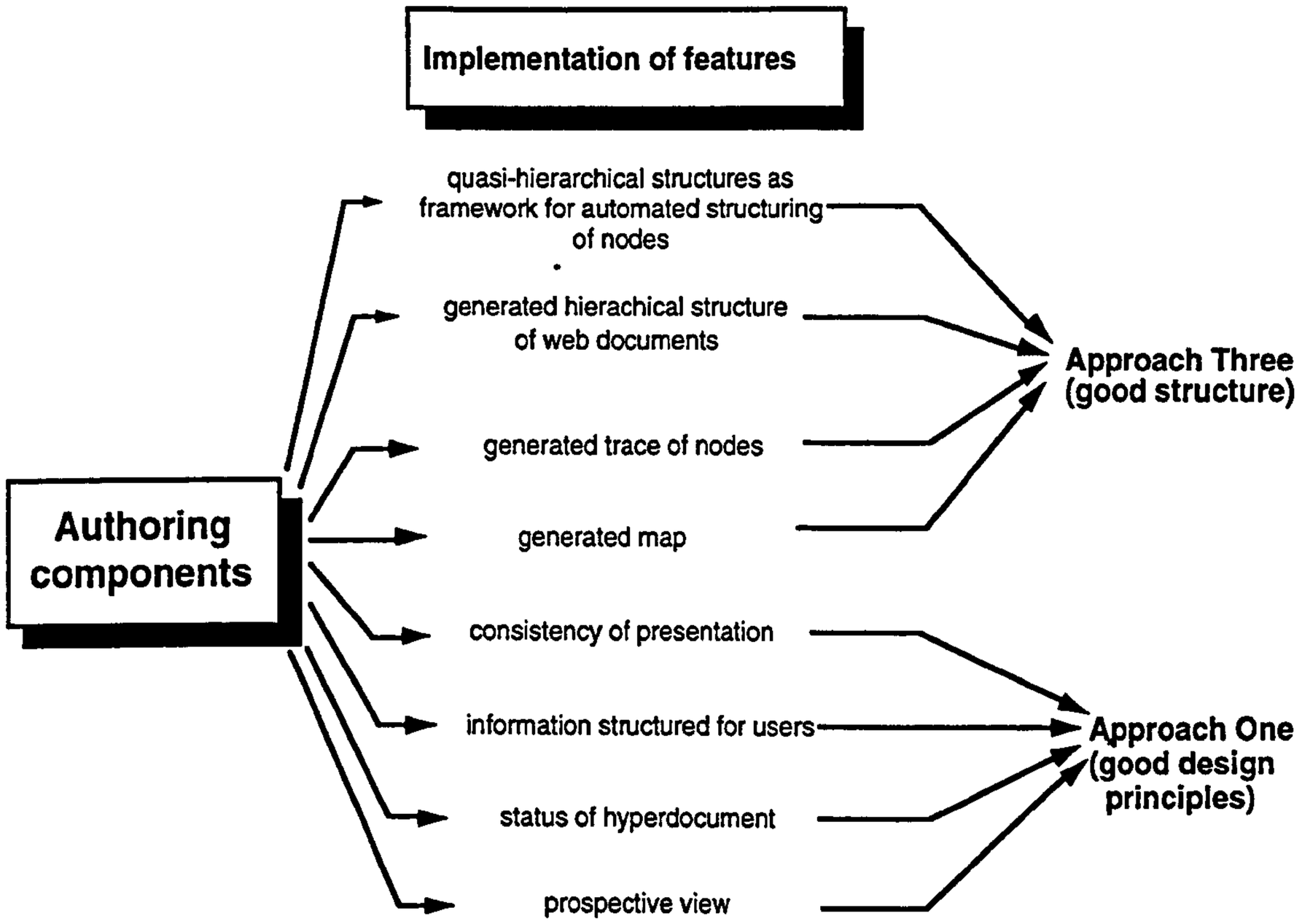


Figure 9.2. Some features implemented in the authoring components in HyperAT

Besides providing the basic authoring facilities to produce web pages, HyperAT also delivers usability results to designers regarding any usability problems that might be detected during its analysis (§7.3.4). Figure 9.3 summarises the features implemented in the usability components of HyperAT.

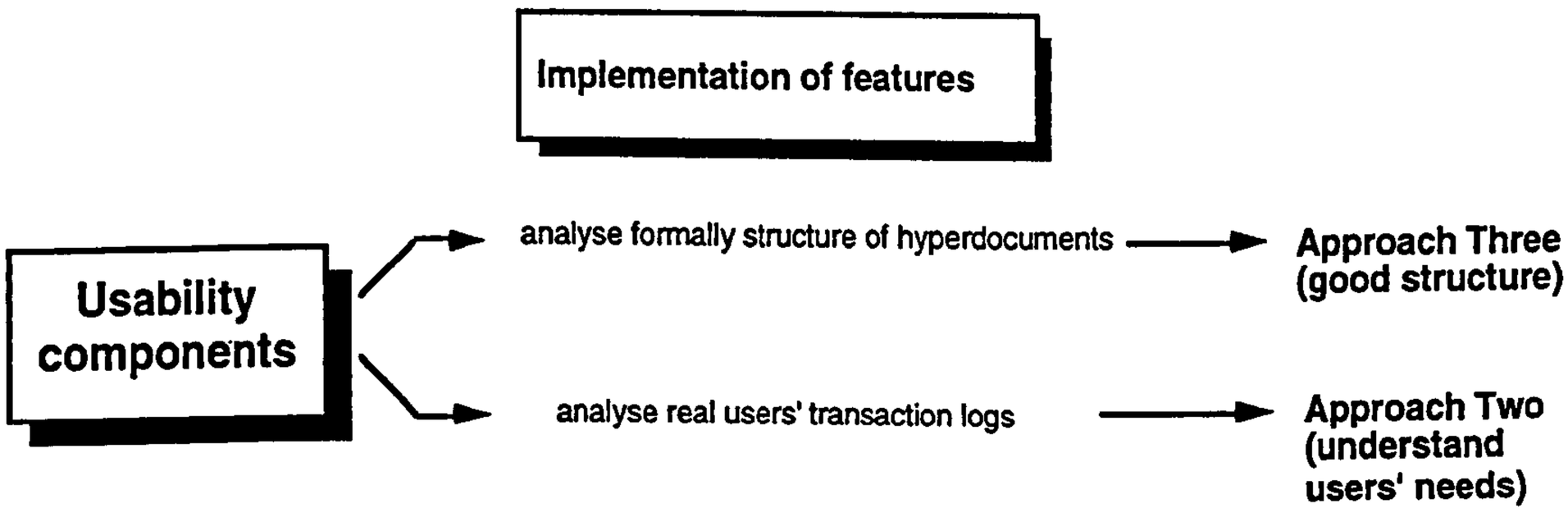


Figure 9.3. Some features implemented in the usability components in HyperAT

Informal evaluation of HyperAT conducted in chapter 8 indicates that it is a usable and useful tool. This thesis has shown how the authoring as well as usability features implemented in HyperAT can help designers manage the complexity of the design and validation processes of hypertext authoring without getting "lost" (§7.3).

This thesis has argued for and built a practical authoring tool to demonstrate that collaborative efforts involving many disciplines are necessary to address the LIH problem.

9.3.7 General contribution

The research methodology adopted in this thesis recruited all user-centred methods to ensure that well-structured, usable hypertexts are produced. As discussed in §2.4.1, these user-centred methods used in the methodology focus on the activities in the *Design Stage* of a "typical" user-centred design cycle. In fact, when discussing the approach taken in this thesis with experts (for example, Patricia Wright;*etc.*), they pointed out that the approach taken has great merit in the design of interactive systems even if no one ever got "lost". Because the work carried out is situated in well-established literature, this thesis argues that the design and usability issues investigated for the design of hypertext, and the web in particular can also be applicable to the design of general interactive systems.

This thesis has also shown that the contributions made extend beyond just addressing the LIH problem in hypertexts, but has made inroads into knowledge in the various disciplines in human-computer interaction, cognitive psychology and software engineering from which solutions were sought.

9.4 Lessons learned from software practices experience

In this thesis, HyperAT was being developed as a specific computer science contribution to help designers build well-structured hypertexts. In this section, the author would like to relate her experience involved in selecting, learning and using a programming language for the building of a prototype hypertext, and the building of HyperAT. How the author arrived at the decisions was a combination of problem-solving skills, intuition and circumstantial reasons.

- **Design of a prototype hypertext**

  When I[1] started, I was working on a Mac IIx and a natural choice for a programming language was HyperCard. Because of the features provided by HyperCard and the many good books written using HyperCard, I was able to build a simple prototype hypertext quite rapidly. Like good software engineering practice, HyperCard allowed the breaking down of corpus of materials into parts in the form of stacks and cards.

  However, it became very tricky to program if I wanted to build a more sophisticated system using HyperCard. Because of this, I needed to look for another more powerful programming language to build HyperAT.

---

[1]The author has decided to use the subject "I" when writing the remaining sections of this chapter to personalise the experience and lessons learned when doing this PhD.

- **Design of HyperAT**

  Because the hypertext structure was complex, I needed a programming language that was flexible enough for me to represent that structure. I was faced with 3 possibilities: Common Lisp, Prolog or C++. Finally after much careful investigation and deliberation, I decided on Common Lisp instead of Prolog or C++. Java was not a consideration in early 1995 since it was still then an exploratory language. However, since HyperAT was a research tool, and not a commercially-viable product, Common Lisp would still be preferred for the following reasons:

  i. There are two distinctive qualities in Lisp that make it stand out (Graham, 1994): Common Lisp can be tailored to suit the program being written. Common Lisp itself is a Common Lisp program, and Common Lisp programs can be expressed as lists, which are Lisp data structures.

  ii. Common Lisp encourages the bottom-up design, that is, changing the language to suit the problem. Bottom-up design yields these advantages: smaller and more agile programs are built; codes can be re-used; and programs are easier to read.

  iii. Another practical and important consideration is that with Common Lisp, I could discuss with people in the School of Computing Science and get the technical support whenever I needed it.

9.5 Interdisciplinary working experience

I had this unique opportunity of working together with another PhD student, Cécile Rigny, at Middlesex University. Because she was interested in testing her prototype Common Lisp tool for running cognitive models, and I was interested to find out how designers could be helped to understand end-users better, it seemed natural that we worked together. This has proved to be a fruitful and enriching experience. Not only am I proposing a multi-disciplinary approach in this thesis to address the LIH problem, I was also experiencing what it meant to work together in a team.

When we started, we were very careful to define the joint work and the boundaries of our individual PhD theses. After we decided on the area to work together, we then brainstormed on how we should combine our knowledge and expertise. We were aware that both of us coming from different backgrounds and disciplines had something individual and of value to offer, and had very different ways of looking at a practical question.

Working together with Cécile has been an enjoyable and enriching experience. We spent many hours together discussing and writing papers. The efforts involved in working together to achieve a better understanding of the problem area were rewarding. Monk (1995) uses the analogy of travel to describe the advantages and drawbacks of multi-disciplinary approaches to research. Because travel not only broadens the minds of the individual travellers, it also enriches the culture they return to. I, too, am enriched and it has helped me to be more open in my approach to address the LIH problem.

9.6 Limitations

One always faces a dilemma when deciding on the area and the directions taken in doing a PhD; it is like trying to eat an elephant in the sense that there is so much one would like to investigate, and so much that deserves to be studied. Naturally opinions differ with respect to what should and should not be presented (Lindgaard, 1994).

§9.6.1 discusses the circumstantial limitations surrounding the work. §9.6.2 acknowledges the current conceptual limitations and the work it documents.

9.6.1 Circumstantial limitations on the work

Deciding on what programming language and environment on which to base my work been a daunting as well as a challenging experience. I started exploratory work on the Mac IIx for about eight months, trying out HyperCard, Prolog, C++, HTML, Common Lisp. Being a PC user prior to starting this research added a great deal of stress in the initial eight months of the investigation. It took me a while to decide on a programming language for the building of a prototype hypertext, and the building of HyperAT. From hindsight, I wish I had shortened the time taken, and did more quality work in these initial, stressful eight months.

Switching to the PowerPC Mac Performa 5200 after eight months also presented some inconveniences in terms of familiarising with the new machine, as well as having to solve compatibility problems when switching to software that did not work well with the PowerPC, for example, Common Lisp version 2.01. Because of memory allocation problems, it was inevitable that I had to switch to version 3.9. This also gave me some problems because version 3.9 is a beta release and therefore, some features are buggy, for example, the "save-as" function did not work. Because I started programming in version 2.01, certain changes had to be made to the codes since some functions supported in version 2.01 were not supported in version 3.9.

However, the benefits outweighed the minor inconveniences caused. Programs ran faster and they did not crash due to memory problems. Netscape 2.01 could be opened simultaneously with version 3.9. This is an important feature since Netscape is needed to provide the platform for testing the hypertext created in HTML using HyperAT.

9.6.2 Current conceptual limitations

Because this thesis takes a multi-disciplinary approach to address the LIH problem, it entails work into many "strands" of investigations. It is a useful strategy in trying to obtain a complete understanding of the issues surrounding the LIH problem, which is a major contribution of this thesis. As suggested in Gestalt theory, this productive problem-solving approach generated many new ideas that would otherwise be missed out if this thesis focused on just one aspect of the problem.

However, the main disadvantage of this approach is that owing to time constraints, I was not able to go in-depth into the areas investigated. I have just managed to touch the tip of the

iceberg of feasible solutions. Each of the multi-disciplinary approaches warrants a survey and critical appraisal of the design issues concerned. With any literature survey, it becomes outdated almost as soon as it is written. Likewise in this thesis, though the surveys provided a solid foundation to the diversity of ideas generated from the multi-disciplinary approaches, there are undoubtedly many appropriate papers published since the surveys were carried out, and many which have not been included.

If I were to start this thesis again, I would have concentrated the LIH problem on the web, and not also on [2]standalone hypertexts. The multi-disciplinary approaches would still apply but the problem space to investigate the LIH problem would be much narrowed, and a more detailed investigation could be carried out. Of course, from hindsight all these seemed logical. But unless one does it, one is not aware of the problems, constraints and issues involved.

However, this is not to say that this thesis does not serve its purpose. In the recent Hypertext'97 conference attended, it was generally felt among hypertext researchers that more hypertext research needed to be made relevant to the web community. This is because the web is too relevant and important to be ignored, a decision that left the hypertext veterans regretful of their decision in 1991 when they rejected the paper on the web submitted by Tim-Berners Lee and colleagues at the Hypertext'91 Conference. The direction taken in this thesis to address the LIH problem is novel and is in agreement with the current concern among the hypertext research community to search for solutions from diverse disciplines and apply them to the web. Birch (1997) of the Engineering and Physical Science Research Council (EPSRC) in the panel session on "Time and the Web" held at Staffordshire University in June 1997, urges researchers to step back and address more fundamental issues on the web so as to provide a theoretical base to developers for the implementation of web-based systems.

This thesis acknowledges that the work presented could be refined, developed, extended and approached in an alternative manner.

9.7 Future work

This section outlines some interesting directions for further work which consider relevant issues that are beyond the scope of current work, but merit further investigation. There are some interesting things I wish I had done but was unable to do so at the time of writing this thesis, owing to time constraint or lack of foresight. Some of these concern further work on HyperAT (see §9.7.1), discussing its potential future development to correct its deficiencies as well as to extend its functionality. The rest are concerned with further investigations into the

---

[2]Standalone hypertexts refer to hypertexts with specific contents for specific group of end-users. They are self-contained, and can also be written in HTML.

LIH problem with respect to the multi-disciplinary approaches undertaken in this thesis (see §9.7.2).

9.7.1 Further work on HyperAT

Further work on HyperAT concerns further enhancements to HyperAT, and further evaluations of HyperAT and the approach it represents.

- **Further enhancements to HyperAT**

  As an experiment in collaborative efforts involving many disciplines, HyperAT demonstrates that useful ideas drawn from multi-disciplinary approaches can be effectively operationalised in an authoring tool to aid designers manage the complexity of the design process.

  However, as a *practical authoring tool*, HyperAT may be too restrictive and not offering maximum benefits to designers in the following features as described in the authoring components:

  i. node text needs to be entered in the HTML format (§7.3.3.1). Although an HTML editor is available, HyperAT assumes designers have some knowledge of HTML;

  ii. only hierarchical and cross-referenced links are captured (§7.3.3.1);

  iii. generated maps only show global and fisheye views of the hypertext structures, but do not give more information about the nodes (§7.3.3.2);

  iv. web page layout does not include the URL address and overview of related topics may not be so easily visualised without providing a graphical representation (§7.3.3.3);

  v. maintenance support provided by HyperAT could be improved by not just automatically updating the data structures (§8.2.4), but also by executing the affected programs; and

  vi. name of a new node needs to be entered at the point of creation, and this is sometimes not feasible since designers may themselves be uncertain what information be included in the new node (§7.3.3.1).

  As an *analytic research tool*, HyperAT can be further improved by enhancing the following features as described in the usability components:

  vii. checks on structural inconsistencies do not consider pages with no links but are present in the website (§7.3.4.1);

  viii. designers are not provided with design recommendations after running checks on structural complexity (§7.3.4.1);

ix. end-users' browsing pattern and goals achieved parsed by HyperAT are generally based on a goal-oriented approach, however, there is no in-depth analysis of the implications of the results (§7.3.4.2); and

x. transaction log files have to be copied from the server, converted to Lisp files before being parsed by HyperAT (§7.3.4.2).

As a recommendation for future work, the following propositions are made to address the current limitations of HyperAT:

1. Instead of having to type in the node text in HTML format, designers can type in text by underlining headings and links in different colours. HyperAT can parse the text and generate a HTML version of the text. Another possible development is to read off a text document, putting in the relevant tags.

2. More complex link relationships could be captured, for example, whether the links are strong, weak or weighted. This is because the weights of links would reflect the semantic relationships between corresponding nodes. These types of information can be used together with information on end-users' browsing pattern, which can be further developed in HyperAT to make the hypertext more adaptable to end-users' needs.

3. To make the maps even more useful, not only can designers view the structure of hypertext, they can also read and edit the attributes of the nodes.

4. When I started with the implementation of HyperAT, Common Lisp was a good choice since I was developing a research tool and not a commercially viable authoring tool. A research challenge will be to make HyperAT more usable and interesting in the real web environment. Applets can be used to represent the global as well as fisheye views of the hypertext structure. Another important inclusion is the generation of the URL address of the page, so that anyone who wishes to print a copy of the web page need not copy the URL address.

5. Changes made could be dynamically captured.

6. A "rename" option could be provided to change the name of the node, as well as update all links.

7. Structural consistency checks should also include parsing a website to pick out those pages that do not have links to any other pages.

8. HyperAT should generate a list of recommendations based on the results of analysing the complexity of the hypertext structure. Based on these recommendations, designers can modify the hypertext.

9. After analysing end-users' browsing pattern and goals achieved, HyperAT should make comparisons and make design recommendations. A feature could be

incorporated to analyse not only the browsing pattern, but the structure of end-users' interaction for the different types of hypertexts. The aim is to develop, demonstrate the utility of, and mechanisms for recognising structure in end-users' interaction with a hypertext, not just from designers' point of view.

10. A program could be written to link HyperAT to the server to read off the log files so that parsing of the files could be carried out without having to convert them to Lisp files.

- **Further evaluations of HyperAT**

As discussed in chapter 8, it is outside the scope of this thesis to carry out an extensive evaluation of HyperAT. Instead an informal evaluation of HyperAT was carried out in §8.3 to gather feedback on the usability and usefulness of HyperAT's authoring and usability environments. Figure 9.4 gives an overview on how an extensive evaluation of HyperAT approached from two perspectives could be carried out:

i. From the perspective of the hypertext designer, that is, user of HyperAT

The aim is to obtain feedback from hypertext designers regarding the usefulness and usability of HyperAT. The question to ask is: is HyperAT effective in helping designers build usable web documents without themselves getting "lost"?

ii. From the perspective of the hypertext end-user, that is, user of the web document produced by HyperAT

The aim is to obtain feedback from hypertext end-users on the usability of the web documents produced by HyperAT. In other words, is the web document produced by HyperAT effective in helping end-users navigate without getting "lost"?

To evaluate the usefulness and usability of HyperAT, expert and observational evaluation techniques could be used. To evaluate the usability of web documents produced by HyperAT, two forms of real user testing such as experimental and survey evaluation techniques could be selected.

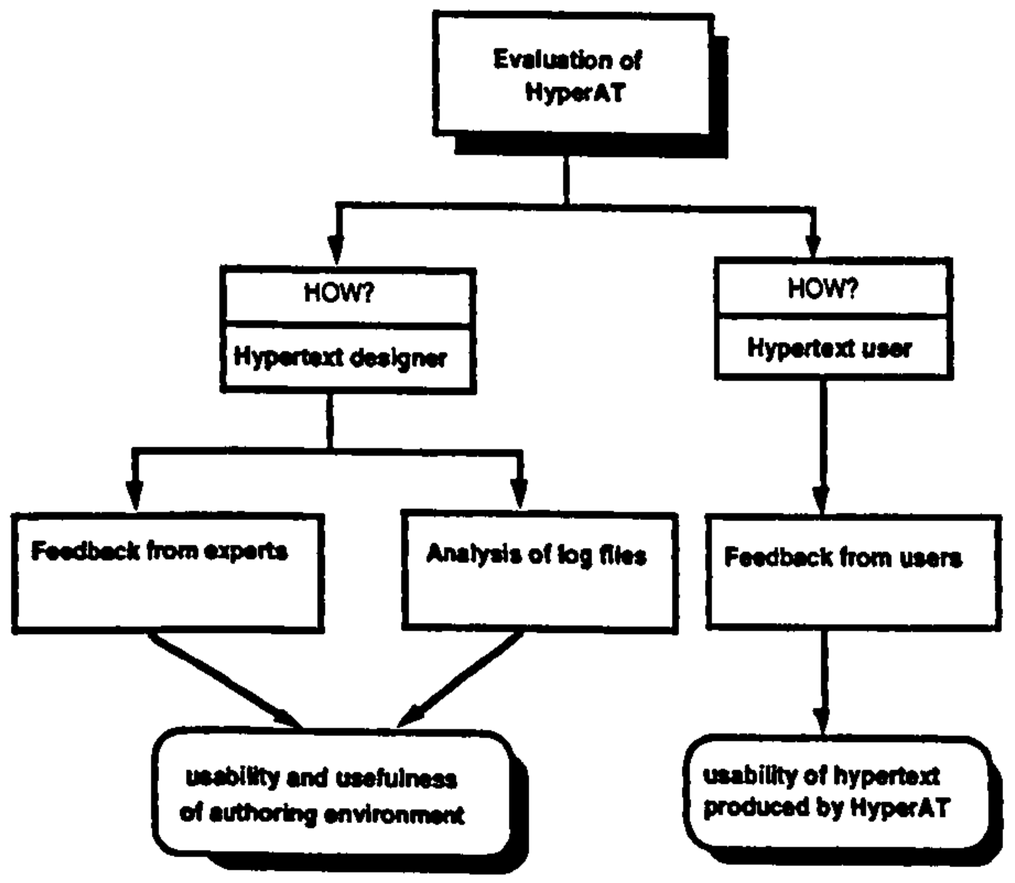


Figure 9.4. Evaluation of HyperAT from two perspectives

9.7.2 Further investigations into the "lost in hyperspace" problem

The author thesis believes that much interesting work could be carried out to refine and strengthen the investigations being carried out into the different disciplines to address the LIH problem:

- **Design principles**

  In chapter 3, a list of design principles for creating good hypertext was drawn up. Suggestions were given as to how these design principles could be implemented. At the outset of the design of the hypertext prototype, the intention was to build a prototype quickly, therefore the decision was to incorporate only some ideas as suggested by the design principles (§3.3.3). Subjects' responses were not too positive about the prototype. Many factors could have affected their satisfaction too, for example, the interface was too simple and not much fun to use, perhaps due to a flaw in the design decision as well as a technical limitation of HyperCard (version 2.01) resulting in less fanciful interface when compared with current hypermedia products, *etc.*

  From §3.5.2, the author also learned from designing the prototype that knowing and implementing design principles are two separate issues. The chapter discusses the knowledge of good design principles, but not how to go about implementing these principles.

  Instead of using HyperCard version 2.01, it might be worthwhile to use a more powerful, advanced programming languages such as Toolbook, MacroMedia Director, *etc.* to build the prototype. This is to ensure that any criticism about the prototype is about the implementation of the design principles, and not due to the limitations of the programming language.

  Future work could also include the formulation of a design methodology, that is, how to implement the design principles to structure and to guide the process of design when followed in sequence. Although the ideal design methodology may not be realisable, it may provide insightful and useful information on how designers can go about designing good hypertext. This could be a step towards a more systematic approach to designing hypertext.

- **Design guidelines**

  As discussed in §3.6.3, the questionnaire was not validated since the intention of this thesis was to get quick results and impressions about the design guidelines proposed.

  More testing on the questionnaire should be carried out to establish external, construct or predictive validity of the questionnaire, that is, the design guidelines. Future work could also involve coming up with useful metrics to detect the LIH problem, and thus the overall usability of hypertext. These metrics could provide a basis of comparison among hypertexts.

- **Task analysis**

  In chapter 4, a combination of task analysis and task decomposition techniques was employed to understand the task browsing represented as cognitive task diagrams. To validate and refine the cognitive task diagrams for browsing, preliminary work was conducted. Two typical end-users' behaviour and actions using ACM's *Hypertext-on-Hypertext* (ACM, 1989) to complete some exercises associated with browsing were recorded. A video protocol of the end-users' interaction was conducted but unfortunately the quality of the video was not very good. Although I was able to interview end-users after the experiment and to ask them to explain step-by-step why certain decisions and actions (for example, moving to the next screen, going to "home" screen, moving to the "table of contents", *etc.*) taken while using the hypertext, it would have been better if the video had also captured the end-users' interactions with the computer clearly. This is because the end-users could not remember all the steps taken. Future work includes improving the technique for capturing and recording end-user interactions. This would help to understand end-users' reasoning and learning processes better such as when they choose to perform certain actions. The feedback could be used to modify and refine the task diagrams accordingly, a limitation of the work reported in this thesis.

  As discussed previously, this task analysis approach is only one way to understand end-users' browsing needs and behaviour. Future work should include exploring other areas. An interesting area to investigate is to explore the "foraging theory", a theory developed within biology for understanding the opportunities and forces of adaptation. Pirolli and Card (1995) advocate using this theory to help designers understand human adaptations for gaining and making sense out of information. They think that it could also help in understanding end-users' browsing behaviour.

- **Executable user modelling**

  A preliminary and innovative investigation into using cognitive user modelling techniques to perform usability testing on web documents was jointly carried out with Cécile Rigny from the School of Computing Science at Middlesex University. CUM-DesTool is a tool for running executable user models to pinpoint potential usability problems for interactive systems (Rigny, 1997), so that hypertext designers can make informed decisions based on recommendations suggested for improved design. Further work involves incorporating CUM-DesTool into HyperAT to automate the usability evaluation of web documents produced by HyperAT. Preliminary work carried out to investigate incorporating CUM-DesTool into HyperAT demonstrates that it is feasible. Given that HyperAT generates a formal description of the hyperdocument (using HTML format), it is possible in principle to use the description as inputs into CUM-DesTool. All that needs to be done is to define and implement an interpreter to read and parse the description (see figure 9.5). Since CUM-DesTool and HyperAT are both written in Common Lisp, combining the two tools is a matter of programming.

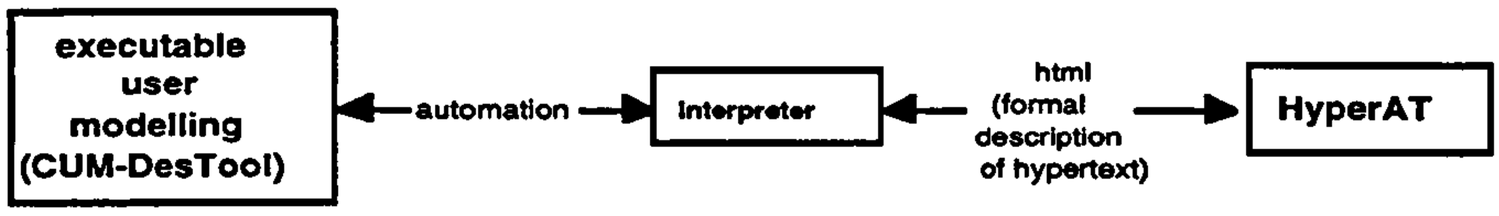


Figure 9.5. Implementing executable user models into HyperAT

- **Conceptual design stage**

  In non-human user testing, CUM-DesTool was used to interpret the cognitive task diagrams for browsing, to produce executable user models. The performance of the executable user model for browsing was compared with real users (§4.3.4). There is, thus, great potential in using executable user models for rapid prototyping and testing, without requiring real users' attendance. The Conceptual Design stage of the development lifecycle can be improved by incorporating cognitive task graphs and executable user models. However, this claim is not based on any experimental or computational results.

  Further work involves investigations to find out if the improved Conceptual Design stage using cognitive task graphs and executable user models may indeed be a more cost-effective way of designing in terms of time spent by designers. More work should be done to verify if this is true. If it is, then it opens up a challenging, exciting area in exploring the use of executable user models as design aids.

- **Hypertext structuring**

  In chapter 5, the relationships between subjects' representation of the structure of ACM's *Hypertext-on-hypertext* with factors such as subjects' experience, subjects' satisfaction, subjects' perceived effectiveness, subjects' feeling "lost" and subjects' performance were studied. However, the study did not investigate whether the interface of the hypertext has an influence on subjects' representation of the hypertext.

  A promising area for future work is to study the impact of end-user's representation of the structure of hypertext, and to examine how it might be transferred to, or affected by screen presentation. In other words, do end-users' representation of hypertext interfere with their navigation? If the answer is positive, how then can they be helped? Or, does the screen display interfere with end-users' representation of the hypertext structure? If the answer is "yes", then what would be a good interface?

  Experiment 5.1 identified hierarchical-like structures as commonly used structures to represent mainly educational hypertext, but not hypertexts in general. This experiment could be extended to search for optimal structures for different functional purposes such as information sources, retrieval systems, *etc.* A parallel "strand" of investigation is to look at it not only from the designer's perspective but also from the end-user's perspective.

9.8 Conclusion

To begin with, the work done and the opportunities arising from the work at the time of writing this thesis was summarised. This chapter reflected on how this thesis has met its objectives, the successes and the pitfalls encountered. Lessons learned from software experience practices in selecting a suitable programming language, design of prototype hypertext and HyperAT were recorded. The interdisciplinary working experience gathered while working on this thesis was described. The findings of the investigation, the limitations and the inevitable things not done at the time of writing this thesis were highlighted. This was followed by propositions on future work which consider relevant issues that are beyond the scope of current work, but merit further investigation. Several of these concern HyperAT, and its potential future development to correct its deficiencies as well as to extend its functionality. Others include more work into the various disciplines.

Chapter 10

Conclusion

This thesis has made several theoretical and practical contributions to address the LIH problem in hypertext, and the web in particular. It argues for a move from treatment to prevention, from treating the end-users' symptoms to avoiding the bad design. As a result, this thesis has generated many "strands" of investigation and novel ideas by assuming a multi-disciplinary approach to the LIH problem, resulting in contributions made to hypertext and various disciplines. Monk and Gilbert (1995) claim that "inter-disciplinary research can only be effective when it involves the creative juxtaposition of different approaches around a specific problem, so that each can shed its own light on the issues". By integrating and developing the approaches suggested to address the LIH problem, this thesis proposes a methodology and implements an authoring tool for the design and building of usable web documents. This has significant implications. This thesis has shown that research efforts need not be focused on a particular discipline to solve a problem. It has argued for collaborative efforts involving many disciplines that may be necessary to solve complex problems, of which the LIH is an example. This thesis has also shown how these approaches can be applied in a creative way to address the LIH problem in hypertext.

Because the work carried out in this thesis is based on well-established literature in hypertext, human-computer interaction, cognitive psychology and software engineering, the emergent ideas that come out from the investigations can also be adapted to address design issues for general interactive systems.

To summarise, the main features of this thesis are:

- it proposes a multi-disciplinary framework to eliminate, or ameliorate the LIH problem in hypertext;

- it provides a framework to draw up design principles for hypertext, and a methodology for evaluating hypertext;

- it presents a systematic, engineering approach to understand end-user behaviour and the common tasks they perform when navigating hypertext;
- it suggests both analytic and empirical methods of evaluation for hypertext;
- it focuses on structural design issues to ensure the building of well-structured hypertext; and
- it identifies useful and helpful features in future authoring tools.

The author acknowledges that though HyperAT, a prototype authoring tool implemented in the course of this research, may not replace any commercial authoring tool since commercial tools are concerned with solving different problems, for example, increasing sales, *etc.* rather than usability. This thesis, however, has demonstrated that interesting design and usability ideas can be realised in a practical authoring tool to address the LIH problem. The benefits that can be derived from using an efficient authoring tool are tremendous: savings in terms of money and time; contributions in the design of better hypertexts which can be tested, further built upon by other researchers. Nielsen (1996) once said that "much existing hypertext research could benefit [not only researchers] but web designers as well as designers of web softwares". It is hoped that the work undertaken in this thesis would stimulate further research and discussion in addressing more fundamental issues surrounding the LIH problem in hypertext, and the web in particular.

Section V

Supporting materials

List of acronyms and abbreviations

ACM	Association for Computing Machinery
APCHI	Asia Pacific Conference on Computer Human Interaction
CERN	European Center for Nuclear Physics Research
CCT	Cognitive Complexity Theory
CLG	Common Language Grammar
CP	Cognitive Psychology
CS	Cognitive Science
Cum-DesTool	Cognitive User Modelling Tool
gIBIS	graphical Issue Based Information System
GOMS	Goals, Operators, Methods and Selection Rules
HCI	Human-Computer Interaction
HT	Hypertext
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HyperAT	Hypertext Authoring Tool
ICS	Interacting Cognitive Subsystems
IEE	The Institution of Electrical Engineers
IP	Internet Protocol
KMS	Knowledge Management System
LIH	"lost in hyperspace"
MHP	Model Human Processor
NLS	oN-Line System
PC	Personal Computer
PUMs	Programmable User Models
SE	Software Engineering

TA	Task Analysis
TAG	Task Action Grammar
TAL	Task Action Language
TD	Task Decomposition
URL	Uniform Resource Locatot
WWW/Web	World Wide Web

List of tables

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

Chapter 7

Chapter 8

Chapter 9

List of figures

Chapter 2

Chapter 3

Chapter 4

Chapter 8

Chapter 9

Bibliography

ACM (1989), "ACM Hypertext-on-Hypertext," ACM Press Database and Electronic Products Series.

Addison, M. and Dudman, K. (1995), "Organisational issues for managing multimedia information," *Proceedings of 6th International Conference on Database and Expert Systems Applications*, pp. 241–246.

Akscyn, R.M., McCracken, D.I. and Yoder, E.A. (1988), "KMS: A distributed hypermedia system for managing knowledge in organisations," *Communications of the ACM*, 31(7), pp. 820 – 835.

Allinson, L. and Hammond, N. (1989), "A learning support environment: The Hitchhikers' Guide," *Hypertext: Theory into practice*, Intellect Books, pp. 62-74.

Am, O. (1994), "Cyberspace and the structure of knowledge," *http://www.hsr.no/onar/ess/cyberspace_and_the_structure_of_knowledge.html*.

Anderson, R., Carroll, J. McGrew, J., Grudin, J. and Scapin, D. (1990), "Task analysis: the oft missed step in the development of computer-human interfaces; its desirable nature, value and role," *Report on panel discussion at Interact'90*, pp. 1051 – 1054.

Annett, J. and Duncan, K.D. (1967), "Task analysis and training design," *Occupational Psychology*, 41, pp. 211 – 221.

Babiker, E. M., Fujihara, H., and Boyle, C.D.B. (1991), "A Metric for Hypertext Usability," *1991 Conference Proceedings of ACM 9th Annual Conference on Systems Documentation*, ACM, pp. 95 – 104, U.S.A.

Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S. (eds.) (1995a), "Design and evaluation," *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 73 – 91, U.S.A.

Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S. (eds.) (1995b), "Hypertext and Multimedia," *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 833 – 842, U.S.A.

Barnard, P. (1987), "Cognitive resources and learning of human-computer dialogues" in Carroll, J.M. (ed.), *Interfacing Thought: Cognitive aspects of Human-Computer Interaction*, MIT Press.

Barnard, P.J. and May, J. (1993), "Cognitive modelling for user requirements," in Byerley, F., Barnard, P.J. and May, J. (eds.), *Computers, Communication and Usability: Design issues, research and methods for integrated services*, North-Holland series in Telecommunication, Elsevier, pp. 101 – 145.

Bates, M.J. (1989), "The design of browsing and berrypicking techniques for the on-line search interface," *Online Review*, 13(5), pp. 407 – 431.

Batley, S. (1989), "Visual information retrieval: browsing strategies in pictorial databases," *PhD Thesis*, as cited in **McAleese, R.** (1993), "Navigation and browsing in hypertext," *Hypertext: Theory into Practice*, pp. 1 – 38, Intellect Books, U.K.

Batsy, G.P., McRae, S. and Timmer, P. (1997), "Against generalising hypermedia navigation," *Adjunct proceedings of HCI'97*, pp. 37 – 39.

Beekman, G. (1995), *HyperCard 2.2 in a hurry : The fast track to Multimedia*, Wadsworth, U.S.A.

Beeman, W., Anderson, K., Bader, G., Larkin, J., McClard, A., McQuillan, P. and Shields, M. (1987), "Hypertext and pluralism: from lineal to nonlineal thinking," *ACM Proceedings of Hypertext '87*, pp. 67 – 88, U.S.A.

Benest, I. (1990), "A hypertext system with controlled hype," in McAleece, R. and Green, C. (eds.),*Hypertext: State of the Art*, Intellect Books, pp. 52 – 63.

Berners-Lee, T. (1995), "Style Guide for Online Hypertext," *http://www.w3.org/pub/www/provider/style/all.html.*

Bernestein, M. (1988), "The bookmark and the compass: Orientation tools for hypertext users," *ACM SIGOIS Bulletin*, Vol. 9(4), pp. 34 – 45.

Berk, B. and Devlin, J. (eds.) (1991), "Why hypertext?" in *Hypertext/Hypermedia Handbook*, Armadillo Associates, pp. 9– 11.

Birch, N. (1997), "Time and the Web panel session – issues from EPSRC," Panel presentation at Time and the Web, Staffordshire University, *http://www.soc.staffs.ac.uk/seminars/web97/*

Blandford, A. E. and Young, R. M. (1995), "Applying programmable user models to real design problems,"*Amodeus Project DocumentD10, MRU-APU.*

Botafogo, R.A., Rivlin, E. and Shneiderman, B. (1992), "Structural analysis of hypertexts: Identifying hierarchies and useful metrics," *ACM Transactions on Information Systems*, 10(2), pp. 142 – 180.

Booth, P. (1989), *An introduction to Human-Computer Interaction*, Lawrence Erlbaum & Associates, Hillsdale, NJ.

Boyle, C. and Snell, J. (1988), "Intelligent navigation for semi-structured hypertext documents," in McAleece, R. and Green, C. (eds.),*Hypertext: State of the Art*, Intellect Books, pp. 28–39.

Brown, C. (1988), *Human-computer interface design guidelines*, Ablex.

Brown, P.J. (1990), "Assessing the quality of hypertext documents,"*Proceedings of ECHT'90*, pp. 1 – 12, U.K.

Brown, P. (1987), "Turning ideas into products: The Guide System," *ACM Proceedings of Hypertext '87*, pp. 33 – 39.

Brusilovsky, P. (1996), "Methods and techniques of adaptive hypermedia," *User Modelling and User Adapted Interaction*, 6.

Buckley, P. and Johnson, P. (1987), "Analysis of communication tasks for the design of a structured message system," *People and Computers III: Proceedings of 3rd Conference British Computer Society*, pp. 29 – 40.

Buckingham-Shum, S. (1996), "The missing link: Hypermedia usability research and the web," *ACM SIGCHI Bulletin*, 28(4), pp. 68 – 75.

Busch, D. (1997), "Fingers on the pulse," *Internet World*, July/August, pp. 88 – 98.

Bush, V. (1945), "As We May Think," *Atlantic Monthly*, 7, pp. 101 – 108.

Campagnoni, F. R. and Ehrlich, K. (1989), "Information retrieval using a hypertext-based help system," *ACM Transaction Information Systems*, 7(3), pp. 271 – 291.

Campbell, B. and Goodman, J.M. (1988), "HAM: A general purpose hypertext abstract machine," *Commuincations of the ACM*, 31, 7, pp. 856 – 861.

Canter, D., Rivers, R. and Storrs, G. (1985), "Characterising user navigation through complex data structures," *Behaviour and Information Technology*, 4(2), pp. 93 – 102, as cited in **McAleese, R.** (1993), "Navigation and browsing in hypertext," *Hypertext: Theory into Practice*, pp. 1 – 38, Intellect Books, U.K.

Card, S.K., Moran, T.P. and Newell, A. (1979), "The keystroke-level model for user performance time with interactive systems, " *Report SSL-79-1*, AIP Memo 122, Palo Alto Research Centre, California.

Callagher, C. A. (1974), "Perceptions of the value of a management information system," *Academy of Management Journal*, 17, 1, pp. 46 – 55, as cited in **Chin, J.P., Diehl, V.A. and Norman, K.L.** (1988), "Development of an instrument measuring user satisfaction of the human-computer interface," *CHI'88 Proceedings*, pp. 213 – 218.

Carlson, P.A. (1990), "The rhetoric of hypertext,"*Hypermedia*, 2(2), pp. 109 – 131.

Carlson, P.A. (1989), "Hypertext and Intelligent Interfaces for Text Retrieval," *The Society of Text : Hypertext, Hypermedia and the Social Construction of Information*, pp. 59 – 76, MIT Press, U.K.

Carmel, E., Crawford, S. and Chen, H. (1992), "Browsing in hypertext: A cognitive study," *IEEE Transactions on Systems, Man and Cybernetics*, 22(5), pp. 865 – 884.

Carroll, J. M. (1982), "The Adventure of getting to know a Computer,"*IEEE Computer*, 15(11), pp. 49 – 58.

Chin, J.P., Diehl, V.A. and Norman, K.L. (1988), "Development of an instrument measuring user satisfaction of the human-computer interface," *CHI'88 Proceedings*, pp. 213 – 218.

Clibbon, K. and Callaghan, M. (1996), "Beyond Halasz's Hypertext Research Agenda – The WWW?" in Buckingham Shum, S. and McKnight, C. (eds), Special issue of *International Journal of Human-Computer Studies on 'Web Usability'*.

Cockburn, A. and Jones, S. (1995), "Trails, trials and tributions: unravelling navigational problems in the world-wide web," *Proceedings of the 5th Workshop on Information Technologies and Systems (WITS '95)*, Netherlands.

Colborne, G. (1996), "D3 tools review: Web development," *Web Development Tools Review*, October 1996, pp. 16 – 25.

Coleman, W.D., Williges, R.C. and Wixon, D. R. (1985), "Collecting detailed user evaluations of software interfaces," *Human Factors Society 29th Annual Meeting*, pp. 240 – 244, as cited in **Nielsen, J.** (1993), *Usability Engineering*, AP Professional, U.S.A.

Conklin, J. (1987), "Hypertext : An Introduction and Survey," *IEEE Computer*, pp. 17 – 41.

Coutaz, J., Nigay, L. and Salber, D. (1993), "The MSM Framework: A design space for multi-sensory-motor systems," in Bass, L., Gornostaev & Unger, C. (eds.), *Lecture Notes in Computer Science 753 (EWHCI'93) Selected Papers*, pp. 231 – 241, Springer-Verlag.

Cove, J.F. and Walsh, B. C. (1988), " Online text retrieval via browsing," *Information Processing and Management*, 24(1), pp. 31 – 37.

Creech, M.L. (1996), "Author-oriented link management," *Proceedings of the 5th W3 Conference*, pp. 1015 – 1025

Crespo, A. and Bier, E. (1996), "WebWriter: A browser-based editor for constructing web applications," *Proceedings of the 5th W3 Conference,*, pp. 1291 – 1306.

De Bra, P. (1996), "Hypermedia structures and systems," *http://www.win.tue.nl/win/cs/is/debra/cursus/*

Diaper, D. (1989), "Task Analysis for Knowledge Descriptions (TAKD): the method and an example, " in Diaper, D. (ed.),*Task Analysis for Human-Computer Interaction.*, Ellis Horwood Ltd., pp. 108–159.

Dickerson, K.R. and Hedman, L. R. (1993), "Usability standards," in Byerley, P. F., Barnard, P. J. and May, J. (eds.), *Computers, Communications and Usability*, North-Holland Studies in Telecommunication.

Dillon, A. (1994), *Designing usable electronic text: Ergonomic aspects of human information usage*, Taylor & Francis.

Dix, A., Finlay, J., Abowd, G. and Beale, R. (1993), *Human-computer Interaction*, Prentice-Hall.

Dorling-Kindersley (1994), *The Ultimate Human Body*, Multimedia CD ROM.

Dvorak, R.R. (1995), *Text into hypertext books: An evaluative investigation into reader-centred link structures for hypertext*, PhD Thesis, Queen Mary and Westfield College of London of University.

Eason, K.D. and Harker, S. (1991), "Human factors contributions to the design process," in Shackel, B. and Richardson, S. (eds.), *Human Factors for Informatics Usability*, Cambridge: CUP.

Eason, K. D. and Harker, S. (1980), "An Open Systems Approach to Task Analysis," *Internal Report, HUSAT Research Centre*, Loughborough University of Technology.

Edwards, D.M. and Hardman, L. (1993), "'Lost in hyperspace': cognitive mapping and navigation in a hypertext environment," *Hypertext : Theory into Practice*, pp. 90-106, Intellect Books, U.K.

Elm, W. and Woods, D. (1985), "Getting lost: A case study in interface design," *Proceedings of the 29th Annual Meeting of the Human Factors Society.*

Eklund, J. (1995), "Cognitive models for structuring hypermedia and implications for learning from the world-wide web," *Ausweb95 The First Australian World Wide Web Conference.*

Engelhart, D.C. (1963), "A Conceptual Framework for the Augmentation of Man's Intellect," *Vistas in Information Handling Spartan Books*, 1, pp. 1 – 29, U.K.

Fillion, F.M. and Boyle, C.D.B. (1991), "Important issues in hypertext documentation usability," *Proceedings of the 9th ACM Annual International Conference on Systems Documentation*, SIGDOC'91, U.S.A.

Foley, J. and Wallace, V. (1974), "The art of natural graphic man-machine conversation," *Proceedings of the IEEE*, 62(4), pp. 462 – 471, as cited in **Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S.** (eds.) (1995a), "Design and evaluation," *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 73 – 91, U.S.A.

Foss, C.L. (1989), "Tools for reading and browsing hypertext," *Information Processing and Management*, 25(4), pp. 407 – 418.

Fox, P. (1996), "Spin your own web," *Computer Life*, October 1996, pp. 48 – 53.

Furnas, G.W. (1986), "Generalised fisheye views," *CHI'86 Proceedings*, pp. 16 – 23.

Gaines, B. and Shaw, M. (1984), *The art of Computer Conversation*, Prentic-Hall, as cited in **Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S.** (eds.) (1995a), "Design and evaluation," *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 73 – 91, U.S.A.

Garzotto, F., Paolini, P., Schwabe, D. and Bernstein, M. (1991), "Tools for designing hyperdocuments," in Berk and Devlin (Eds), *Hypertext/Hypermedia Handbook*, McGraw-Hill, pp. 179 – 207, U.S.A.

Garrett, L.N., Smith, K.E. and Meyrowitz, N. (1986), "Intermedia: Issues, strategies and tactics in the design of a hypermedia document system," *Proceedings of the first Conference on Computer-Supported Cooperative Work*, pp. 163 – 174.

Ginige, A., Lowe, D.B. and Robertson, J. (1995), "Hypermedia Authoring," *IEE Multimedia Winter*, pp. 24– 35.

Glushko, R.J. (1992), "Seven ways to make a hypertext project fail," in Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S. (eds.) *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 849 – 853, 1995, U.S.A.

Graham, P. (1994), *On Lisp: Advanced Techniques for Common Lisp*, Prentice-Hall.

Gould, J.D. (1988), "How to design usable systems," in Helander, M. (ed.) *Handbook of Human-Computer Interaction*, North-Holland: Elsevier, pp. 757 – 789.

Gould, J., Boies, S. and Lewis, C. (1991), "Making usable, useful, productivity-enhancing computer applications," *Communications of the ACM*, 34(1), pp. 74 – 85.

Halasz, F.G. (1987), "Reflections on NoteCards : Seven Issues for the Next Generation of Hypermedia Systems," *Proceedings of Hypertext '87*, pp. 345 – 365, ACM Press, U.S.A.

Hall, W., Carr, L. and Roure, D.D. (1995), "Linking the World Wide Web and Microcosm," *BCS Workshop on new directions in software development*, U.K.

Hansen, W. (1971), "User engineering principles for interactive systems," *AFIPS Conference Proceedings 39*, AFIPS Press, pp. 523 – 532, as cited in **Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S.** (eds.) (1995a), "Design and evaluation," *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 73 – 91, U.S.A.

Hardman, L. (1988), "Hypertext tips: Experiences in developing a hypertext tutorial," in Jones, D.M. and Winder, R. (eds.) *People and Computers IV*, Manchester, pp. 437 – 451.

Hardman, L. and Sharratt, B.S. (1990), "User-centred hypertext design: the application of HCI design principles and guidelines," *Hypertext: State of the Art*, Intellect Books, pp. 252 – 259.

Harold, E.R. (1996), "The comp.lang.java FAQ List," *http://sunsite.unc.edu/javafaq/javafaq.html*.

Harris, P. (1986), *Open guides to psychology: Designing and reporting experiments*, Open University Press.

Heckel, P. (1991), *The elements of friendly software design*, The new edition, Sybex, as cited in **Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S.** (Eds) (1995a), "Design and evaluation," *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 73 – 91, U.S.A.

Henninger, S., Haynes, K. and Reith, M.W. (1995), "A framework for developing experience-based usability guidelines," *DIS'95 Symposium on designing interactive systsems: Processes, practices, methods and techniques*, pp. 43 – 53.

Hewett, T. (1986), "The role of iterative evaluation in designing systems for usability," *People and Computers: Proceedings of 2nd Conference BCS HCI specialist group*, Cambridge University Press.

Hildreth, C. (1982), "The concept and mechanics of browsing in an online library catalogue," *Proceedings of 3rd National Online Meeting*, as cited in **McAleese, R.** (1993), "Navigation and browsing in hypertext," *Hypertext: Theory into Practice*, pp. 1 – 38, Intellect Books, U.K.

Horn, R. (1989), "Mapping hypertext: The analysis organisation and display of knowledge for the next generation of on-line text and graphics," Lexington Institute as cited in **Wray, R. E., Chong, R., Phillips, J., Rogers, Wlash, W. and Laird, J.** (1994), "Organising information in Mosaic: A classroom experiment," http://ai.eecs.umich.edu/cogarchO/mosaic-paper/wrayre-experiment-mosaic94.html.

Howard, S. and Murray, D.M. (1987), "A taxonomy of evaluation techniques for HCI," *Proceedings of Human Computer Interaction Interact'87*, Elsevier Science Publishers, pp. 453–459.

Howes, A. (1995), "An introduction to cognitive modelling in human-computer interaction," in Monk, A. and Gilber, N. (eds.), *Computers and People Series*, Academic Press.

InterNIC Directory and Database Administration (1997), "Meta-Search engines," *http://www.internic.net:80/tools/meta.html*

ISO (1990), "International Standards Organisation, Draft CD 9241 Part 11: Usability Statements," *ISO/TCI159/SC4/N186*, as cited in **Lindgaard, G.** (1994), *Usability testing and system evaluation: A guide for designing useful computer systems*, Chapman & Hall.

Jensen, J. (1997), "Between the style sheets: typographic control comes to the Web," *http://webreview.com/96/11/08/feature*

Johnson, B. and Shneiderman, B. (1991), "Tree-maps: A space-filling approach to the visualisation of hierarchical information structures," *IEE*, pp. 284 – 291.

Johnson, P. (1992), *Human Computer Interaction: Psychology, Task Analysis and Software Engineering*, McGraw-Hill.

Johnson, P., Diaper, D. and Long, J.B. (1984), "Tasks, skill and knowledge; task analysis for knowledge based descriptions," in *Proceedings 1st IFIP Conerence Human-Computer Interaction (Interact'84)*, pp. 499–503.

Jonassen, D.H. (1989), "Semantic network elicitation: tools for structuring hypertext," in McAleece, R. and Green, C. (eds.), *Hypertext: State of the Art*, Intellect Books, pp. 142–152.

Jones, S. and Hewitt, J. (1997), "Heuristic evaluation of web site usability: experiences from two case studies," *Adjunct proceedings of HCI'97*, pp. 23 – 25.

Jones, M. (1996), "Uniting authors and readers – active links on the World Wide Web," *APCHI'96 Conference Companion*, pp. 77–81.

Kaplan, C. and Fenwick, J. (1993), "Adaptive hypertext navigation based on user goals and context," *User modelling and user-adapted interaction*, 3, pp. 193 – 220.

Karis, D. and Zeigler, B. L. (1989), "Evaluation of mobile telecommunication systems," *Proceedings of Human Factors Society 33rd Annual Meeting*, pp. 205 – 209, as cited in **Nielsen, J.** (1993), *Usability Engineering*, AP Professional, U.S.A.

Kellogg, W.A. (1987), "Conceptual consistency in the user interface: Effects on user performance," *Second IFIP Conference on Human-Computer Interaction – Interact '87*, Stuttgart, Elsevier Science Publishers, North-Holland, pp. 389 – 394.

Kieras, D. E. and Polson, P. G. (1985), "An approach to the formal analysis of user complexity," *International Journal of Man-Machine Studies*, 22, pp. 365 – 394.

Kim, S. (1995), "Interdisciplinary Cooperation," In Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S. (eds.) *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 304 – 311, U.S.A.

Knuth, R.A. and Brush, T.A. (1990), "Results of the Hypertext '89 design survey," *Hypermedia*, 2(2), pp. 91 – 107.

Koike, H. (1995), "Fractal views: A fractal-based method for controlling information display," *ACM Transactions on Information Systems*, 13(3), pp. 305 – 323.

Larcker, D. F. and Lessig, V. P. (1980), "Perceived usefulness of information: A psychometric examination," *Decision Science*, 11, 1, pp. 121 – 134, as cited in **Chin, J.P., Diehl, V.A. and Norman, K.L.** (1988), "Development of an instrument measuring user satisfaction of the human-computer interface," *CHI'88 Proceedings*, pp. 213 – 218.

Landauer, T.K. (1995), *The trouble with computers: Usefulness, usability and productivity*, MIT Press.

Lewis, C. and Rieman, J. (1993), "Getting to know users and their tasks," in *Human-Computer Interaction: Toward the Year 2000*, Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S. (eds.), Morgan Kaufmann Publishers, pp. 122 – 127, 1995, U.S.A.

Lewis, C. (1982), "Using the 'thinking-aloud' method in cognitive interface design," *Research Report RC9265*, IBM T.J. Watson Rsearch Center, Yorktown Heights, N.Y.

Lim, K.Y. (1996), "Structured task analysis: an instantiation of the MUSE method for usability engineering," *Interacting with Computers*, 8(1), pp. 31 – 50.

Lindgaard, G. (1994), *Usability testing and system evaluation: A guide for designing useful computer systems*, Chapman & Hall.

Lowe, D.G. (1985), "Cooperative Structuring of Information: The Representation of Reasoning and Debate," *International Journal of Man-Machine Studies*, 23, pp. 97 – 111, as cited in **Conklin, J.** (1987), "Hypertext : An Introduction and Survey,""*IEEE Computer*, pp. 17 – 41.

Lynch, P. J. (1995), " Yale's World Wide Web Style Manual," *http://info.med.yale.edu/caim/manual-1.html*.

MacCaulay, L., Fowler, C., Kirby, M. and Hutt, A. (1990), "USTM: a new approach to requirements specification," *Interacting with Computers*, 2(1), pp. 92 – 108.

Maes, P. (1994), "Agents that reduce work and information overload," *Communications of the ACM*, 37, 7, ACM Press: New York, 31–40.

Mantei, M. (1982), "Disorientation behaviour in Person-Computer Interaction, " *PhD Thesis*, University of Southern California.

Marchionini, G. (1989), "Information-seeking strategies of novices using a full-text electronic encyclopaedia," *Journal American Society Information Science*, 40(1), pp. 54 – 66.

Marchionini, G. and Shneiderman, B. (1988), " Finding facts vs browsing knowledge in hypertext systems," *IEEE Computing*, pp. 70 – 79.

Maurer, H. (1996), *HyperWave: The Next Generation WEB Solution*, Addison-Wesley.

Mayes, T., Kibby, M. and Anderson, T. (1990), "Learning about learning from hypertext," in *Designing hypermedia for learning*, Jonassen, D.H. and Mandi, H. (Eds), Springer-Verlag, pp. 13.1 – 13.24.

Mayhew, D. (1992), *Principles and guidelines in software user interface design*, PTR Prentice Hall, as cited in **Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S.** (eds.) (1995a), "Design and evaluation," *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 73 – 91, U.S.A.

McAleese, R. (1993), "Navigation and browsing in hypertext," *Hypertext: Theory into Practice*, pp. 1 – 38, Intellect Books, U.K.

McKnight, C., Dillon, A. and Richardson, J. (1991), *Hypertext in Context*, Cambridge University Press, U.K., pp. 64 – 104.

Mea, V.D., Beltrami, C.A., Roberto, V. and Brunato, D. (1996), "HTML generation and semantic markup for telepathy," *Proceedings of the 5th W3 Conference*,, pp. 1085 – 1094.

Mellor, P. (1995), "Computer-Related Factors in the Excalibur Incident: A320 G-KMAM of Excalibur Airways Ltd., " Centre for Software Reliability, Issue no. 3, City University.

Mendes, M.E.X. and Hall, W. (1997), "The SHAPE of Hypermedia Authoring for Education," to appear in *Proceedings of ED-MEDIA and ED-TELECOM'97*, Calgary, Canada.

Mohageg, M. F. (1992), "The influence of hypertext linking structures on the efficiency of information retrieval, " *Human Factors*, 34(3), pp. 351 – 367.

Monk, A. (1995), "Interdisciplinary Approaches to Multimedia Research," in Emmott (ed.) *Information Superhighways: Multimedia Users and Futures*, Academic Press, U.S.A.

Monk, A. and Gilbert, N. (1995), "Inter-disciplinary research," in Monk, A. and Gilbert, N. (eds.), *Perspectives on HCI: Diverse Approaches*, Computers, People Series, Academic Press.

Monk, A. (1990), "Getting to known locations in hypertext," in McAleece, R. and Green, C. (eds.),*Hypertext: State of the Art*, Intellect Books, pp. 20 – 27.

Moran, T.P. (1981), "The command language grammar," *International Journal of Man-Machine Studies*, 15, pp. 3 – 50.

Morse, P.M. (1973), "Browsing and search theory," *Toward a Theory of Librarianship:Papers in honour of Jesse Hauk Shera*, as cited in **McAleese, R.** (1993), "Navigation and browsing in hypertext," *Hypertext: Theory into Practice*, pp. 1 – 38, Intellect Books, U.K.

Nanard, J. and Nanard, M. (1995), "Hypertext design environments and the hypertext design process," *Communications of the ACM*, Special issue on hypermedia design, pp. 49 – 56.

Nanard, J. and Nanard, M. (1993), "Should anchors be typed too? An experiment with MacWeb," in Kacmar, C.J. and Schnase, J.L. (eds.), *Hypertext '93 Proceedings*, pp. 51-62, ACM Press.

Nelson, T. H. (1987), *Literary Machines*, Vers. 87.1, The Distributors, South Bend, IN.

Newman, W. M. and Lamming, M. G. (1995), *Interactive System Design*, Addison-Wesley.

Nielsen, J. (1996), "Relationships on the web," *http://www.useit.com/alertbox/9601.html.*

Nielsen, J. (1995a), "A home-page overhaul using other web sites," *IEEE Software*, U.S.A.

Nielsen, J. (1995b), *Multimedia and Hypertext: The Internet and Beyond*, AP Professional, U.S.A.

Nielsen, J. (1993), *Usability Engineering*, AP Professional, U.S.A.

Nielsen, J. and Landauer, T. (1993), "A Mathematical Model of the Finding of Usability Problems," *Proceedings of Interchi'93*, pp. 206 – 213, ACM Press, U.S.A.

Nielsen, J. and Molich, R. (1990), "Heuristic evaluation of user interfaces," *ACM CHI'90 Conference*, pp. 249 – 256.

Norman, D. (1986), "Cognitive Engineering," *User Centered System Design: New Perspectives on Human-Computer Interaction*, pp. 31 – 61, Lawrence Erlbaum Associates, London.

Norman, D. (1983), "Design principles for human-computer interaction," *Proceedings of CHI'83*, ACM, pp. 1 – 10.

Nunnally, J. C. (1978), *Psychometric Theory*, McGraw-Hill Book Company, U.S.A., as cited in **Chin, J.P., Diehl, V.A. and Norman, K.L.** (1988), "Development of an instrument measuring user satisfaction of the human-computer interface," *CHI'88 Proceedings*, pp. 213 – 218.

O'Connor, B.C. (1985), "Access to moving image documents: background concepts and proposals for surrogates for film and video works," *Information Retrieval Research*, 41(4), pp. 209 – 220, as cited in **McAleese, R.** (1993), "Navigation and browsing in hypertext," *Hypertext: Theory into Practice*, pp. 1 – 38, Intellect Books, U.K.

Parunak, H. (1991), "Ordering the information graph," in Berk, B. and Devlin, J. (eds.), *Hypertext/Hypermedia Handbook*, Armadillo Associates, pp. 299 – 325.

Parunak, H. (1989), "Hypermedia Topologies and User Navigation," *Proceedings of Hypertext '89*, pp. 43 – 50, ACM Press, U.S.A.

Payne, S. and Green, T. R. G. (1989), "Task-action Grammar: The model and its developments," in *Task Analysis for Human-Computer Interaction*, Ellis Horwood, as cited in **Johnson, P.** (1992), *Human Computer Interaction: Psychology, Task Analysis and Software Engineering*, McGraw-Hill.

Payne, S. and Green, T.R.G. (1986), "Task-action grammars: a model of the mental representation of task languages," *Human-Computer Interaction*, 2(2), pp. 93 – 133.

Pejtersen, A. M. (1989), "A library system for information retrieval based on cognitive task analysis and supported by an icon-based interface," *Proceedings SIGIR'89 12th Annual International Conference on Research and Development in Information Retrieval*, pp. 40 – 47, as cited in **Nielsen, J.** (1995b), *Multimedia and Hypertext: The Internet and Beyond*, AP Professional, U.S.A.

Perez, E. (1991), "Tools for authoring hypertexts," in Berk and Devlin (eds.), *Hypertext/Hypermedia Handbook*, McGraw-Hill, pp. 209 – 224, U.S.A.

Perlman, G. (1988), "Software tools for user interface development," in Helander, M. (ed.), pp. 819 – 833, as cited in **Preece, J., Benyon, D., Davies, G., Keller, L., and Rogers, Y.** (1993), *A guide to usability: Human factors in computing*, Addison-Wesley.

Phillips, M.P., Bashinski, H.S., Ammerman, H. and Fligg, C. (1988), "A task analytic approach to dialogue design, in Helander, M. (ed.),*Handbook of Human-Computer Interaction*, pp. 835 – 857, North-Holland.

Pirolli, P. and Card, S. (1995), "Information foraging in information access environments," *CHI'95*, pp. 51 – 58.

Pitkow, J.E. and Jones, R.K. (1996), "Supporting the web: A distributed hyperlink database system," *Proceedings of the 5th W3 Conference*,, pp. 981 – 991.

Pitkow, J. and Kehoe, C. (1995), *GVU's WWW 4th User Surveys*, http://www.cc.gatech.edu/gvu/user_surveys/survey-10-1995/

Pittas, M. (1995), *Navigation in Hypertext*, PhD Thesis, Queen Mary and Westfield College of London of University.

Polson, P. G. and Lewis, C.H. (1990), "Theory-based design for easily learned interfaces," *Human Computer Interaction*, 5, pp. 191 – 220.

Pontin, R. and Neto, A.G. (1996), "A link-oriented tool for the evaluation of hyperdocuments," *The Missing Link: Hypermedia usability research and the Web*, Open University, 1996.

Pozadzides, J. and Quinn, L. (1997), "Cascading Style Sheet Quick Tutorial," *http:www.htmlhelp.com/reference/css/quick-tutorial.html*.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. and Carey, T. (1994), *Human-Computer Interaction,* Addison-Wesley.

Preece, J., Benyon, D., Davies, G., Keller, L., and Rogers, Y. (1993), *A guide to usability: Human factors in computing,* Addison-Wesley.

Rada, R. and Murphy, C. (1992), "Searching versus browsing in Hypertext," *Hypermedia,* 4(1), pp. 1-30, U.K.

Reisner, P. (1981), "Formal grammar and human factors design of an interactive system, *IEEE Transactions on Software Engineering,* 7(2), pp. 229 – 240.

Reason, J. (1988), "Framework models of human performance and error," in Rasmussen and Leplat (eds), *New Technology and Human Error,* Viley and Sons, Chichester.

Rigny, C. (1997), "Automatic generation of executable user models for evaluating the usability of interactive systems," *unexamined PhD Thesis* , Middlesex University.

Rittel, H. and Webber, M. (1973), "Dilemmas in a General Theory of Planning," *Policy Sciences,* 4, as cited in **Conklin, J.** (1987), "Hypertext : An Introduction and Survey," *IEEE Computer,* pp. 17 – 41.

Rivlin, E., Botagogo, R. and Snneiderman, B. (1994), "Navigating in hyperspace: Designing a structure-based toolbox," *Communications of the ACM,* 37(2), pp.87 – 96.

Root, R. W. and Draper, S. (1983), " Questionnaires as a software evaluation tool," *Proceedings of ACM CHI'83 Conference,* pp. 83 – 87, as cited in **Nielsen, J.** (1993), *Usability Engineering,* AP Professional, U.S.A.

Rosson, M.B., Mass, S. and Kellogg, W.A. (1990), "The designer as user: building requirements for design tools from design practice," *Communications of the ACM,* 31, 11, pp. 1289 – 1298, as cited in **Johnson, P.** (1992), *Human Computer Interaction: Psychology, Task Analysis and Software Engineering,* McGraw-Hill.

Rubinstein, R. and Hersh, H. (1984), *The human factor,* Digital Press.

Suchman, L. A. (1987), *Plans and situated actions: The problems of human-machine communication,* Cambridge University Press.

Sebillotte, S. (1988), "Hierarchical planning as a method for task analysis: the example of office task analysis," *Behaviour and Information Technology,* 7(3), pp. 275 – 293.

Scott, J.R. (1994), "Library information access client," *ACM CHI'94 Conference Campanion,* pp. 143 – 144, as cited in **Nielsen, J.** (1995b), *Multimedia and Hypertext: The Internet and Beyond,* AP Professional, U.S.A.

Shackel, B. (1991), "Usability - context, framework, definition, design and evaluation," In Shackel, B. and Richardson, S. (eds.), *Human Factors for Information Usability,* Cambridge University Press.

Shneiderman, B. (1997), "Designing information-abundant websites: issues and recommendations," to appear in *International Journal of Human-Computer Studies (Special issue on Human-Computer Interaction and the World Wide Web).*

Shneiderman, B. (1992), *Designing the User Interface: Strategies for Effective Human-Computer Interaction,* Addison-Wesley, U.S.A., pp. 1 – 49.

Shneiderman, B. and Kearsley, G. (1989), *Hypertext Hands-On! An Introduction to a New Way of Organising and Accessing Information*, Addison-Wesley, U.S.A.

Signore, O. (1995), "Issues on Hypertext Design," *DEXA'95 Conference Proceedings*, London.

Smith, J. (1997), "The King is dead; long live the King," *Hypertext'97 keynote address*, pp. 240.

Smith, P. (1994), *Navigating Hyperspace*, PhD Thesis, University of Nottingham.

Smith, S. and Mosier, J. (1986), *Guidelines for designing user interface software*, Report No. 7 MTR-10090, ESD-TR-86-278, MITRE Corporation, as cited in **Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S.** (eds.) (1995a), "Design and evaluation," *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, pp. 73 – 91, U.S.A.

Smith, P. and Newman, I. (1997), "Applying usability research to the Web: Virtual hypermedia domains and virtual search hierarchies," in Buckingham Shum, S. and McKnight, C. (eds.), Special Issue of *International Journal of Human-Computer Studies on 'Web Usability'* .

Smith, J.B. *et al.* (1986), "WE: A Writing Environment for Professional," *Technical Report 86 – 025*, Department of Computer Science, University of North Carolina at Chapel Hill, as cited in **Conklin, J.** (1987), "Hypertext : An Introduction and Survey," *IEEE Computer*, pp. 17 – 41.

Taylor, C. and Self, J. (1990), "Monitoring hypertext users," *Interacting with Computers*, 2(3), pp. 297 – 312.

Thimbleby, H. (1997), "Gentler: A tool for systematic web authoring," in Buckingham Shum, S. and McKnight, C. (eds.), Special issue of *International Journal of Human-Computer Studies on 'Web Usability'*.

Thimbleby, H. (1995a), "Authoring consistent hypermedia without getting lost," *HCI'95 Adjunct Proceedings*, pp. 118 – 124.

Thimbleby, H. (1995b), "Treat People Like Computers?" *Extraordinary People and Human-Computer Interaction*, Edwards A, editor, Cambridge University Press, pp. 283 – 295.

Thimbleby, H. (1995c), "Middlesex University Style Guide," *http://www.cs.mdx.ac.uk/esrc/style.html*.

Thimbleby, H. (1992), "Heuristics for Cognitive Tools," in NATO ASI Series F, *Proceedings NATO Advanced Research Workshop on Mindtools and Cognitive Modelling, Cognitive Tools for Learning*, Kommers P A M, Jonassen D H & Mayes J T, editors, pp. 161 – 168, Springer Verlag.

Thimbleby, H. (1990), *User Interface Design*, ACM Press, U.S.A.

Thimbleby, H. and Addison, M. (1995), "HyperDoc: An interactive systems tool," *HCI'95 Proceedings*, pp. 95 – 106, U.K.

Tilton, J. (1996), "Composing good HTML," *http://www.cs.cmu.edu/~tilt/cgh*.

Tripp, S.D. and Roby, W. (1990), "Orientation and disorientation in a hypertext lexicon," *Journal of Computer-Based Instruction*, 17(4), pp. 120 – 124.

Tydesley, D. A. (1988), "Employing usability engineering in the development of office products," *Computer Journal*, 31, 5, pp. 431 – 436.

Van Dam, A. (1987), "Hypertext '87 keynote address," *Communications of the ACM*, 31(7), pp. 887 – 896.

Vocht, J.W. (1994), "Experiments for the characterisation of hypertext-structures," *Master's Thesis*, Eindhoven University of Technology, Department of Mathematics and Computing Science.

Voltaire, F.M. A. (1764), *Dictionnaire Philosophique*, English translation published in 1765 as *Philosophical Dictionary* as cited in **Nielsen, J.** (1993), *Usability Engineering*, AP Professional, U.S.A.

Vries, E. (1993), "Stretching the initial problem space for design problem solving: Browsing versus searching in network and hierarchy structures," *Technical Report OCTO-report 93/02*, Eindhoven University of Technology, Department of Mathematics and Computing Science.

Wharton C., Rieman J., Lewis, C. and Polson P. (1994), "The Cognitive Walkthrough method: a practitioner's guide," in *Usability Inspection Methods*, Nielsen (ed.), John Wiley and Sons, pp. 105–140.

Whitefield, A., Wilson, F. and Dowell, J. (1991), "A framework for human factors evaluation," *Behaviour and Information Technology*, 10 (1), pp. 65 – 79.

Wilson, M., Barnard, P.J. and MacLean, A. (1986), "Task analysis in human computer interaction," *Report HF 122*, IBM Hursley Human Factors.

Wilson, F. and Clarke, A. (1993),"Evaluating system design realisations," in Byerley, F., Barnard, P.J. and May, J. (eds.), *Computers, Communication and Usability: Design issues, research and methods for integrated services*, North-Holland series in Telecommunication, Elsevier.

Witten I.H., Cunningham S.J., Vallabh M. and Bell, T.C. (1995), "A New Zealand digital library for computer science research," *Proc Digital Libraries '95*, Austin, Texas, pp. 25-30.

Wium, L. H. and Bos, B. (1996), "Cascading Style Sheets, Level 1," *http://www.w3.org/pub/WWW/TR/PR-CSS1*.

Wray, R. E., Chong, R., Phillips, J., Rogers, Wlash, W. and Laird, J. (1994), "Organising information in Mosaic: A classroom experiment," http://ai.eecs.umich.edu/cogarchO/mosaic-paper/wrayre-experiment-mosaic94.html.

Wright, P. (1989), "Interface alternatives for hypertexts," *Hypermedia*, 1(2).

Wright, P. and Monk, A.F. (1989), "Evaluation for design," in *People and Computers V. Proceedings of the 5th Conference of the British Computer Society on Human-Computer Interaction*, Cambridge Press, pp. 345 – 358.

Zizi, M. and Beaudouin-Lafon, M. (1994), "Accessing hyperdocuments through interactive dynamic maps," *Proceedings of ACM European Conference on Hypermedia Technology, ECHT'94*, ACM Press, pp. 126 – 135.

Appendices

Appendix A

More information about HyperAT

Appendix A1

Appendix A2

Appendix A3

Appendix A4

Appendix A1

HTML Menu Option

Figure A1.1 shows the HTML Menu and the associated submenus. The HTML Menu allows you to convert a HTML document into a structure recognised by HyperAT. Using this structure, designers can then perform analysis of the hyperdocument.

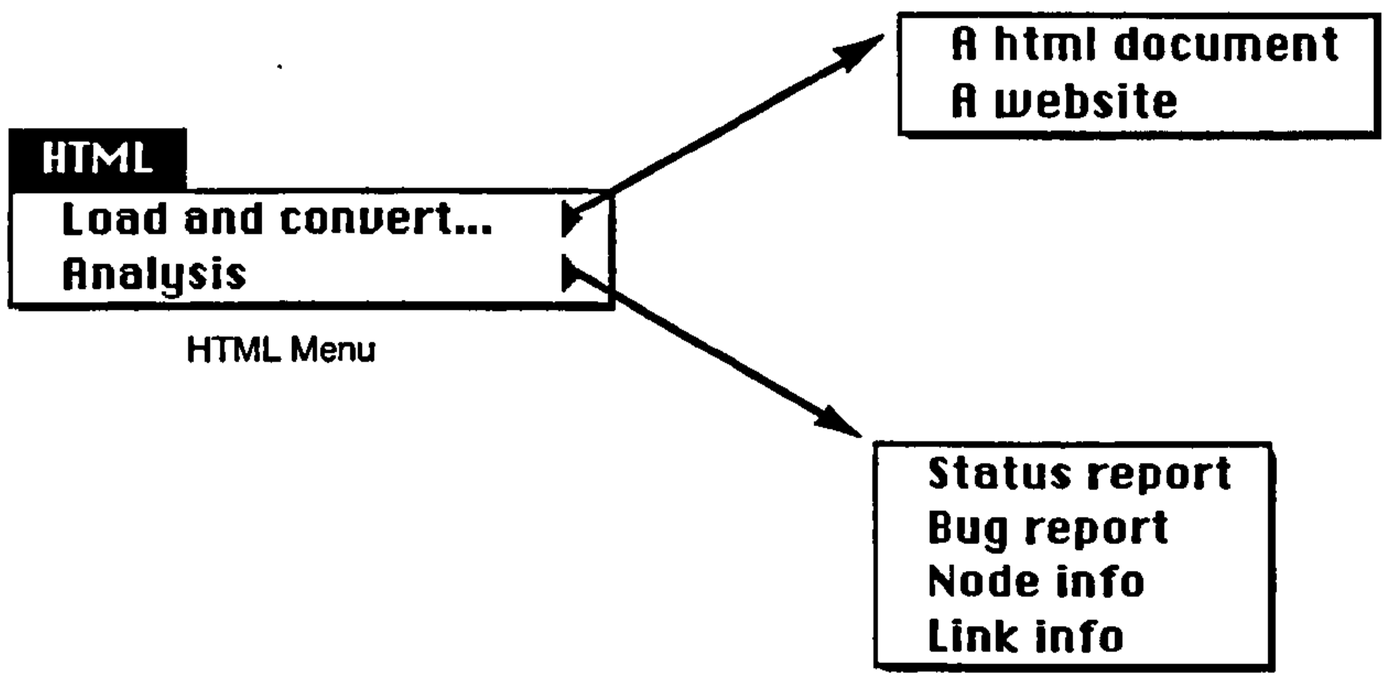


Figure A1.1. HTML Menu and submenus

Loading and converting...

To invoke the loading and converting process, select **Loading and converting...** from the menu. This option allows you to either select a HTML document or an entire website. Clicking on **A HTML document** option in the submenu will open a scrolling window. Once you have selected the required file, press "open" to confirm the selection (see figure A1.2).

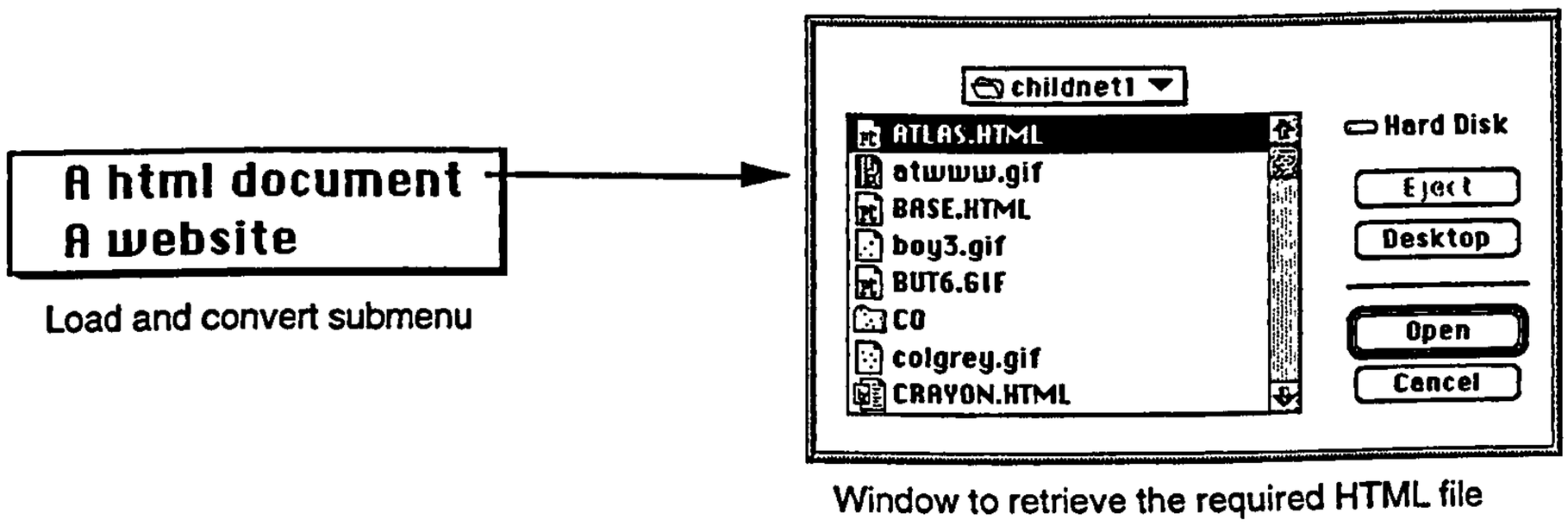


Figure A1.2. Loading a HTML document from directory

After loading the selected file, HyperAT then converts the HTML document into a structure recognised by HyperAT. This means that designers can import any HTML document and convert it to a form that can be analysed further by HyperAT. Figure A1.3 shows two files: a

hyperdocument in HTML format (on the left, before the conversion) and a converted structure (on the right, after the conversion).

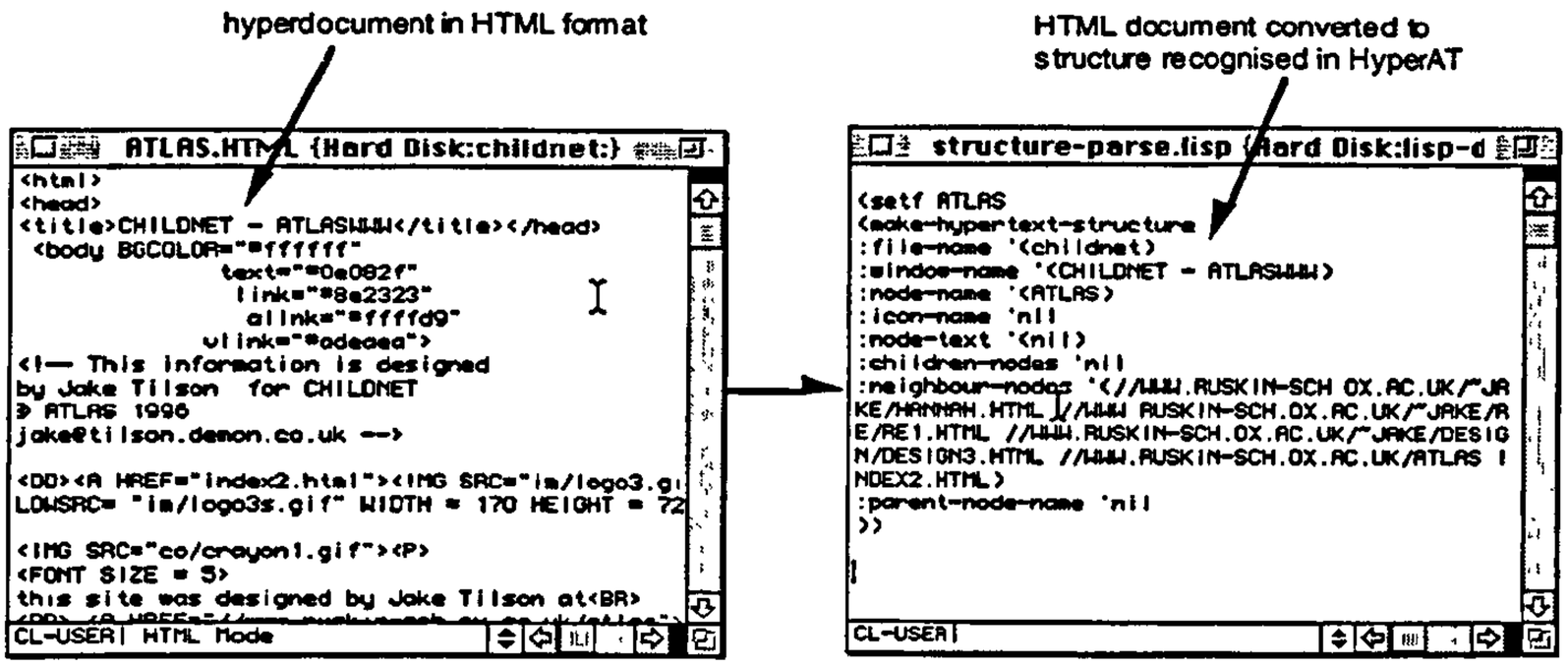


Figure A1.3. A HTML document and the respective structure after the conversion

To load and convert an entire website containing HTML documents, click on **A website** option in the **Loading and converting** menu. A window is then opened to ask you to select the required directory. Once selected, press "open" to confirm. Another window showing the selected directory is opened. Click on "select current folder" to confirm. The converted documents can then be analysed by HyperAT. Figure A1.4 shows the loading and conversion sequence using "Childnet" directory as an example.

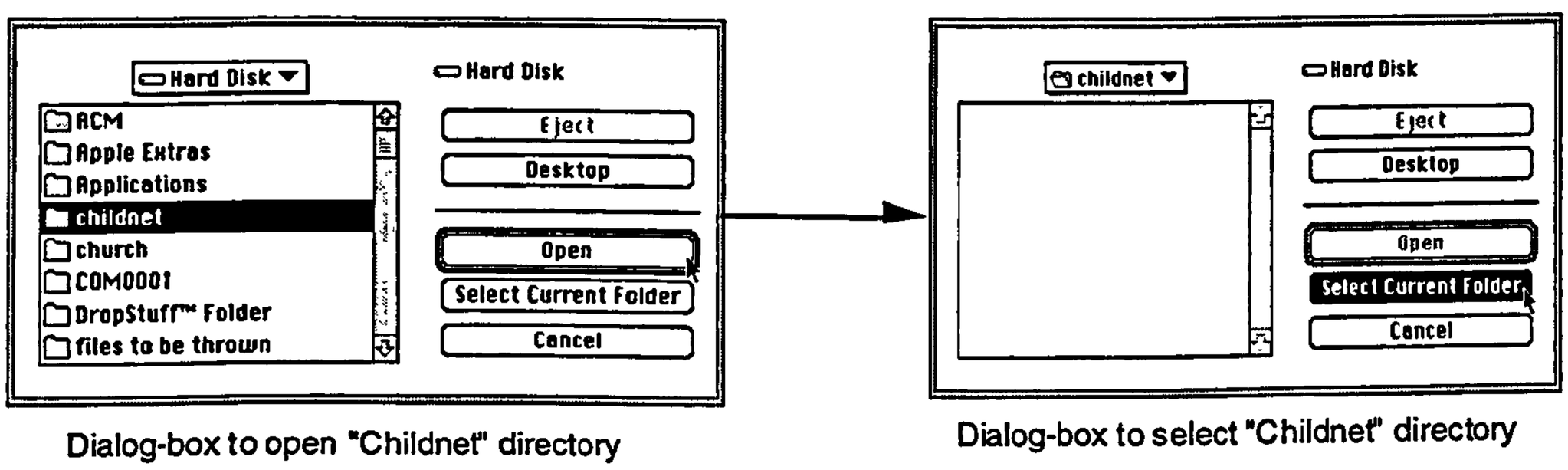


Figure A1.4. Loading a "Childnet" website

The converted file(s) can then be analysed by HyperAT. Figure A1.5 shows the Analysis Menu and the submenus of the analysis that can be performed in HyperAT.

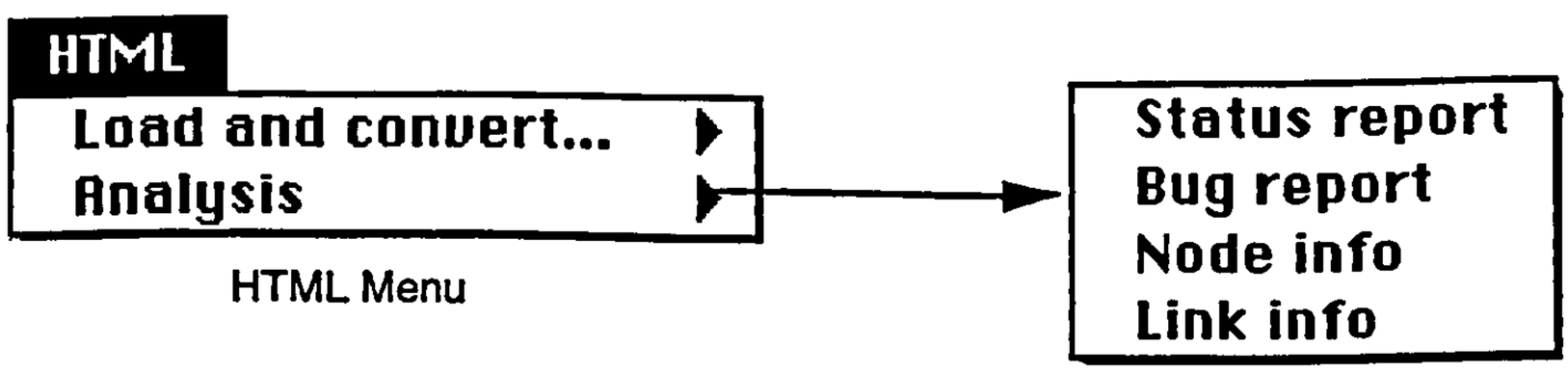


Figure A1.5. Analysis Menu and submenu

Status report

The **Status report** option will generate a report which captures all the information about the nodes and the respective links of the HTML document(s) parsed. You have the option of either viewing it on the screen, saving it or printing it out. Figure A1.6 shows the report.

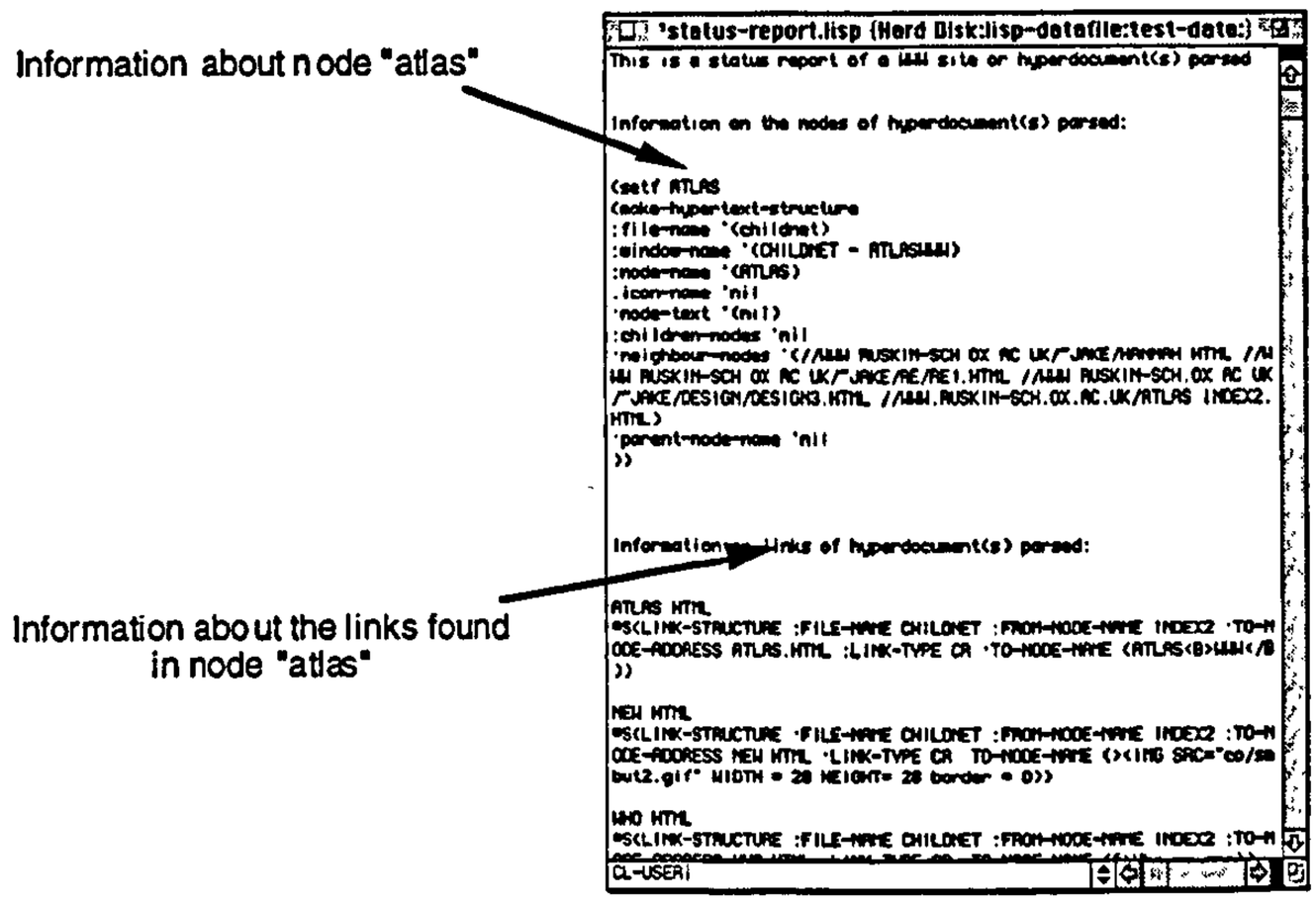


Figure A1.6. Report on the nodes parsed

Bug report

Clicking on the **Bug report** option will generate a report which will point out discrepancies detected by HyperAT. This report identifies the list of .gif files that have been picked up by HyperAT and compares them with the .gif files present in the website (see figure A1.7).

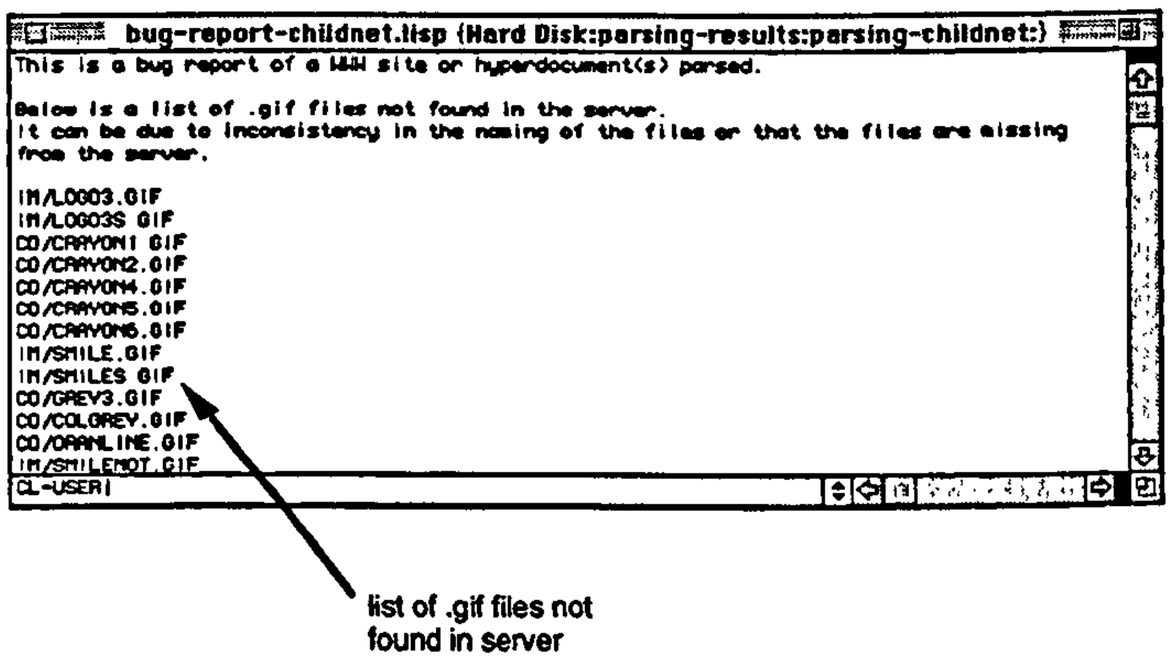


Figure A1.7. Bug report when parsing an example website

Node information

You can also perform a query on a particular node and the information is displayed on the screen. To do that, click on the **Node info** option. A dialog box will appear to prompt you to select the desired node (see figure A1.8). When selected, press "O.K." and the information of the node is displayed on the screen in the Display Window.

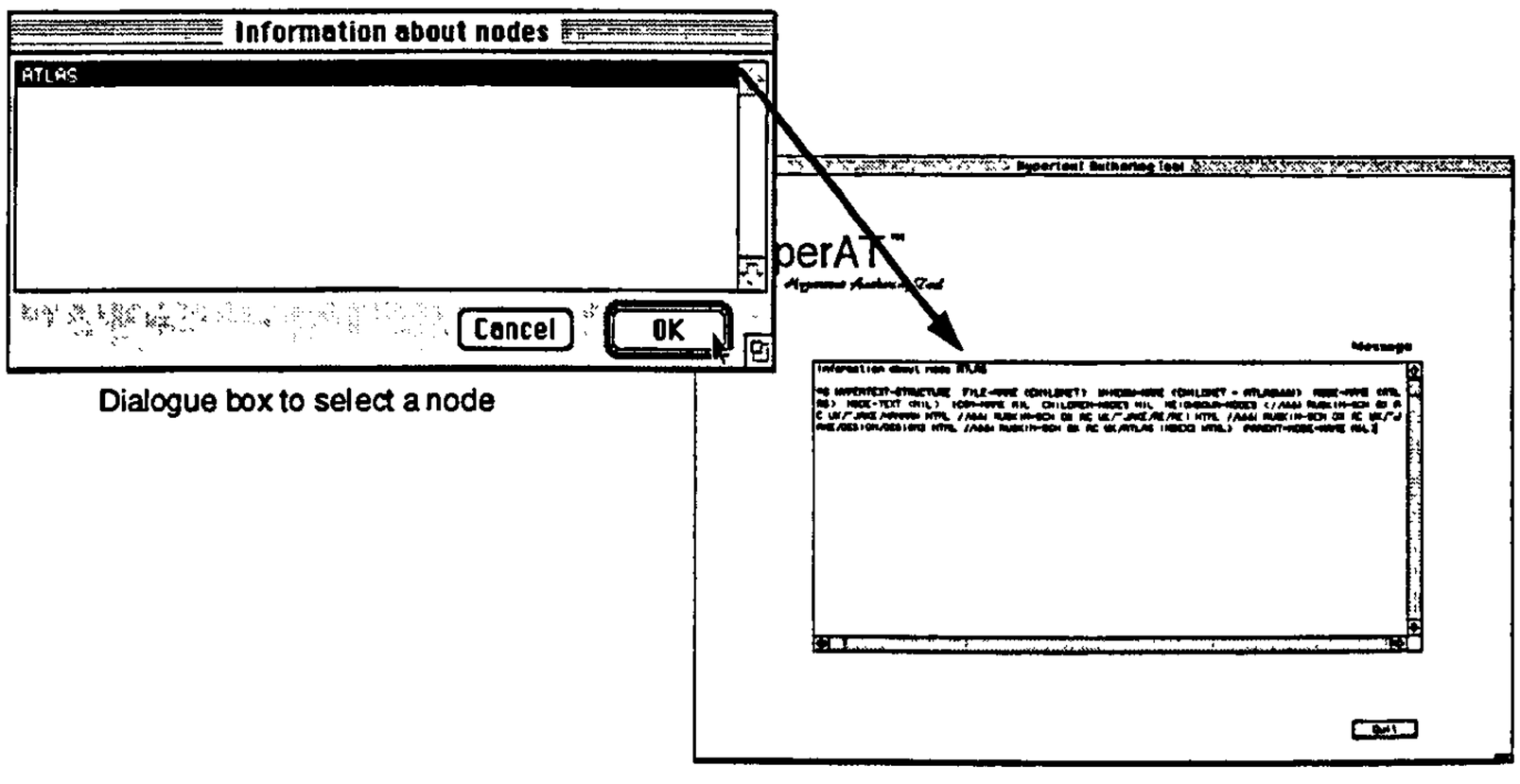


Figure A1.8. Selection and display of node information

Link information

Similarly, if you can make a query about a link found in the hyperdocument(s) parsed. To do this, click on the **Link info** option. A dialog box containing all the links present in the hyperdocument(s) appears. Highlight the desired link and confirm your selection by pressing "O.K.". The information about the selected link is displayed in the Display Window (see figure A1.9).

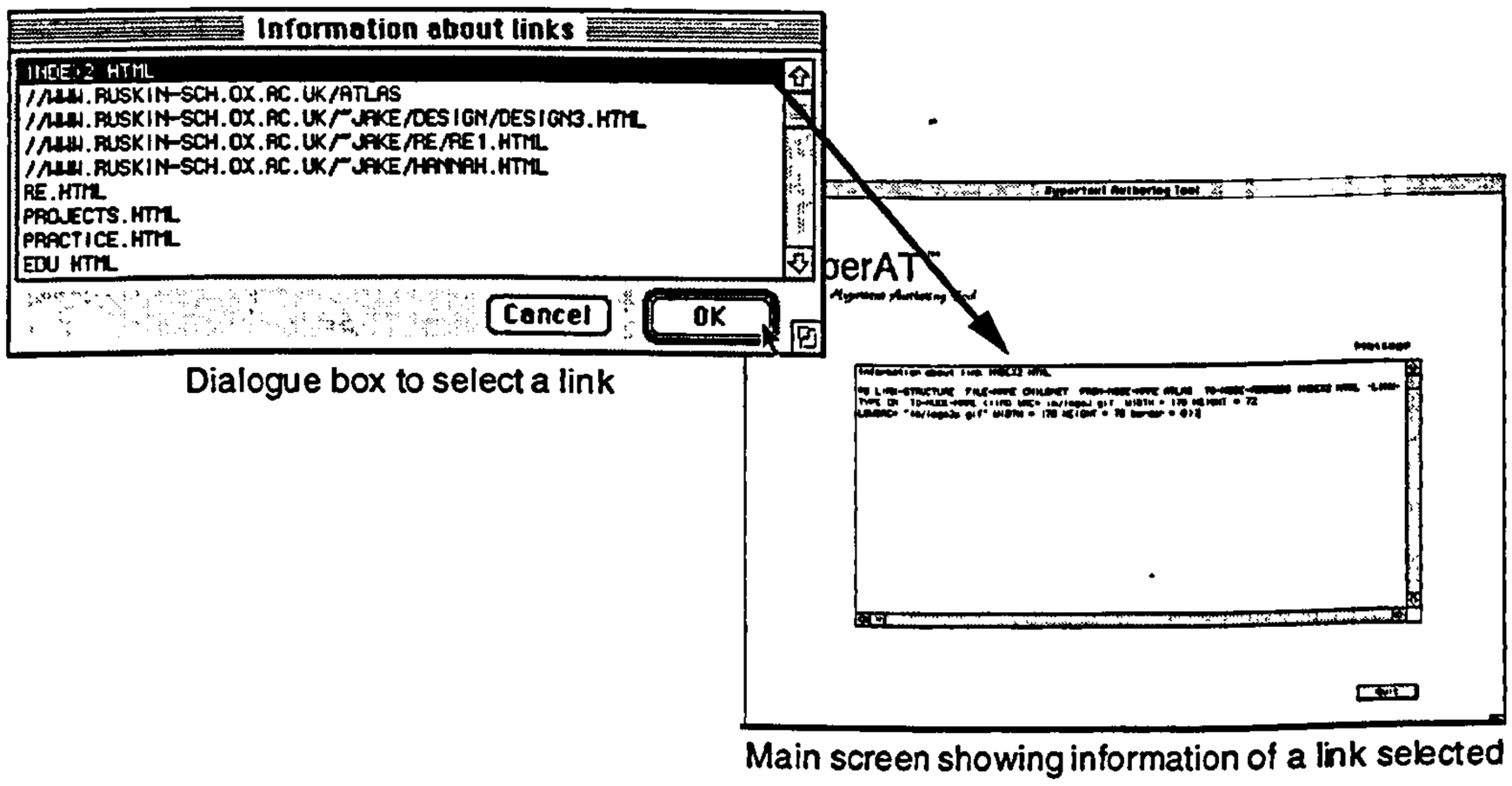


Figure A1.9. Selection and display of link information

Appendix A2

Analysis Menu Option

To help you build usable hyperdocument, we have incorporated within HyperAT a testbed to perform different levels of usability testing. One way to test whether a hyperdocument is well-structured is to provide designers with processed information on the nodes and links created. For example, missing nodes not yet created, inconsistencies in the naming of nodes and links. Besides performing a statistical analysis on the hyperdocument, the Analysis Menu (see figure A2.1) implements one form of usability testing that can be carried out on a hyperdocument involving real users' inputs. Further work to be done involves simulation of non-human users.

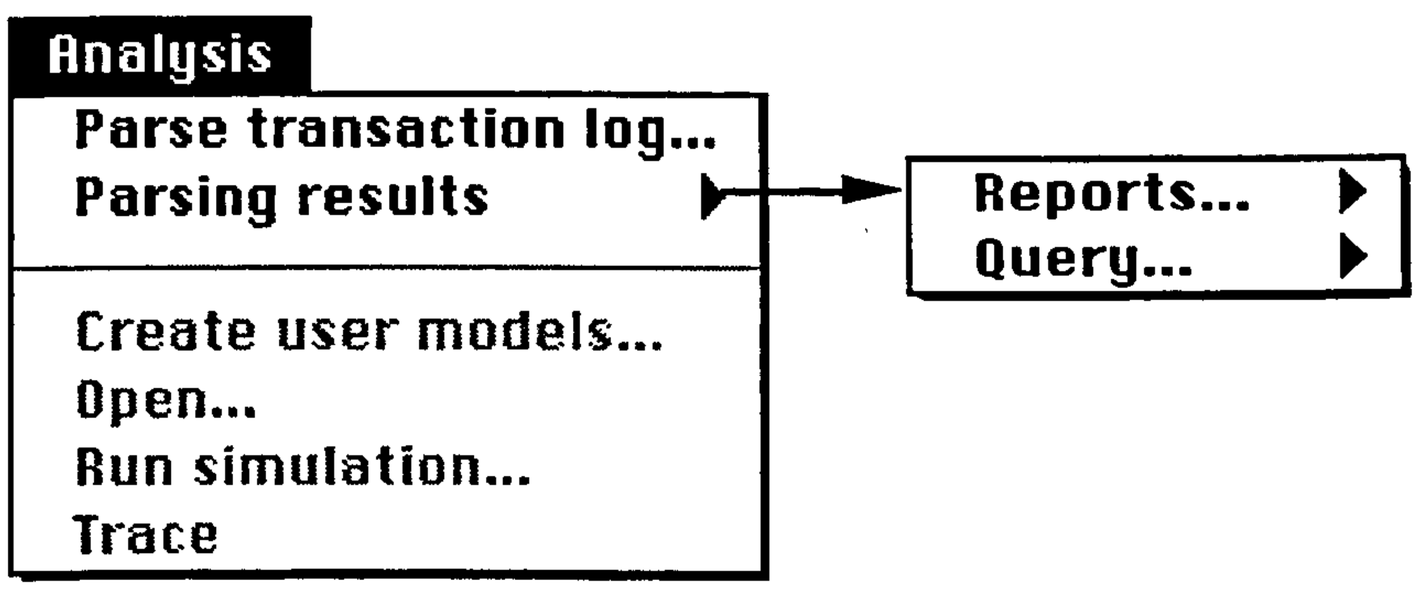


Figure A2.1. Analysis Menu and submenu

A2.1 Parsing transaction log

Figure A2.2 shows a sample transaction log file of a particular WWW server. This server has been configured to display certain types of information in the transaction log file. When you click on the **Parse transaction log** option, HyperAT reads the transaction log file(s) found in a designated directory. You can either obtain a print of the analysis or make a query and read the information off the screen.

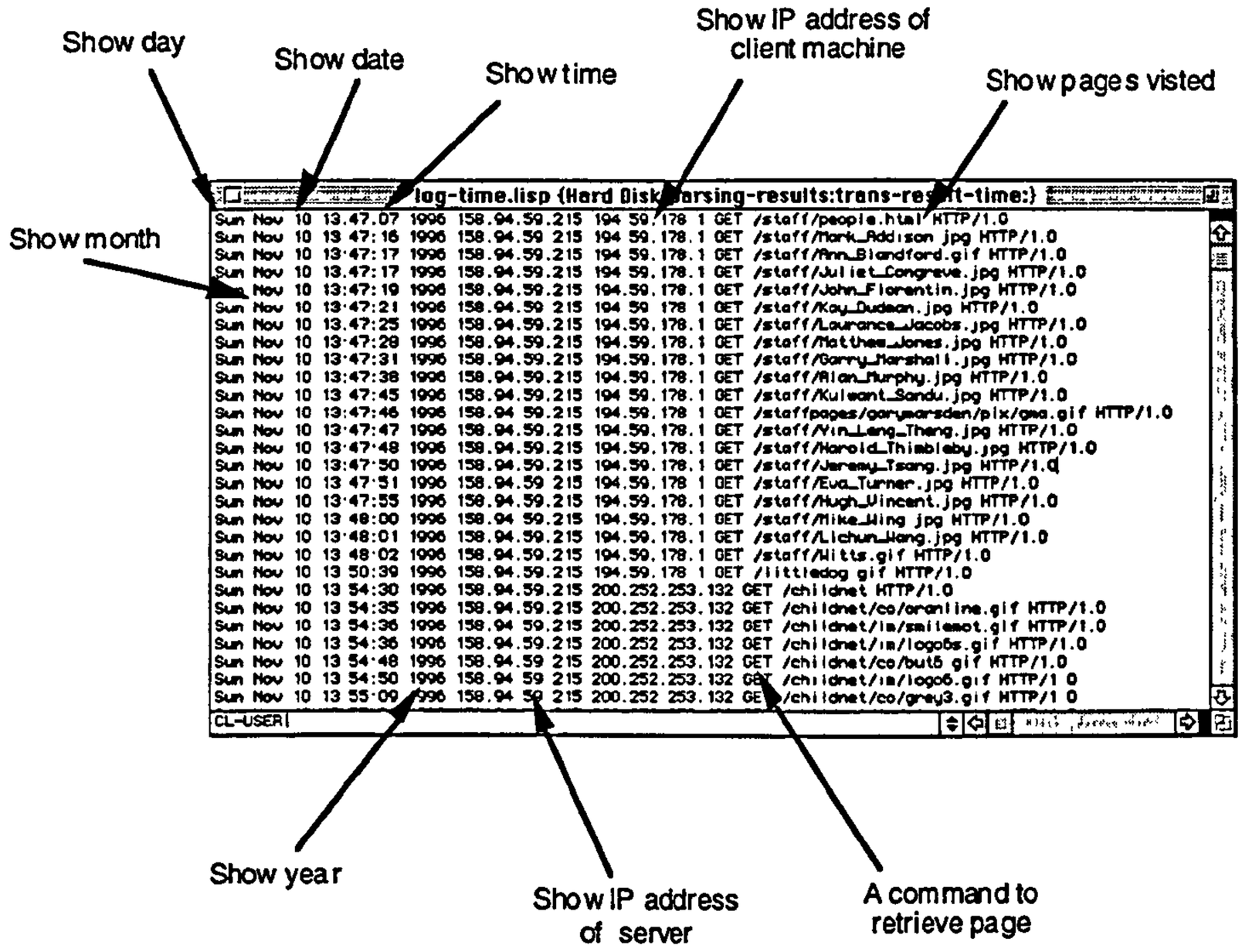


Figure A2.2. A sample of a transaction log file

A2.2 Reports

Real user testing can be carried out on the WWW and the users' transactions are logged in the WWW server. HyperAT analyses a WWW server's transaction log files to provide hints of users' browsing pattern, which may in turn give designers useful information on the design and structure of hyperdocuments. Three kinds of reports are generated by HyperAT You can obtain these reports by selecting the **Reports** option in the **Parsing results** submenu (see figure A2.3).

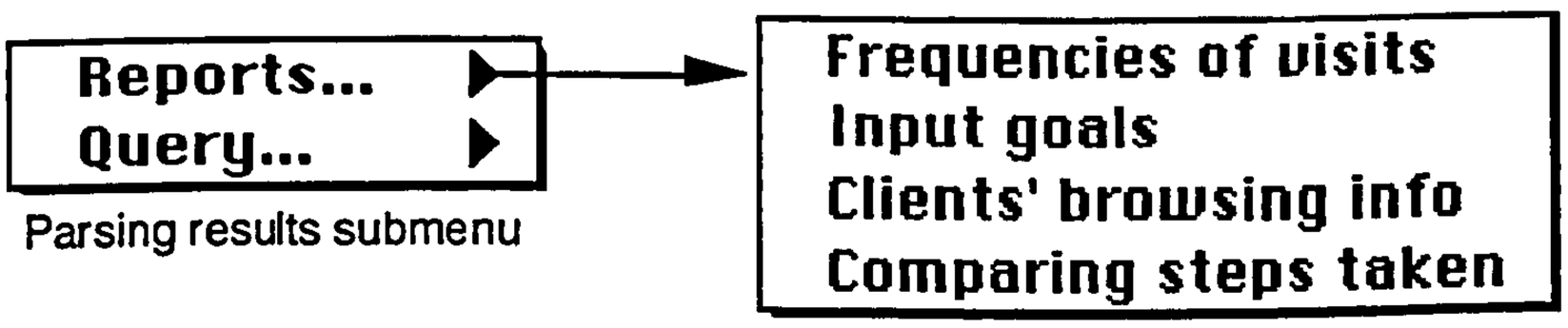


Figure A2.3. Reports Menu and submenu

- **Frequencies of visits**

Clicking on the Frequencies of visits option generates a report (see figure A2.4). This report provides designers with information on the frequencies and dates of visits of the websites resident in the WWW server.

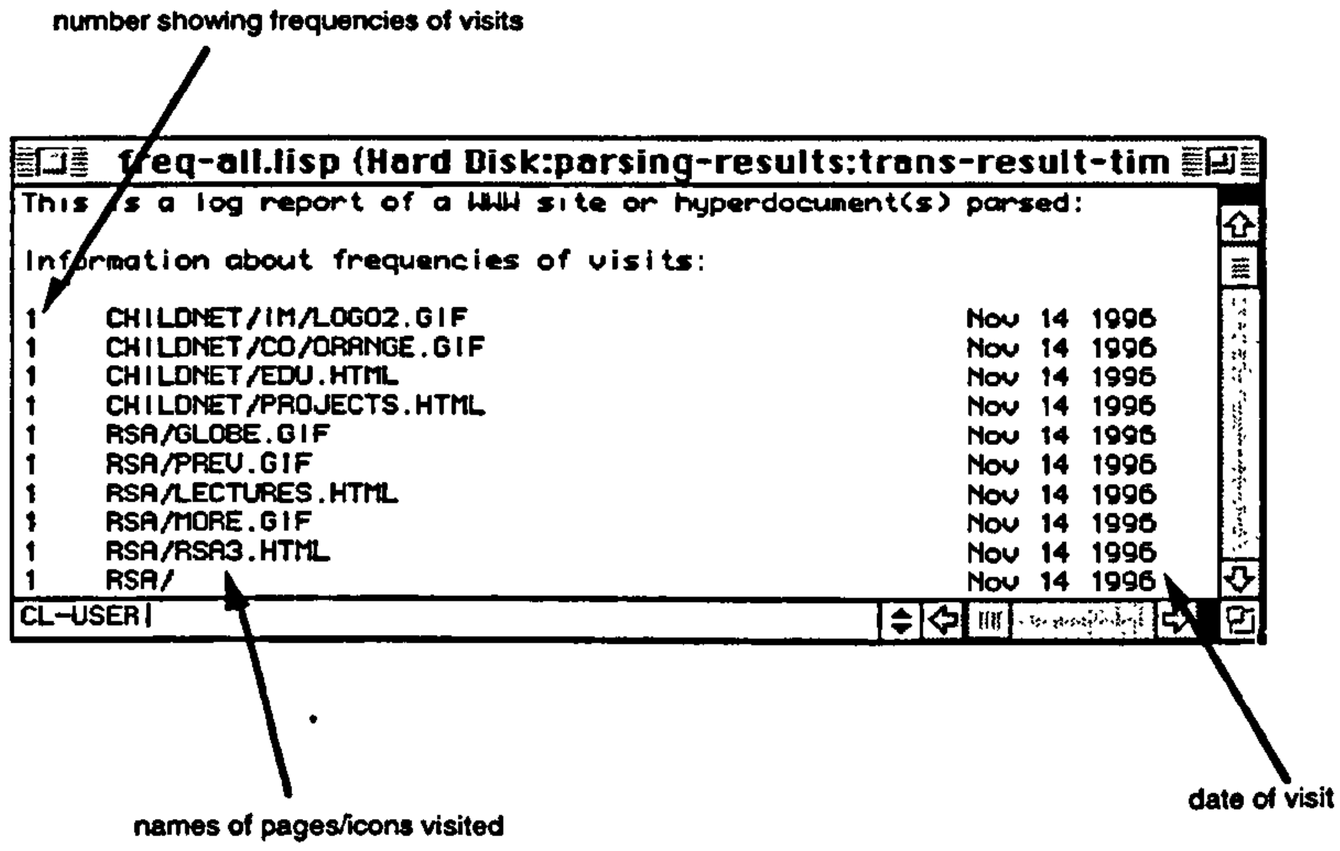


Figure A2.4. Report showing the frequencies of visits to websites held by a web server

- **Input goals**

Clicking the **Input goals** option opens a window (see figure A2.5). This window asks you to specify the goal you would like to investigate and the respective number of steps taken to achieve it. Press "More" if you want to investigate more goals. Press "Done" if you have finished and "Cancel" if you want to abandon the operation.

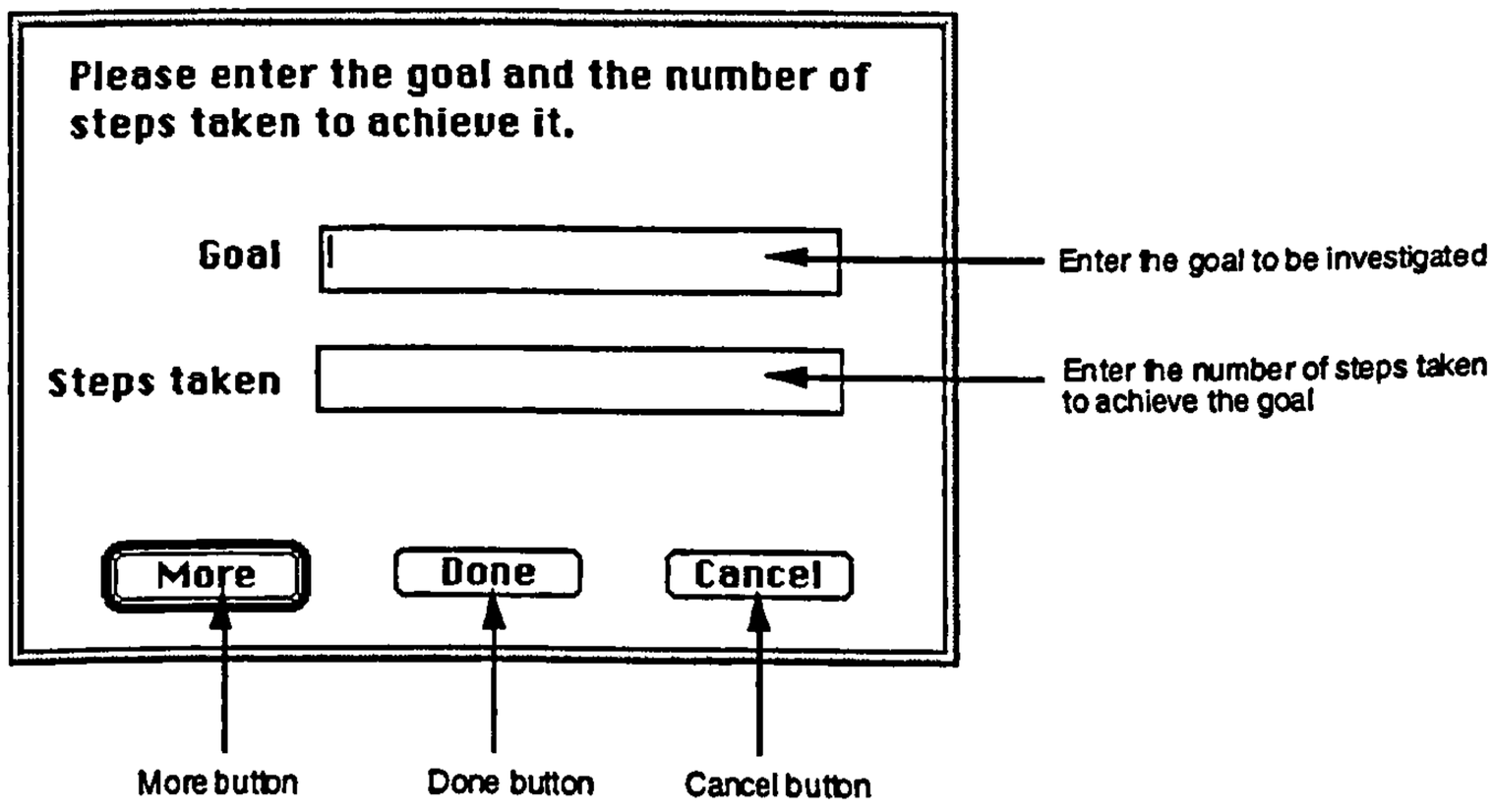


Figure A2.5. Input screen to capture the goal(s) and the respective steps taken to achieve them.

- **Clients' browsing information**

This option can only be activated if the **Input goals** option is completed. Clicking on **Clients' browsing info** option generates a report to print out information on clients' browsing information such as date of visit, browsing path, time spent in each web page, number of steps taken by a client to achieve the goal(s), *etc.* Figure A2.6 is a sample report generated by HyperAT using the "Childnet" website as an example.

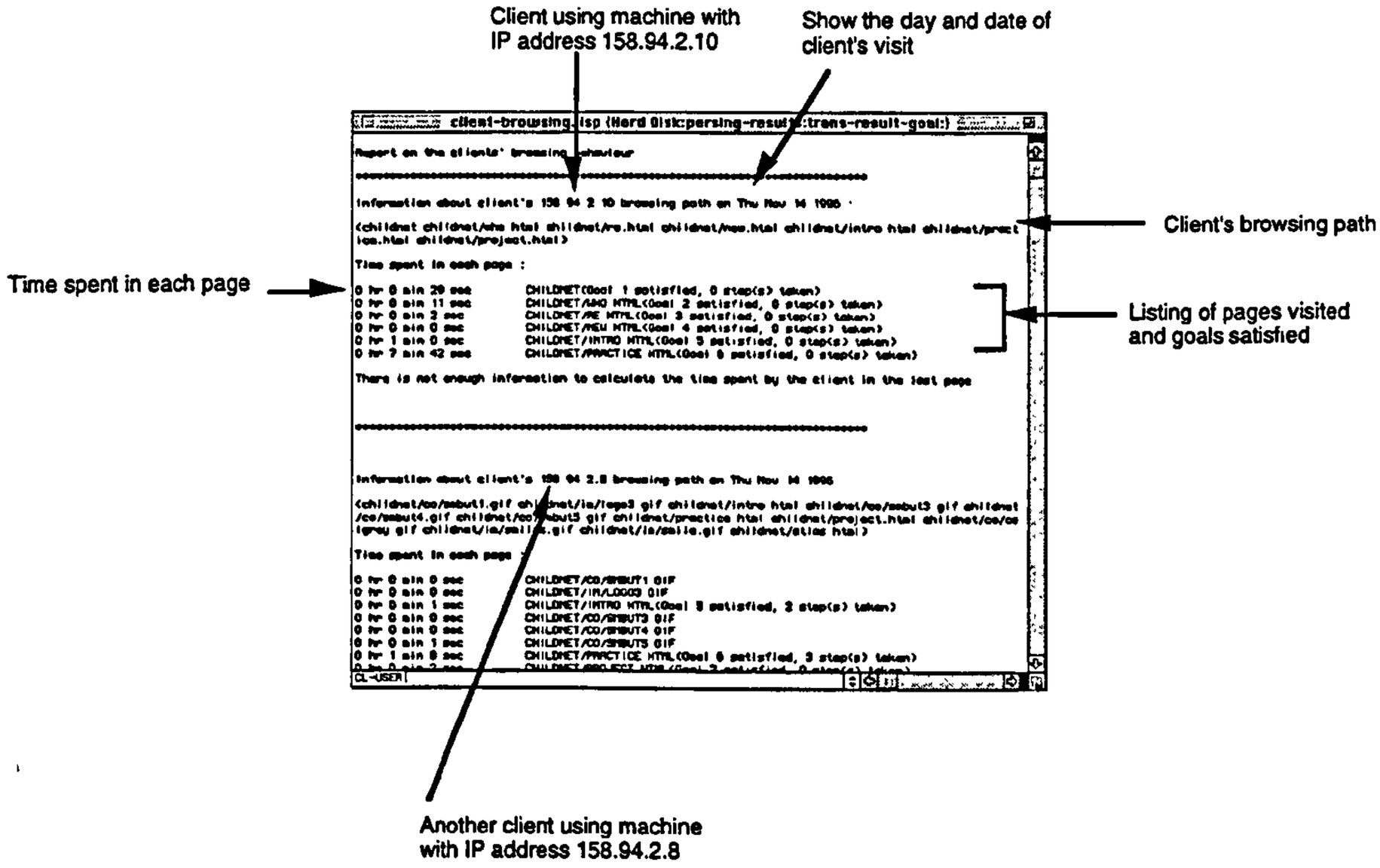


Figure A2.6. Report showing clients' browsing information

- **Comparison between the average number of steps taken by users**

This **Comparing steps taken** option can only be activated if the **Input goals** option is completed. This option calculates the average number of steps taken by users over a certain period for each goal and compares it with the minimum number of steps you think should be achieved. The results are printed in a report (see figure A2.7), which can be used to detect any discrepancies in the number of steps taken to achieve a certain goal.

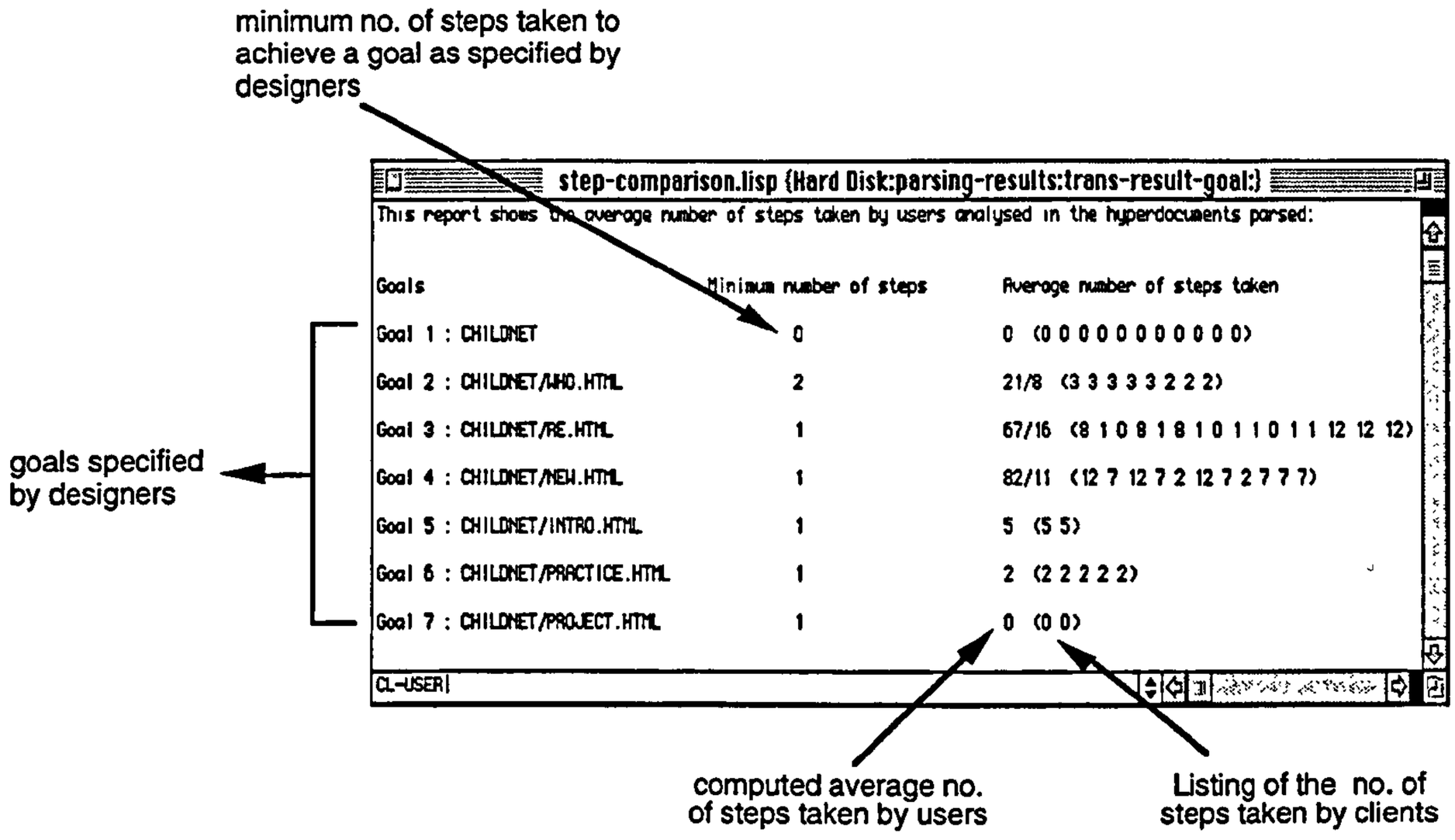


Figure A2.7. Report comparing the steps taken by users and the actual steps that should be taken

A2.3 Query...

You can also perform a query on the information of pages and clients visited. To do that, click on **Query** option in the **Parsing results** submenu (see figure A2.8).

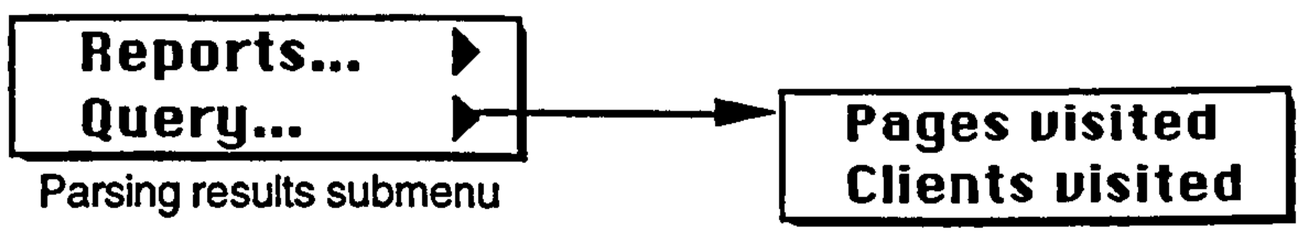


Figure A2.8. Query menu and submenu

- **A page visited**

Clicking onto **Pages visited** option opens up a dialog box showing a list of the pages that have been visited by clients over a period of time, captured in the transaction log file(s). The information is then displayed in the Display Window (see figure A2.9).

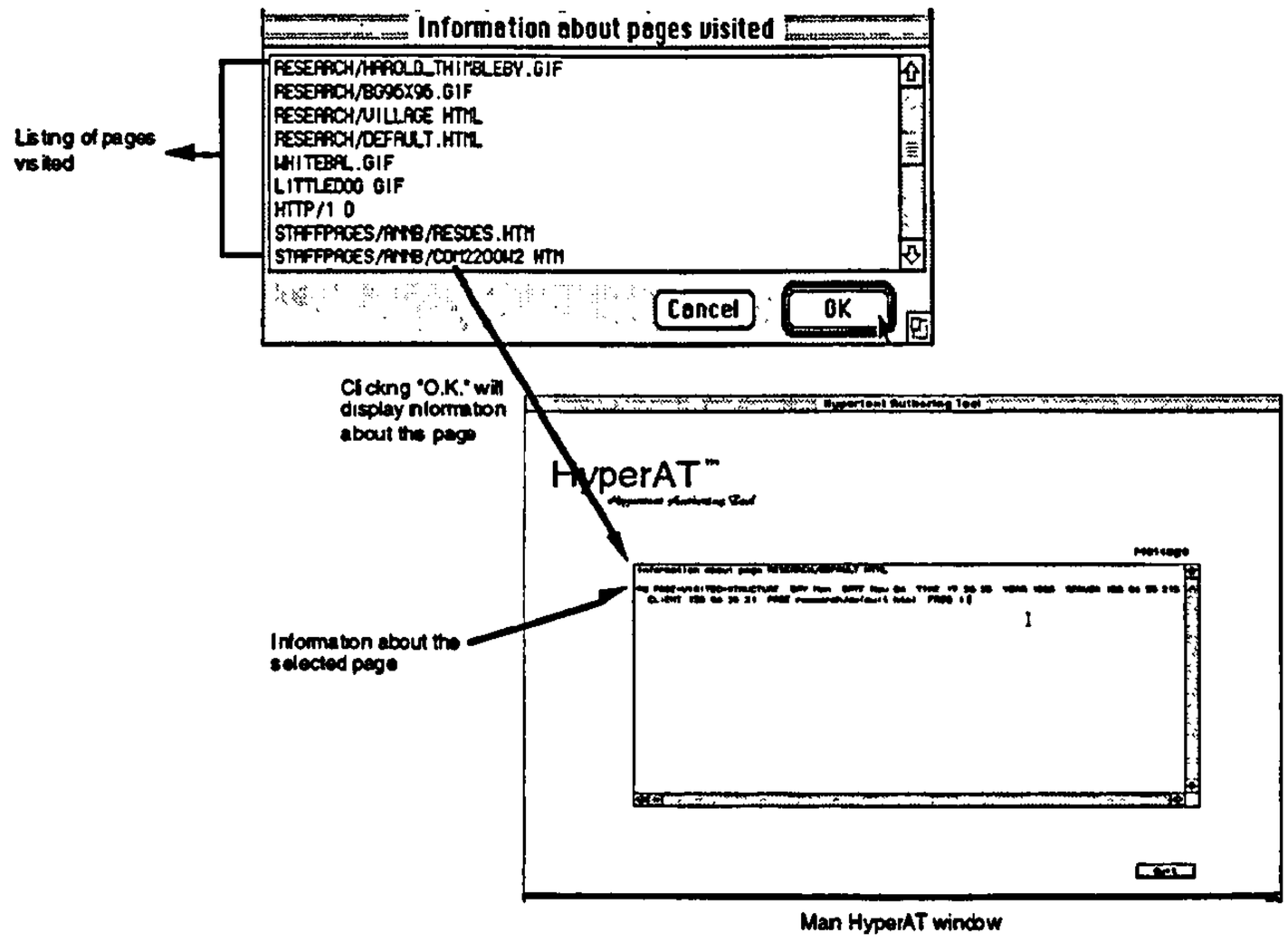


Figure A2.9. Selection and display of information of a page visited

- **A client's visit**

To find out more information about a particular client's browsing pattern, you can click on **Clients visited** option in the **Parsing results** submenu. To confirm your selection, click "O.K." and the information about the selected client is displayed in the Display Window (see figure A2.10).

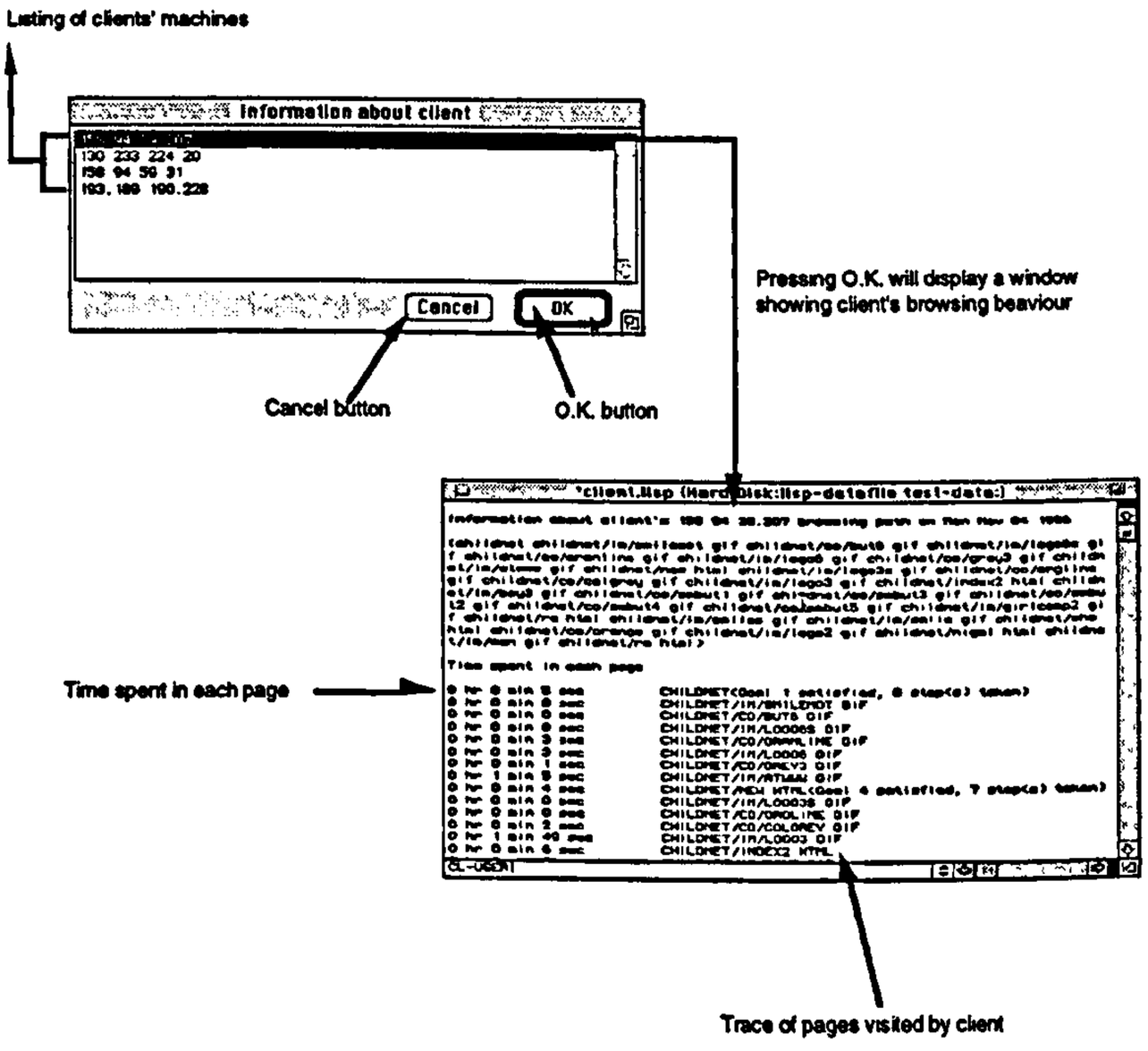


Figure A2.10. Selection and display of a client's browsing information

Appendix A3

HTML-Editor Menu Option

The HTML-Editor Menu[1] allows you to call out HTML Markup tags when creating the hyperdocument. Figure A3.1 below shows the HTML-Editor Menu and the associated submenus. The Markup tags are inserted in red. To toggle to Netscape, press the"control-shift-N" keys. To return to the HyperAT authoring environment, press the"control-alternate-command" keys. Use "command-R" keys to view the changes in Netscape.

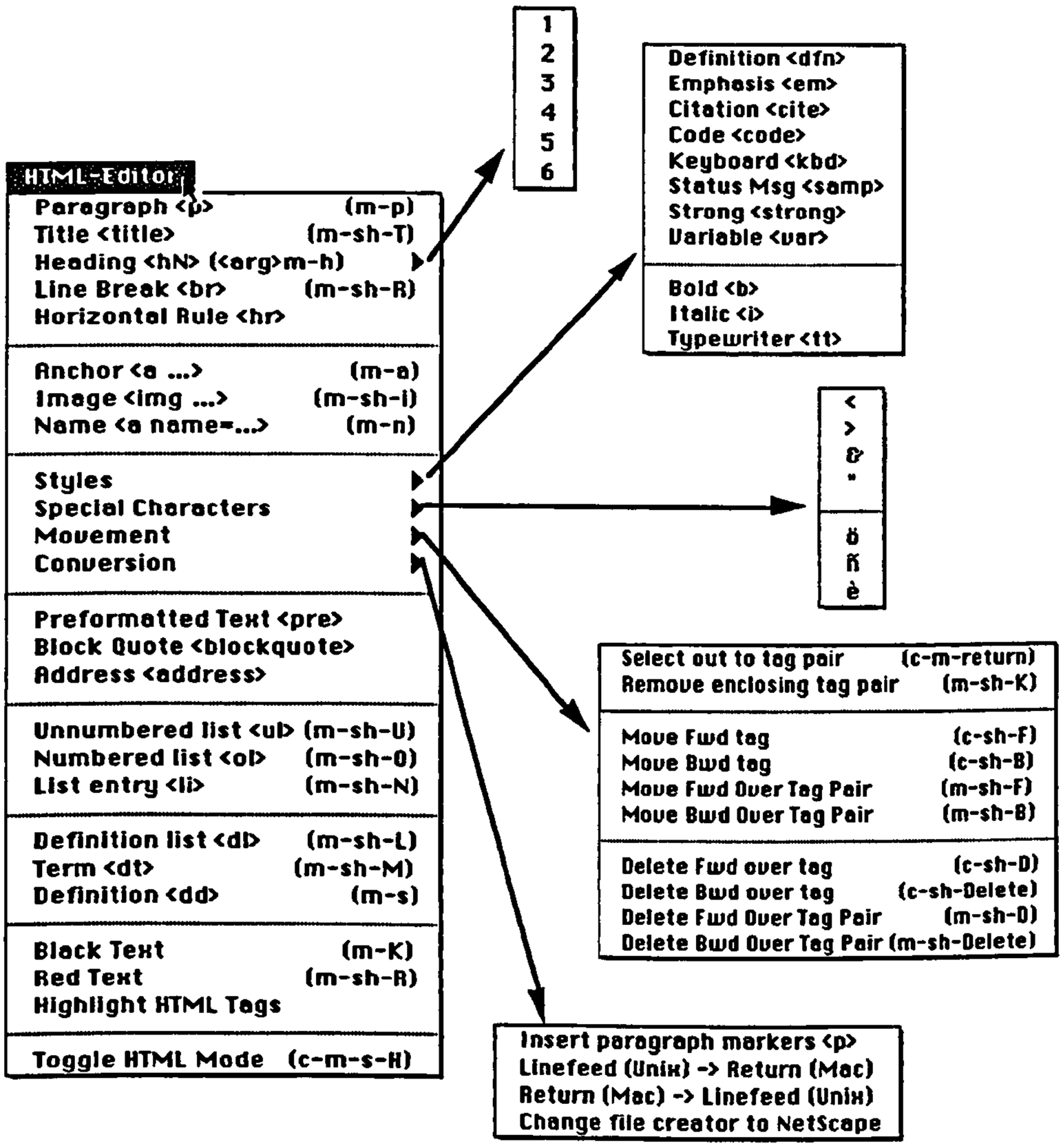


Figure A3.1. HTML-Editor Menu and submenus

[1]This menu was written by Bill St Clair from Apple Computer, Inc. and had been adapted for HyperAT. You can browse this website http://www.ncsa.uic.edu/demoweb/html-primer.html for further information.

We describe briefly below what each of these options does:

Option	Description
• *Paragraph <p>*	Insert a paragraph break.
• *Title <title>*	Insert a title spec.
• *Heading <hN>*	Insert a heading.
• *Line break *	Insert a line break.
• *Horizontal rule<hr>*	Insert a horizontal rule.
• *Anchor <a...>*	Bring a dialog to specify a Hypertext REFerence (HREF).
• *Image <img...>*	Bring a dialog to specify an image file.
• *Name <a name=...>*	Bring a dialog to specify a section name.
• *Styles*	Choose a text style.
• *Definition <dfn>*	For a word being defined. Typically displayed in italics.
• *Emphasis *	For emphasis. Typically displayed in italics.
• *Citation <cite>*	For titles of books, films, etc. Typically displayed in italics.
• *Code <code>*	For snippets of computer code. Displayed in a fixed-width font.
• *Keyboard <kbd>*	For user keyboard entry. Should be displayed in bold fixed-width font, but many browsers render it in plain fixed-width font.
• *Status Msg <samp>*	For computer status messages. Displayed in a fixed-width font.
• *Strong *	For strong emphasis. Typically displayed in bold.
• *Variable <var>*	For a 'metasyntactic' variable, where the user is to replace the variable with a specific instance. Typically displayed in italics.
• *Bold <b>*	Bold text.
• *Italic <i>*	Italic text.
• *Typewriter <tt>*	Typewriter text.
• *Special characters*	Insert special characters.
• *Movement*	Commands for moving the cursor or deleting text.
• *Select out to tag pair*	Move out to or over a tag pair.
• *Delete enclosing tag pair*	Remove the tag pair enclosing the selection.

Option	Description
• *Move Fwd tag*	Move forward to after a tag.
• *Move Bwd tag*	Move backward over a tag pair.
• *Move Fwd over tag pair*	Move forward over a tag pair.
• *Move Bwd over tag pair*	Move backward over a tag pair.
• *Delete Fwd over tag*	Delete between the cursor and the end of the next tag.
• *Delete Bwd over tag*	Delete between the cursor and the beginning of the previous tag.
• *Delete Fwd over tag pair*	Delete between the cursor and the end of the next tag pair.
• *Delete Bwd over tag pair*	Delete between the cursor and the beginning of the previous tag pair.
• *Conversion*	Commands for converting files to the format created by this editor.
• *Insert paragraph markers*	Change double newlines to paragraph markers in the selected region.
• *Linefeed (Unix)*	Change Unix style linefeed characters to Mac style return characters.
• *Return (Mac)*	Change Mac style return characters to Unix style linefeed characters.
• *Change file creator to Netscape*	Make Netscape the creator of the file. Double click on Finder to open it.
• *Preformatted text <pre>*	Enclose a block of preformatted text.
• *Block quote <blockquote>*	Enclose a quotation. Usually indented.
• *Address <address>*	For the address line at the bottom of the page. Usually formatted as 'name/email'.
• *Unnumbered list *	Enclose an unnumbered list. Lists may be nested.
• *Numbered list *	Enclose an numbered list. Lists may be nested.
• *List entry *	Prefix for a list entry.
• *Definition list <dl>*	Enclose a definition list.
• *Term <dt>*	A term to be defined.
• *Definition <dd>*	The definition of the term.
• *Black text*	Change the colour of the selection to black.
• *Red text*	Change the colour of the selection to red.
• *Highlight HTML tags*	Change the colour of the tags in the selection to red.

Appendix A4

File organisation in HyperAT

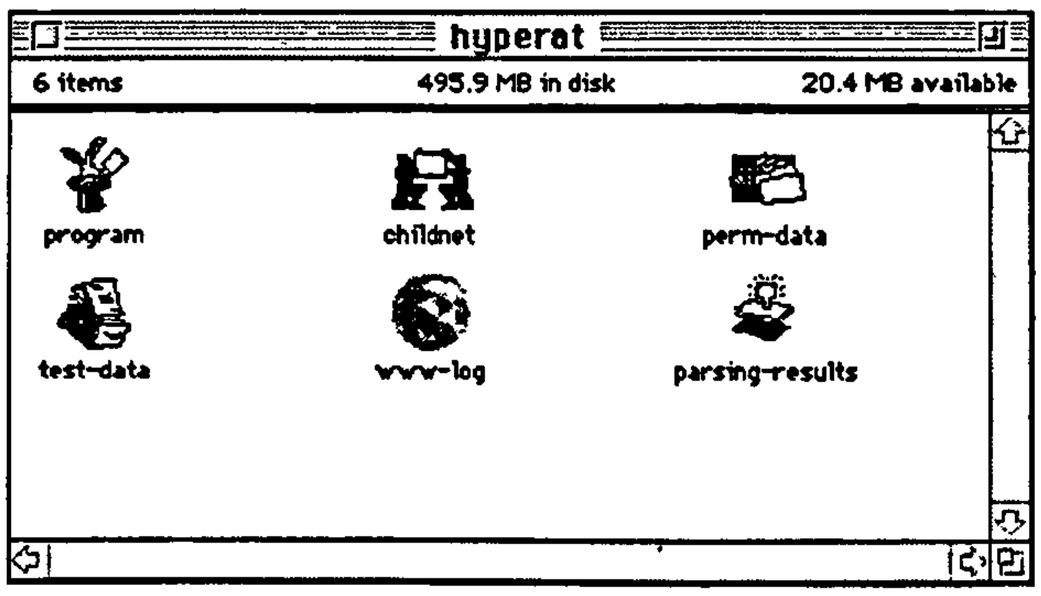


Figure A4.1. HyperAT folder

Figure A4.1 shows the HyperAT folder containing programs and data that are essential to execute HyperAT. These programs and data are kept in the folders described briefly below:

- *Program folder* contains all Lisp programs for running HyperAT.

- *Test-data folder* contains all files on the hyperdocument created. It also keeps the reports generated. These files, however, are temporary storage places and they are cleared at the start of a HyperAT session.

- *Childnet folder* contains a sample website on "Childnet International".

- *Web-log folder* contains a sample transaction log file.

- *Perm-data folder* contains all files on hyperdocuments that are to be kept permanently and can be re-loaded into HyperAT.

- *Parsing-results folder* contains reports after parsing web transaction log files.

Appendix B

Questionnaires

Appendix B1

Appendix B2

Appendix B3

Appendix B1

Instructions

Thank you for agreeing to take part in this experiment. Please take some time to complete this questionaire. Your comments will be invaluable in our investigation into good design guidelines, and 'correct' structure for hypertext systems.

Section 1: About the hypertext system

Name of hypertext system being evaluated: ***ACM's Hypertext-on-Hypertext***

On what platform is the system running on? ***Macintosh IIx***

Section 2: Your experience with computers

Your name: _____ Your email address: _____

How many years of experience have you had using PC/Macintosh? Approximately _____ years.

What do you use PC/Macintosh for? Please tick the appropriate function(s) and circle the type of user you think you are in carrying out these function(s).

___ Programming novice user / elementary user / intermediate user /advanced user

___ Wordprocessing novice user / elementary user / intermediate user /advanced user

___ Spreadsheets novice user / elementary user / intermediate user /advanced user

___ Databases novice user / elementary user / intermediate user /advanced user

___ Others. Please state: _____

Which type of systems do you prefer (tick your choice and state your reason(s))?

___ Command-line system. Reason(s)_____

_____

___ Menu-driven system. Reason(s) _____

_____

Have you gone through any hypertext system previously? Yes/No/Maybe

If yes, please indicate which: _____

Section 3: Your comments about ACM's Hypertext-on-hypertext

Tick the appropriate column to indicate your choice. For questions that do not give any options, please provide us with as much information as you possibly can.

3.1 Overall reactions to ACM's Hypertext-on-hypertext

1. How would you rate the performance of the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
terrible								wonderful

2. How would you rate the usability of the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

3. How would you rate your level of satisfaction when using the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
frustrating								satisfying

4. How would you rate the power of the system in helping you to complete your tasks?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
useless								powerful

5. How would you rate the appeal of the system in encouraging you to use the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
dull								stimulating

6. How flexible is the system in helping you to complete your tasks?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
rigid								flexible

3.2 Screen display

7. How easy are the characters on the computer screen to read?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
hard to read							.	easy to read

8. How does highlighting on the screen simplify the task?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unhelpful								helpful

9. How would you rate the organisation of information on the screen?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
confusing								clear

3.3 Terminology and System Information

10. Is the use of terminology, word and format consistent throughout the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
inconsistent								consistent

11. Does the computer terminology used relate to the task you are asked to do?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unrelated								related

12. Is the positioning of the messages on the screen consistent?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
inconsistent								consistent

13. Do you find prompts for input clear?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
confusing								clear

14. Does the system provide feedback about what it is doing?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unclear								clear

15. Are the error messages useful?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unhelpful								helpful

3.4 Learning

16. Do you find it easy to learn how to use the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

17. How easy is it for you to explore new features by trial and error?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

18. Do you find it easy remembering names and use of commands?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

3.5 System Capabilities and user control

19. Do you find the system responds fast enough?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
slow								fast

20. Do you find the system reliable (e.g., able to complete tasks without system breaking down, etc.)?

unreliable	extremely	quite	slightly	neutral	slightly	quite	extremely	reliable

21. How easy does the system enable you to correct your mistakes?

difficult	extremely	quite	slightly	neutral	slightly	quite	extremely	easy

22. Has the system taken both experienced and inexperienced users' needs into consideration?

untailored	extremely	quite	slightly	neutral	slightly	quite	extremely	tailored

3.6 Navigation

23. Are the links to other information within the system clearly displayed and highlighted?

unclear	extremely	quite	slightly	neutral	slightly	quite	extremely	clear

24. Are there sufficient information on the screen such as the use of title, subtitle, page numbers, etc. to help you know where you are in relation to the structure of the system?

insufficient	extremely	quite	slightly	neutral	slightly	quite	extremely	sufficient

25. Does the system provide sufficient information such as graphical document browsers, maps, document finders, table of contents, references, etc. to help you assess the structure of the system?

insufficient	extremely	quite	slightly	neutral	slightly	quite	extremely	sufficient

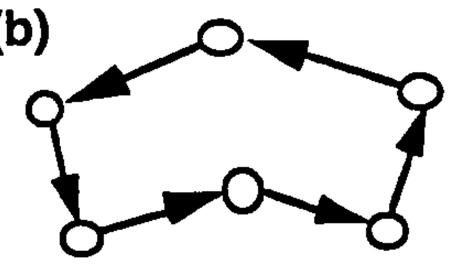
26. Below are 'maps' showing possible structure of the hypertext system (o represents node where node refers to a document that contains text/graphics/audio/video; ------> represents a one-directional link between two nodes, that is, it is the active reference that allows you to move to other parts of the hypertext system whenever you want to.)

Which 'map'/combination of 'maps' do you think best represents the structure of the hypertext system? Please circle your choice(s).

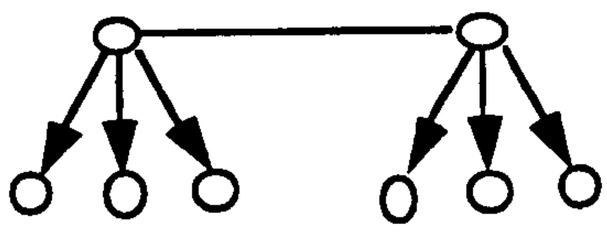
If none of the above 'maps'/combinations of 'maps' best represents the structure of the hypertext system, please draw what you perceive it to be.

(a)

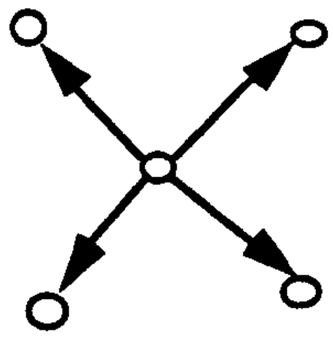
(b)



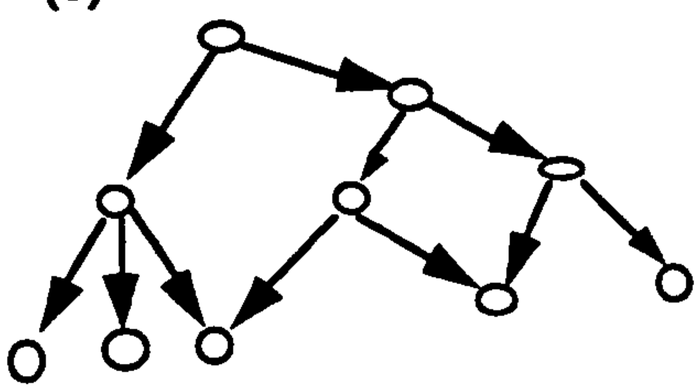
(c)



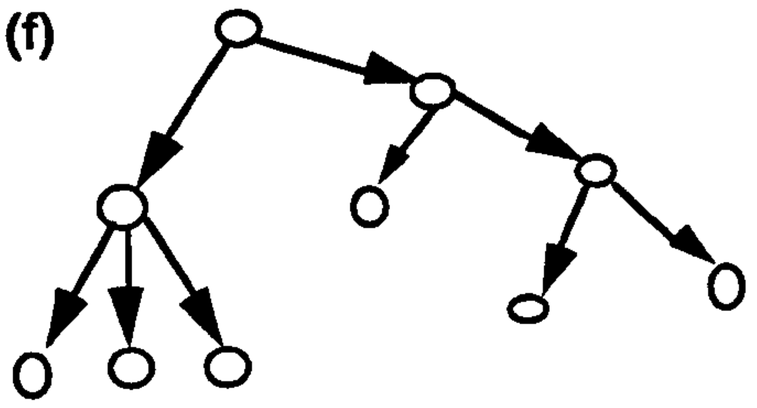
(d)



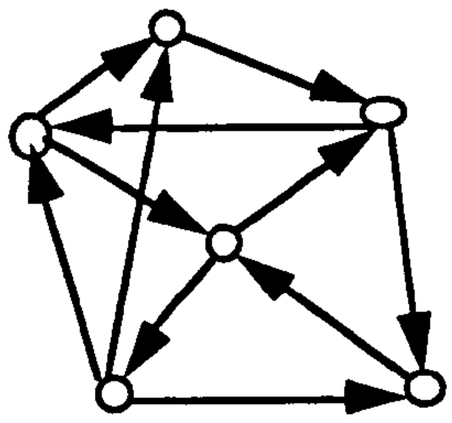
(e)



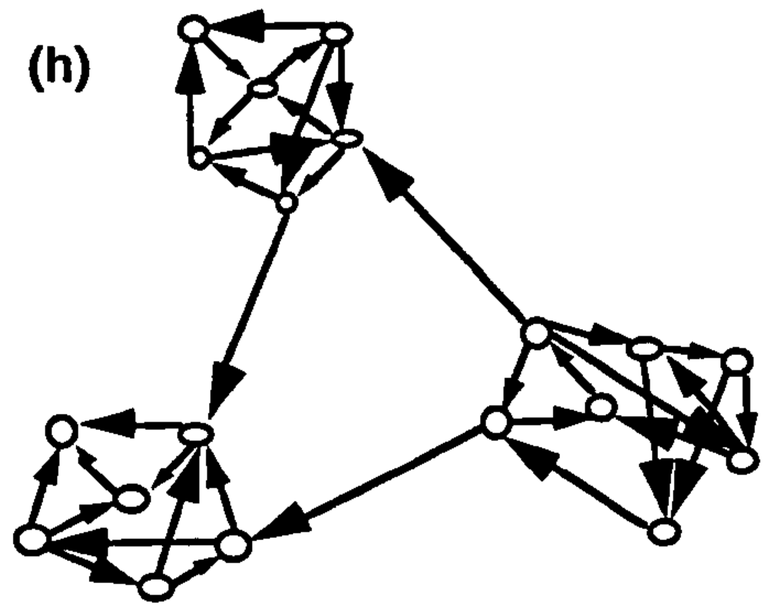
(f)



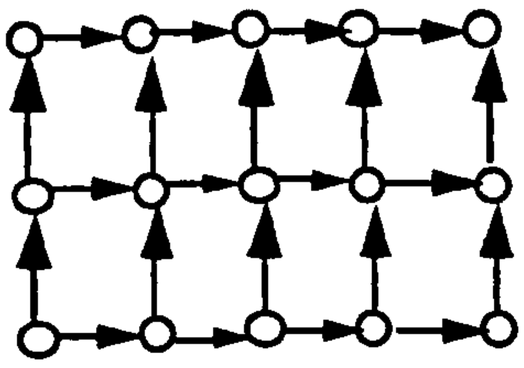
(g)



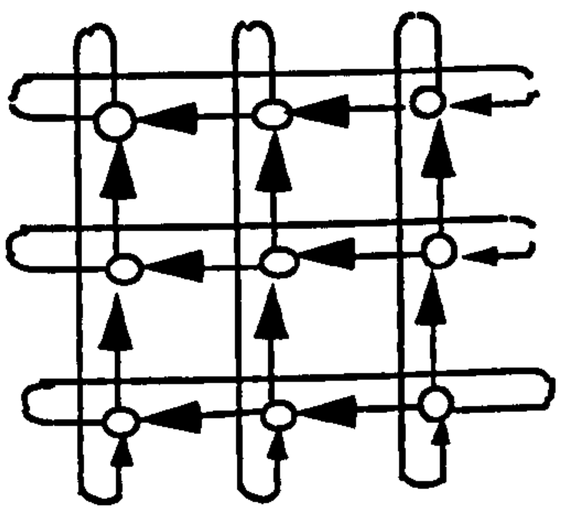
(h)



(i)



(j)



Your 'map' of the structure of the hypertext system.

27. Does the system provide good features/facilities to help you recall what you have browsed or the key points covered?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
poor								good

28. Is it difficult to identify where you are in the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

29. Is it difficult to find a particular topic thought to exist?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

30. Is it difficult to return to a topic left previously?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

31. Do you feel 'lost' when using the hypertext system? Yes/No/Sometimes
If no, please proceed to Question 33.

To what extent do you feel 'lost' when using the hypertext system?

	extremely	quite	slightly	
lost				

32. To what extent do you feel that being 'lost' in the hypertext system is a waste of time?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
waste of time								not a waste of time

3.7 Completing Tasks

33. How long did you take to complete your tasks? Approximately _____ minutes.

34. To what extent are you familiar with the system after completing the tasks with respect to :

a. *navigation tools (these refer to tools like 'Home', <----- , ------>, 'Index', etc.)*

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unfamiliar								familiar

b. *structure/'map' of the hypertext system (this refers to the way information is organised)*

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unfamiliar								familiar

c. *materials covered by the hypertext system (this refers to information/content)*

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unfamiliar								familiar

35. Which feature(s)/tool(s) of the system are useful in helping you complete your tasks such as browsing, seeking references, information search and revision?

Task	*Useful features/tools*
Browsing	_____
Seeking references	_____
Information search	_____
Revision	_____

36. How confident did you feel when using these features/tools to complete the tasks? (Confidence refers to the feeling that you are able to complete the tasks without any difficulty.)

Browsing

unsure	extremely	quite	slightly	neutral	slightly	quite	extremely	confident

Seeking references

unsure	extremely	quite	slightly	neutral	slightly	quite	extremely	confident

Information search

unsure	extremely	quite	slightly	neutral	slightly	quite	extremely	confident

Revision

unsure	extremely	quite	slightly	neutral	slightly	quite	extremely	confident

37. Do you have the feeling that you may be overlooking crucial information when completing the tasks?

____ Yes. Please state reason(s): _____

_____

____ No. Please state reason(s): _____

_____

38(a) What are 3 things you like about the system?

a. _____

b. _____

c. _____

38(b) List 3 features/tools you wish the system would provide. Rank your choices in order of preference.

a. _____

b. _____

c. _____

39. How would you rate the effectiveness of the system in helping you complete the tasks?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
poor								good

40. Other comments.

_____

_____

_____

_____

_____

Thank you. End of Questionaire.

Appendix B2

Instructions

Thank you for agreeing to take part in this experiment. Please take some time to complete this questionaire. Your comments will be invaluable in our investigation into good design guidelines and principles, and 'correct' structure for hypertext systems.

Section 1: About the hypertext system

Name of hypertext system being evaluated: _____

On what platform is the system running on? ***Macintosh IIx***

Section 2: Your experience with computers

Your name: _____ Your email address: _____

How many years of experience have you had using PC/Macintosh? Approximately _____ years.

What do you use PC/Macintosh for? Please tick the appropriate function(s) and circle the type of user you think you are in carrying out these function(s).

___ Programming novice user / elementary user / intermediate user /advanced user

___ Wordprocessing novice user / elementary user / intermediate user /advanced user

___ Spreadsheets novice user / elementary user / intermediate user /advanced user

___ Databases novice user / elementary user / intermediate user /advanced user

___ Others. Please state: _____

Which type of systems do you prefer (tick your choice and state your reason(s))?

___ Command-line system. Reason(s) _____

_____

___ Menu-driven system. Reason(s) _____

_____

Have you gone through any hypertext system previously? Yes/No/Maybe

If yes, please indicate which: _____

Section 3: Your comments about the hypertext system

Tick the appropriate column to indicate your choice. For questions that do not give any options, please provide us with as much information as you possibly can.

3.1 Overall reactions to ACM's Hypertext-on-hypertext

1. How would you rate the performance of the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
terrible								wonderful

2. How would you rate the usability of the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

3. How would you rate your level of satisfaction when using the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
frustrating								satisfying

3.2 Learning

4. Do you find it easy to learn how to use the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

5. How easy is it for you to explore new features by trial and error?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

6. Do you find it easy remembering names and use of commands?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

3.3 Navigation

7. Are the links to other information within the system clearly displayed and highlighted?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unclear								clear

8. Are there sufficient information on the screen such as the use of title, subtitle, page numbers, etc. to help you know where you are in relation to the structure of the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
insufficient								sufficient

9. Does the system provide sufficient information such as graphical document browsers, maps, document finders, table of contents, references, etc. to help you assess the structure of the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
insufficient	•							sufficient

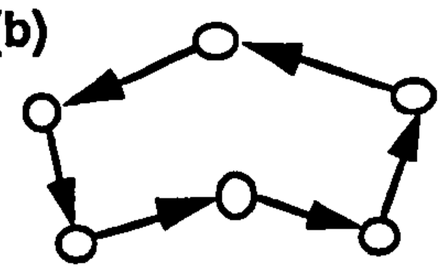
10. Below are 'maps' showing possible structure of the hypertext system (o represents node where node refers to a document that contains text/graphics/audio/video; ------> represents a one-directional link between two nodes, that is, it is the active reference that allows you to move to other parts of the hypertext system whenever you want to.)

Which 'map'/combination of 'maps' do you think best represents the structure of the hypertext system? Please circle your choice(s).

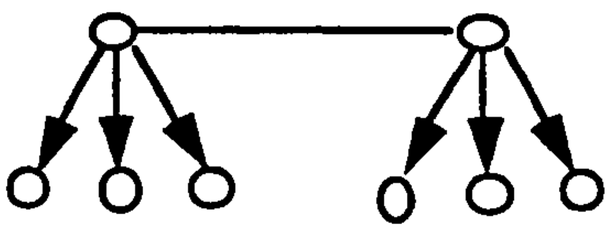
If none of the above 'maps'/combinations of 'maps' best represents the structure of the hypertext system, please draw what you perceive it to be.

(a)

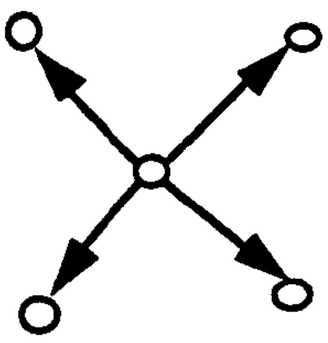
(b)



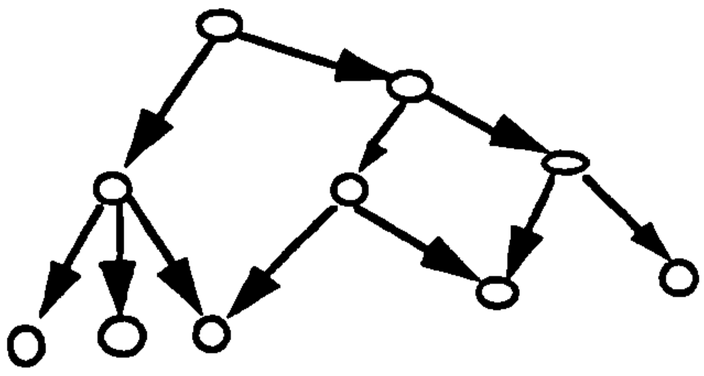
(c)



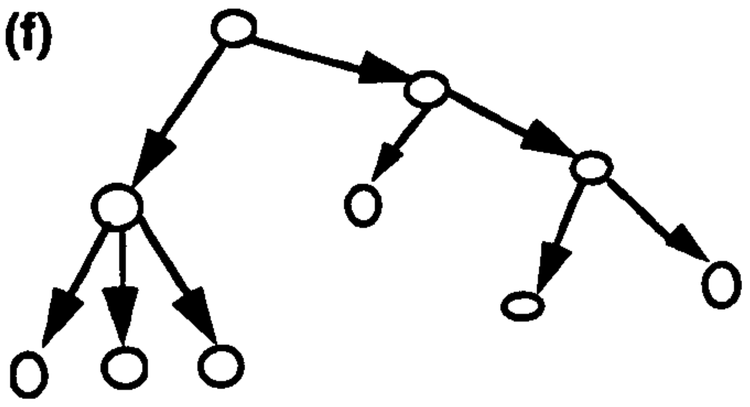
(d)



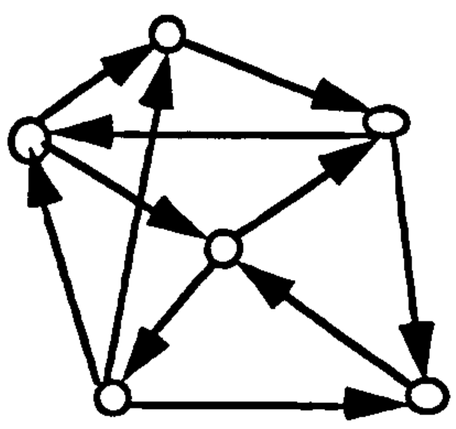
(e)



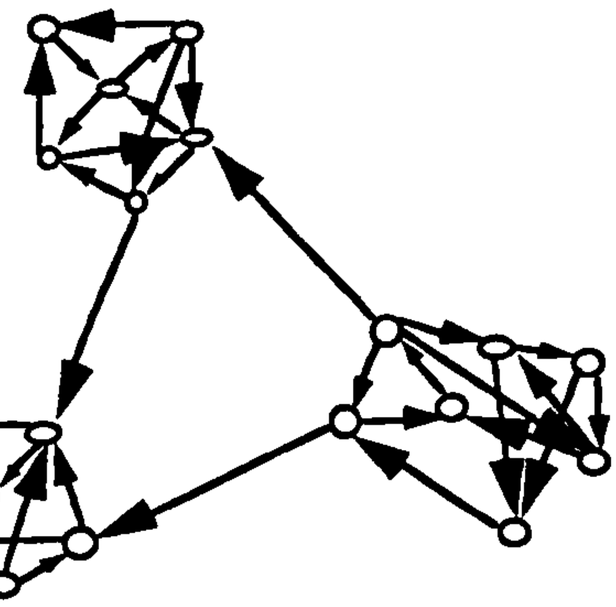
(f)



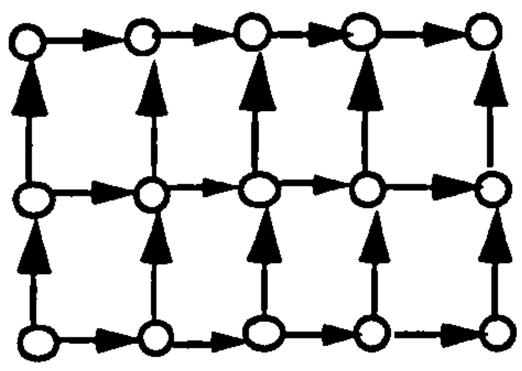
(g)



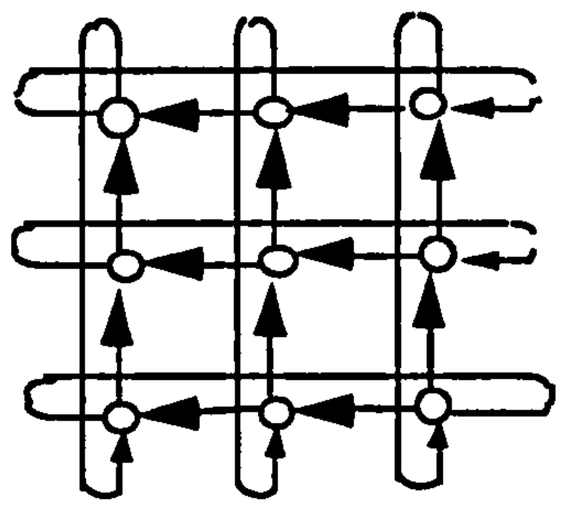
(h)



(i)



(j)



Your 'map' of the structure of the hypertext system.

11. Does the system provide good features/facilities to help you recall what you have browsed or the key points covered?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
poor								good

12. Is it difficult to identify where you are in the system?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

13. Is it difficult to find a particular topic thought to exist?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

14. Is it difficult to return to a topic left previously?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
difficult								easy

15. Do you feel 'lost' when using the hypertext system? Yes/No/Sometimes
If no, please proceed to Question 11.

To what extent do you feel 'lost' when using the hypertext system?

	extremely	quite	slightly
lost			

16. To what extent do you feel that being 'lost' in the hypertext system is a waste of time?

	extremely	quite	slightly	neutral	slightly	quite	extremely	
waste of time								not a waste of time

3.4 Completing Tasks

17. How long did you take to complete your tasks? Approximately _____ minutes.

18. To what extent are you familiar with the system after completing the tasks with respect to :

a. *navigation tools (these refer to tools like 'Home', <----- , ------>, 'Index', etc.)*

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unfamiliar								familiar

b. *structure/'map' of the hypertext system (this refers to the way information is organised)*

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unfamiliar								familiar

c. *materials covered by the hypertext system (this refers to information/content)*

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unfamiliar								familiar

19. Which feature(s)/tool(s) of the system are useful in helping you complete your tasks such as browsing, seeking references, information search and revision?

Task	*Useful features/tools*
Browsing	_____
Seeking references	_____
Information search	_____
Revision	_____

20. How confident did you feel when using these features/tools to complete the tasks? (Confidence refers to the feeling that you are able to complete the tasks without any difficulty.)

Browsing

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unsure								confident

Seeking references

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unsure								confident

Information search

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unsure								confident

Revision

	extremely	quite	slightly	neutral	slightly	quite	extremely	
unsure								confident

21. Do you have the feeling that you may be overlooking crucial information when completing the tasks?

____ Yes. Please state reason(s): _____

_____

____ No. Please state reason(s): _____

_____

22(a) What are 3 things you like about the system?

a. _____

b. _____

c. _____

22(b) List 3 features/tools you wish the system would provide. Rank your choices in order of preference.

a. _____

b. _____

c. _____

23. How would you rate the effectiveness of the system in helping you complete the tasks?

poor	extremely	quite	slightly	neutral	slightly	quite	extremely	good

24. Other comments.

_____

_____

_____

_____

Thank you. End of Questionaire.

Appendix B3

Instructions

Please take some time to complete the questionaire. Your comments will be invaluable in our investigation into desirable features for hypertext designer tools.

Section 1: Your experience with computers and hypertext systems

Your name: _____ Your email address: _____

How many years of experience have you had using PC/Macintosh? Approximately _____ years.

What do you use PC/Macintosh for? Please tick the appropriate function(s) and circle the type of user you think you are in carrying out these function(s).

___ Programming novice user / elementary user / intermediate user /advanced user

___ Wordprocessing novice user / elementary user / intermediate user /advanced user

___ Spreadsheets novice user / elementary user / intermediate user /advanced user

___ Databases novice user / elementary user / intermediate user /advanced user

___ Others. Please state: _____

_____

Which type of systems do you prefer (tick your choice and state your reason(s))?

___ Command-line system. Reason(s) _____

_____

___ Menu-driven system. Reason(s) _____

_____

Have you gone through any hypertext system previously? Yes/No/Maybe

If yes, please indicate which: _____

Section 2: Your comments about authoring tools for hypertexts

1. Have you used any hypertext authoring tool previously? Yes/No
   If no, please go to Question 3.
   If yes, please indicate which authoring tool(s).

   _____

   _____

2. How pleased are you with the support provided by the authoring tool(s) in helping you develop the hypertext system? Very pleased/Partially pleased/Not pleased

   If 'very pleased', please give reason(s):

   _____

   _____

   _____

   _____

   _____

   If 'partially pleased/not pleased', please describe some problems you have encountered and then list 3 most important features you think the authoring tool should have (rank them in order of importance; 1 being the most important):

   Problems: _____

   _____

   _____

   _____

   Features: 1. _____

   2. _____

   3. _____

3. Which of the features/tools do you think hypertext authoring tools should provide to assist hypertext authors in developing efficient and effective hypertexts? Please tick your choice(s) and also circle how essential these features/tools are.

____	text editing and wordprocessing capabilities	essential / helpful / partially helpful
____	import and conversion capabilities	essential / helpful / partially helpful
____	graphic editing capbilities	essential / helpful / partially helpful
____	mode switching (from author mode to user mode)	essential / helpful / partially helpful
____	built-in functions for node and link listing	essential / helpful / partially helpful
____	built-in functions to provide node information	essential / helpful / partially helpful
____	built-in tools for annotating nodes	essential / helpful / partially helpful
____	tools for outlining	essential / helpful / partially helpful
____	tools to group related groups of information	essential / helpful / partially helpful
____	tools for locating specific terms or phrases	essential / helpful / partially helpful
____	tools to link checking	essential / helpful / partially helpful
____	tools for drawing overall map and structure	essential / helpful / partially helpful
____	tools for indexing	essential / helpful / partially helpful
____	tools for generating return paths of unidirectional links	essential / helpful / partially helpful
____	tools that link to external programs or peripherals	essential / helpful / partially helpful

4. What other features/tools do you think hypertext authoring tools should provide in helping hypertext authors develop more efficient and effective hypertexts?

Section 3: Design considerations for effective hypertext systems

5. If you were asked to author a hypertext system and also ensure that it is effective, what do you think are the 3 most important qualities effective hypertext systems should have?

a. _____

_____

b. _____

_____

c. _____

_____

Other Comments

_____

_____

_____

_____

_____

Thank you. End of Questionaire.

Appendix C

Tasks performed by subjects

Appendix C1

Appendix C2

Appendix C1

ACM's Hypertext-on-Hypertext **Name:** _____

Instructions

Please answer all questions as completely as you possibly can.

Section 1: Browsing

1. The "Map" page provides information on links between documents. Name one paper/document which gives more information on "Navigation Problems".

---

2. There are 26 items indexed in this hypertext system. Put a cross (X) against the item(s) not found in the index.

_____ SuperCard

_____ NoteCards

_____ Medicine

_____ User issues

_____ FRESS

_____ Tasks and structures

_____ Authors

_____ Hypermedia

Section 2: Information Search

3(a). In the paper by Mark Edwin Frisse on "Searching for information in a hypertext medical handbook", he described the *Dynamic Medical Handbook* project that attempted to apply the memex concept to medical information management in 2 ways. What are these 2 ways?

a.

---

b.

4(b). How many references are cited with regards to the *Dynamic Medical Handbook* project?

Of the references given, how many are by Thursh, D.? _____

Section 3: Seeking References

5. From the definition of hypertext, what is *one* thing you can say about the structure of hypertexts?

---

6. Neptune, a workstation-based hypermedia system, is described in a paper by Delisle, N. and Schwartz, M. Who developed Neptune?

Section 4: Revision (Recall)

7. Name the tools that are available in ACM's Hypertext-on-Hypertext that help you revise what you have covered.

Appendix C2

Basic Computer Anatomy **Name:** _____

(with design guidelines)

Instructions

Please answer all questions as completely as you possibly can.

Section I: Browsing

Questions	How did you find the answer?
1. How many types of input devices can be used to interact with the computer?	
2. Name 4 types of serial printers that are used with microcomputers. a. _____ b. _____ c. _____ d. _____	

Section II: Search

3. Digitized sound is a form of audio output from a computer. What are digitized sounds?	
4. Explain what ROM is.	

Appendix C2

Basic Computer Anatomy

(without design guidelines)

Name: _____

Instructions

Please answer all questions as completely as you possibly can.

Section I: Browsing

Questions	How did you find the answer?
1. Name 3 common screen sizes. a. _____ b. _____ c. _____	
2. What does this button do?	

Section II: Search

3. What is the function of the front-end processor?	
4. List 2 advantages of thermal printers over dot-matrix printers.	

Appendix C2

Basic Computer Anatomy **Name:** _____

(with guidelines)

Instructions

Please answer all questions as completely as you possibly can.

Section III: Revision

Questions	How did you find the answer?
1. Name 3 common screen sizes. a. _____ b. _____ c. _____	
2. How many types of input devices can be used to interact with the computer?	
3. Digitized sound is a form of audio output from a computer. What are digitized sounds?	
4. List 2 advantages of thermal printers over dot-matrix printers.	

Appendix C2

Basic Computer Anatomy **Name:** _____

(without guidelines)

Instructions

Please answer all questions as completely as you possibly can.

Section III: Revision

Questions	**How did you find the answer?**
1. What does this button do?	
2. Name 4 types of serial printers that are used with microcomputers. a. _____ b. _____ c. _____ d. _____	
3. What is the function of the front-end processor?	
4. Explain what ROM is.	

Appendix D

Task diagrams for recall

Recall

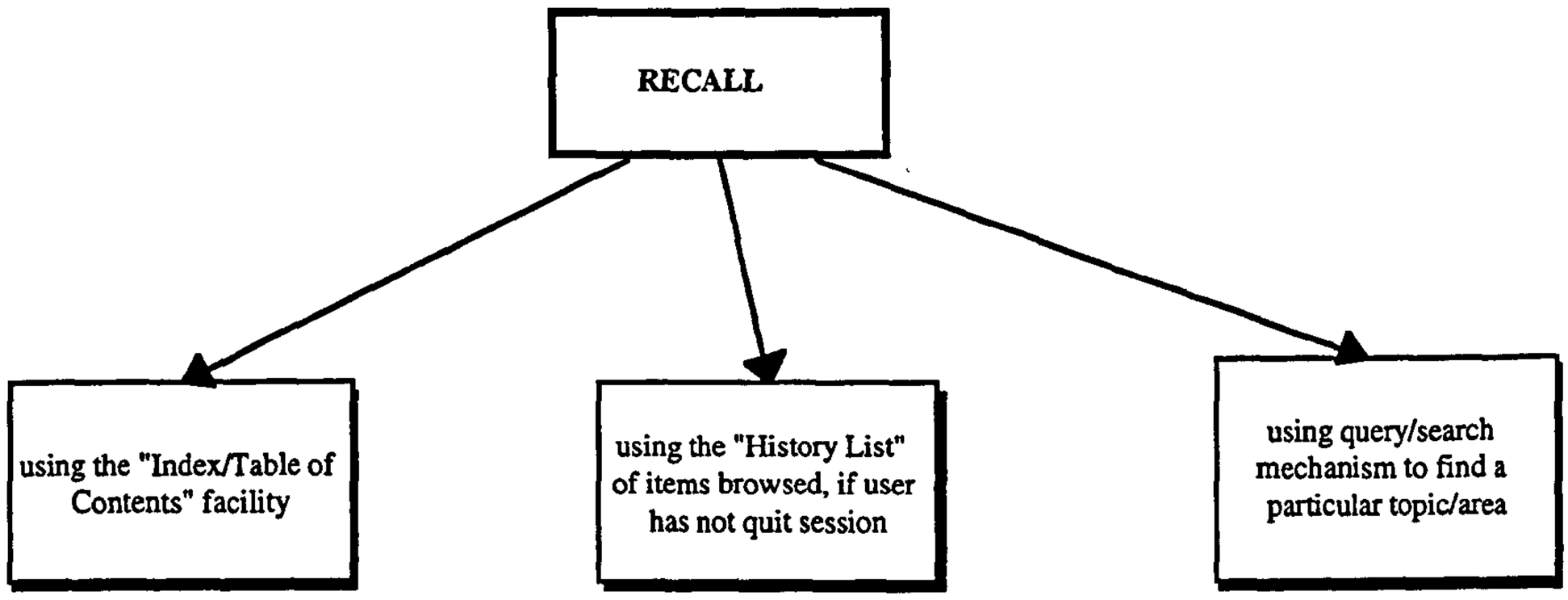


Figure D1. Task diagram for revision

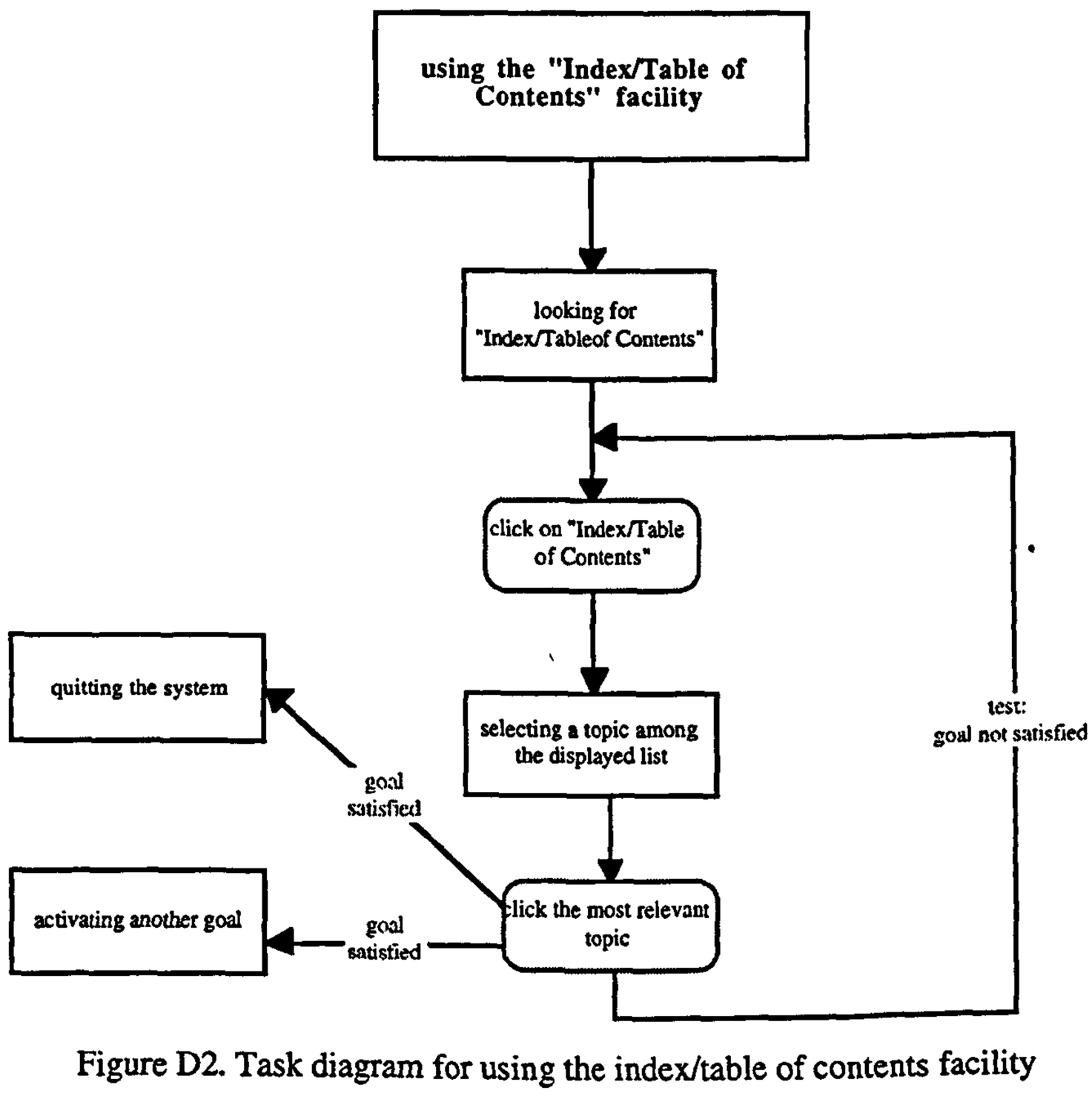


Figure D2. Task diagram for using the index/table of contents facility

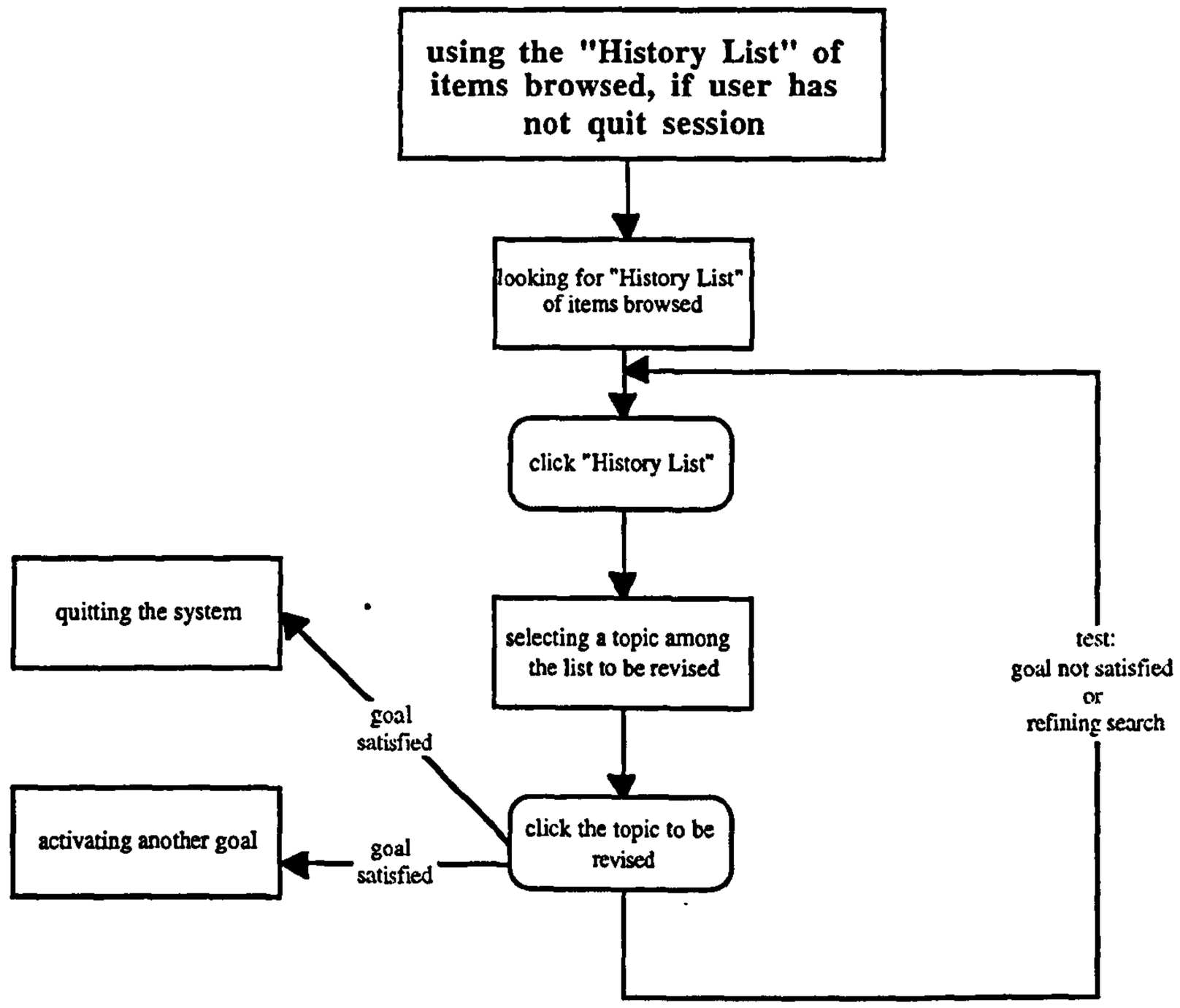


Figure D3. Task diagram for using the "History List"

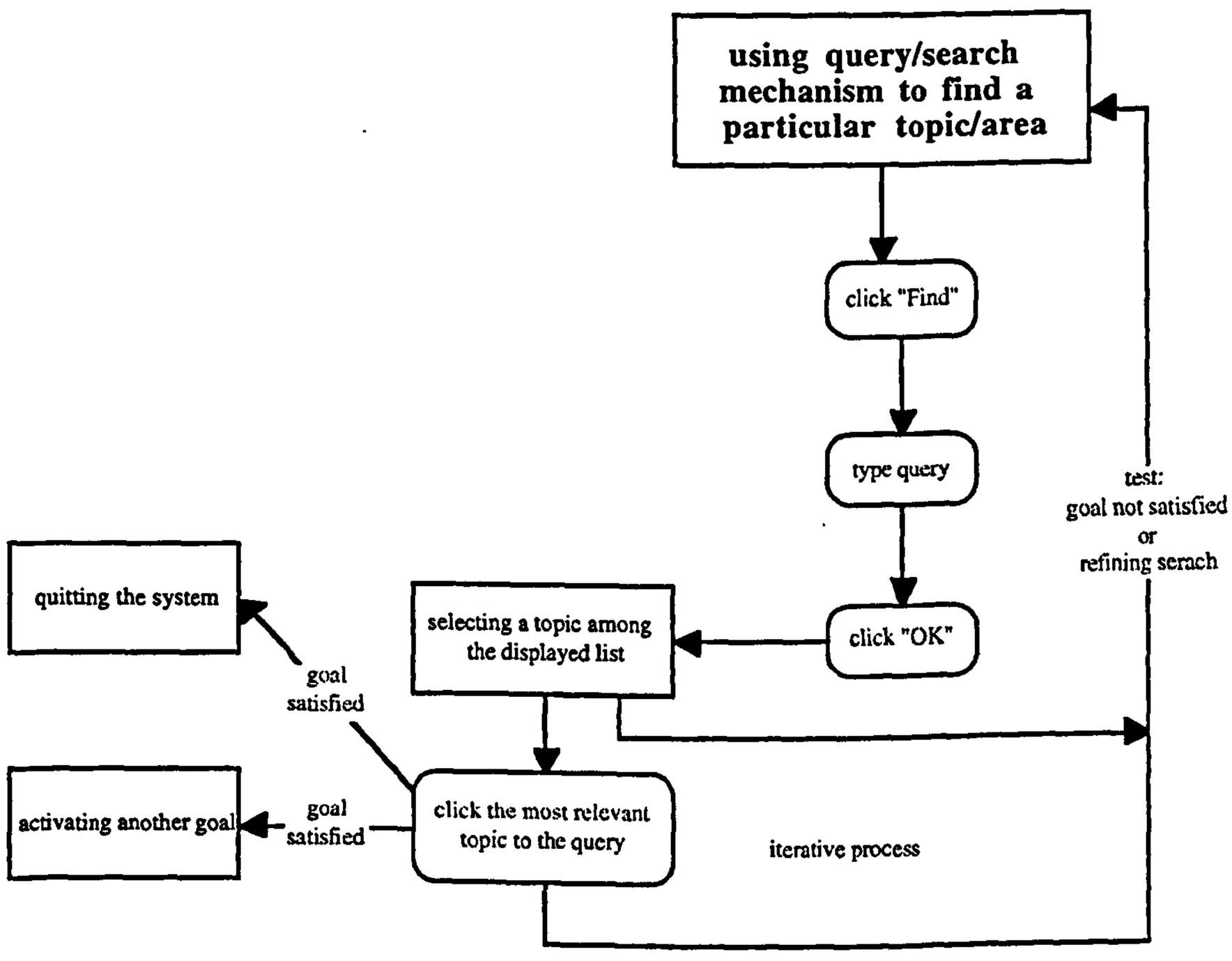
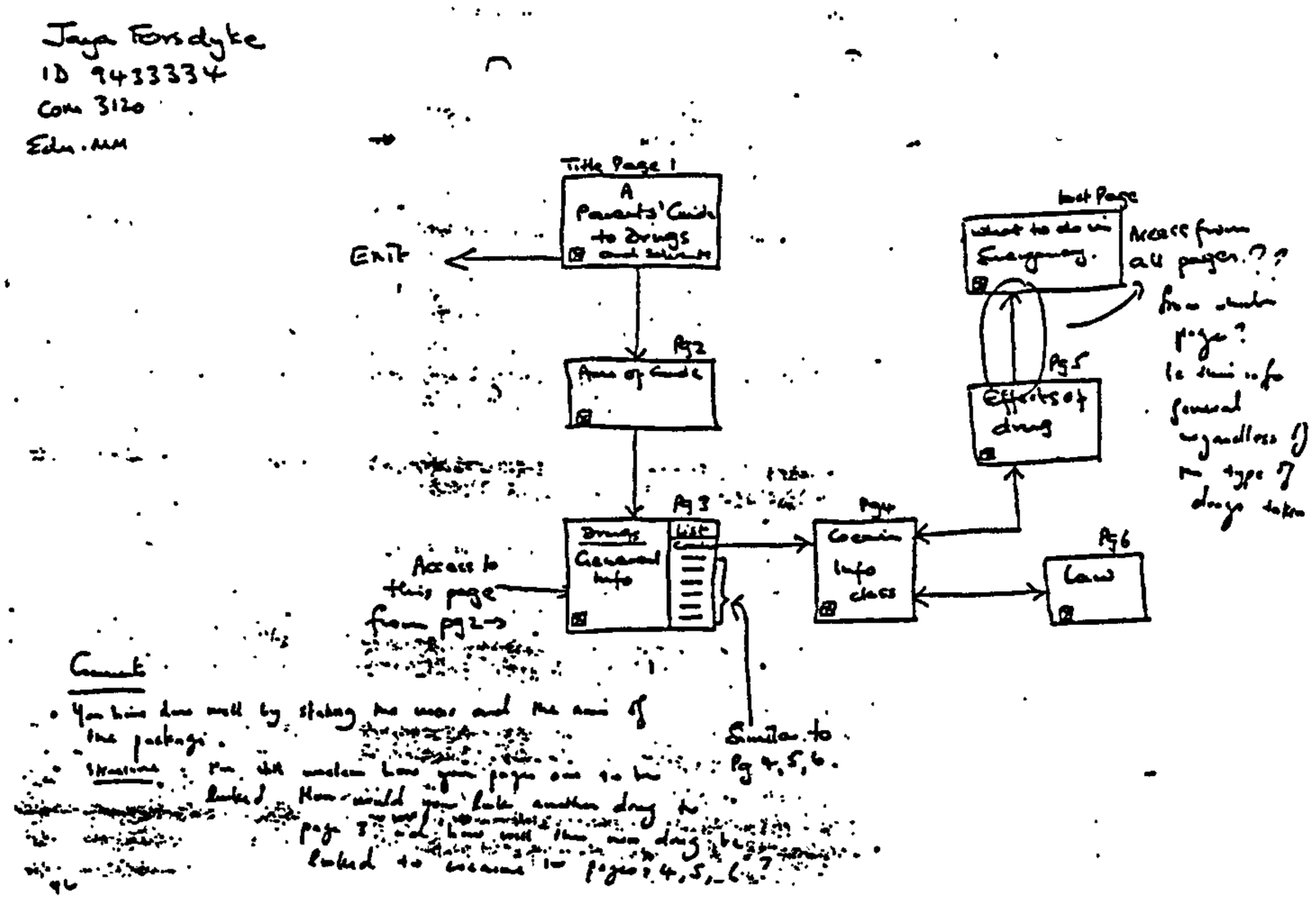


Figure D4. Task diagram for using the query/search mechanism to find a particular topic/area

Appendix E

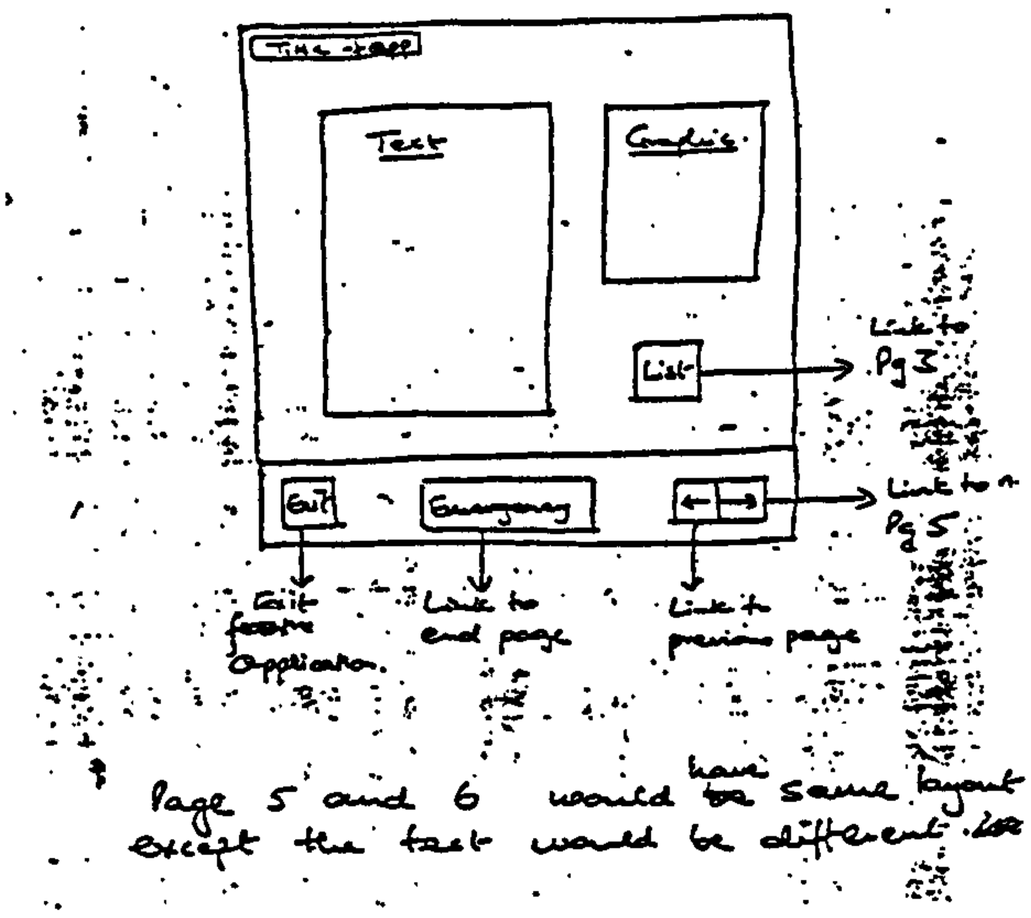
Structures

Student 1



Jaya Forsdyke
ID 9433334
Com 3120-Ed Mc

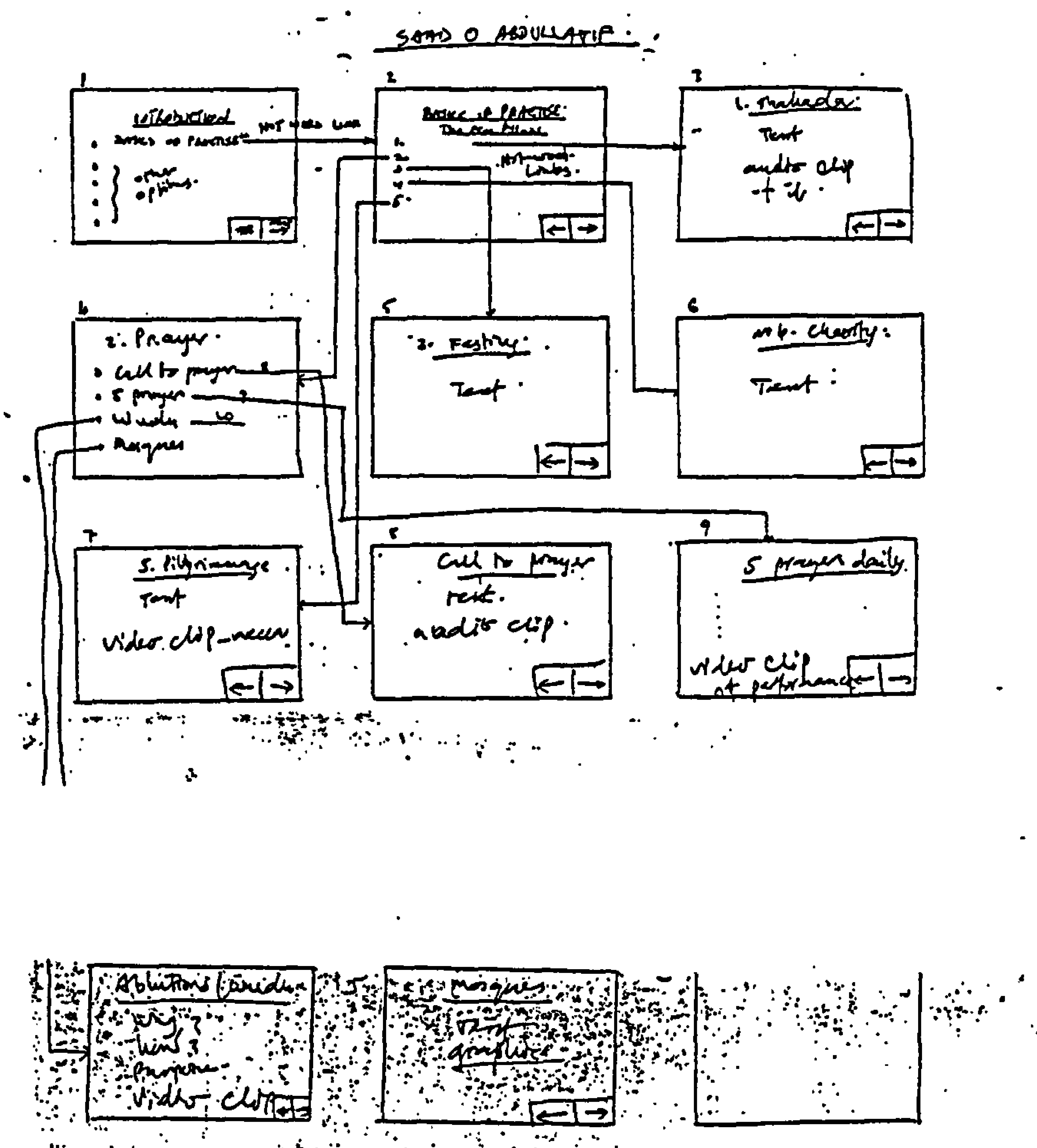
Approximate layout of Pg 4



Page 5 and 6 would have same layout except the text would be different.

All other drugs would have above layout.

Student 2

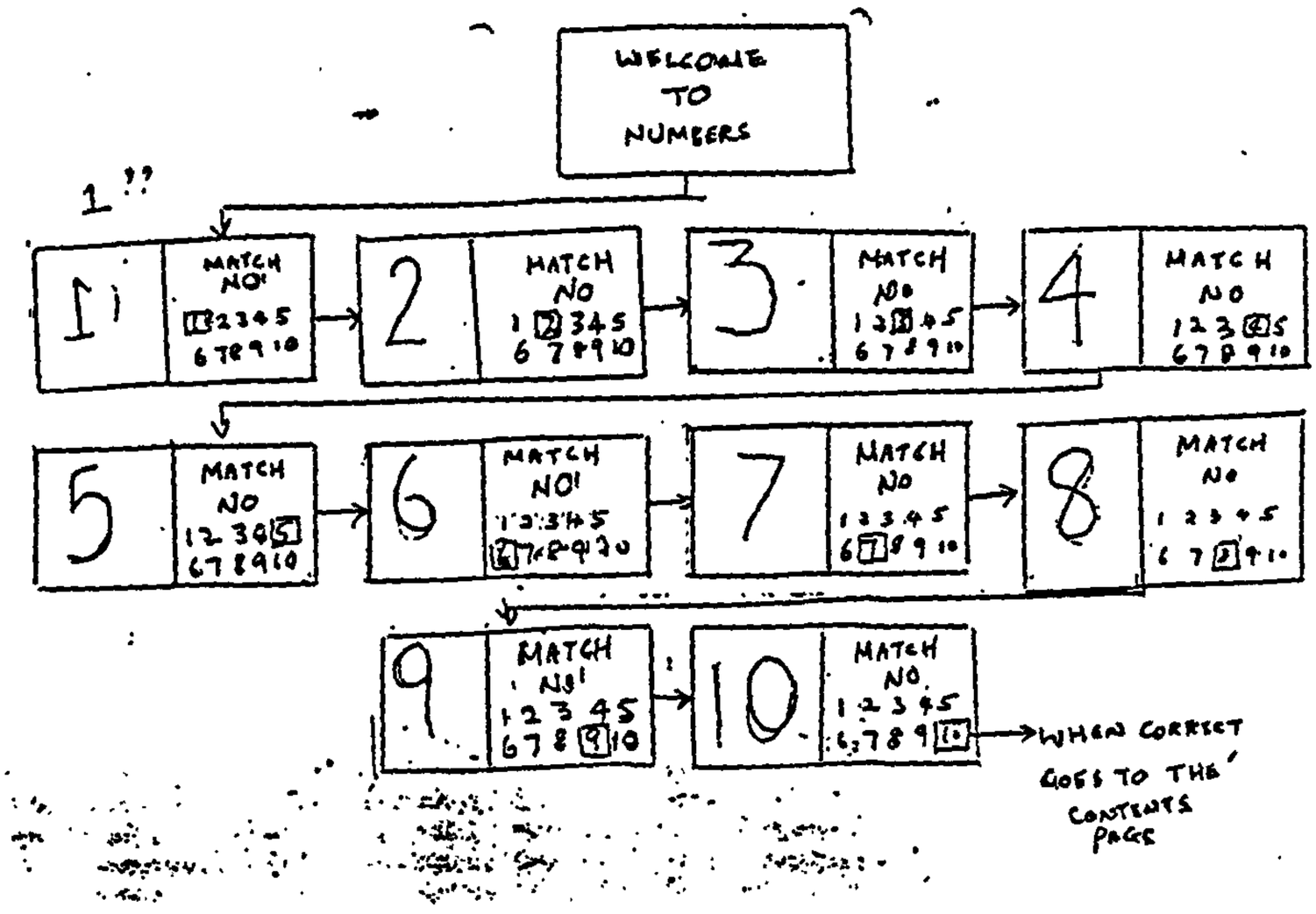


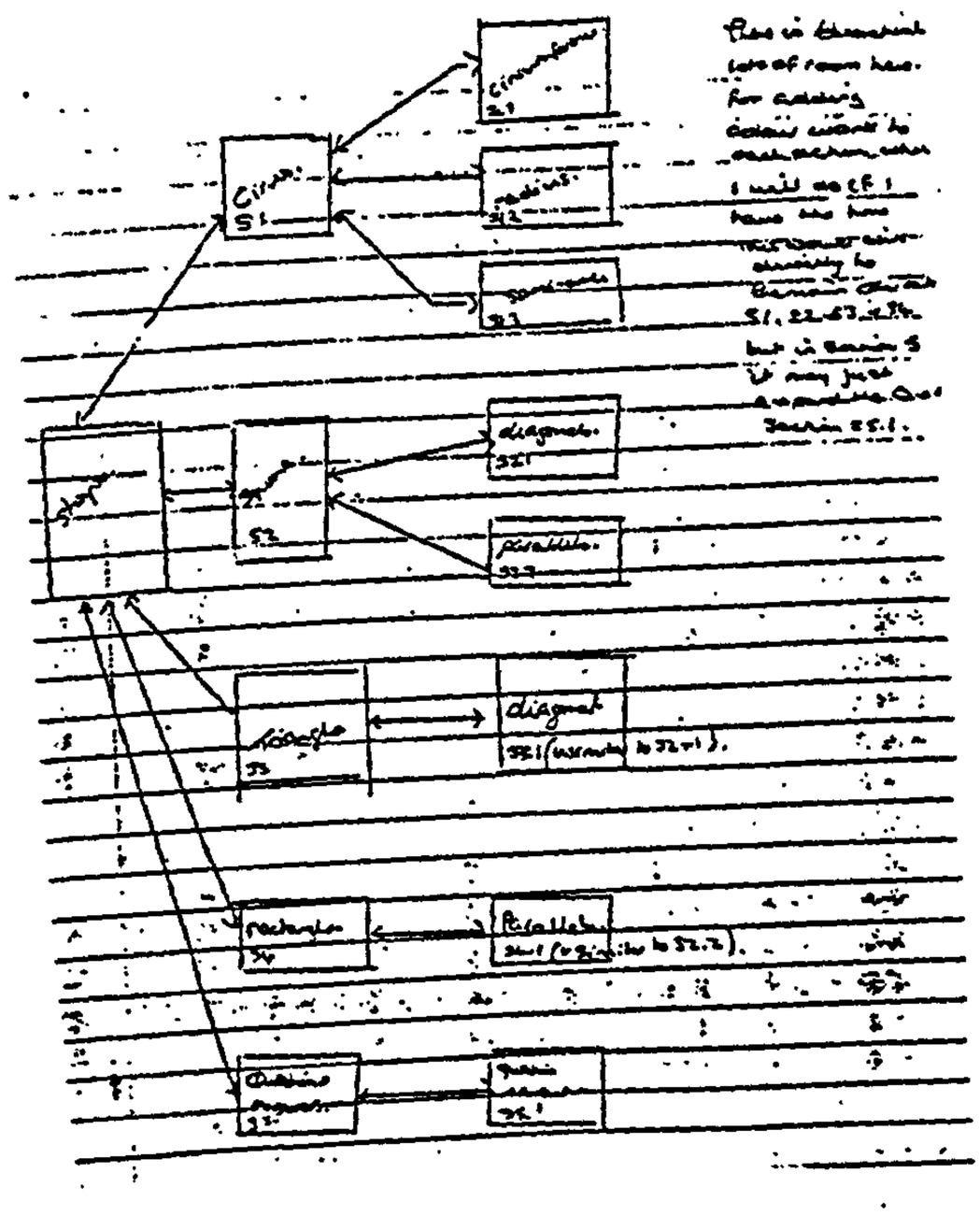
Comments ([illegible] on pages [illegible] for further comments to be made)

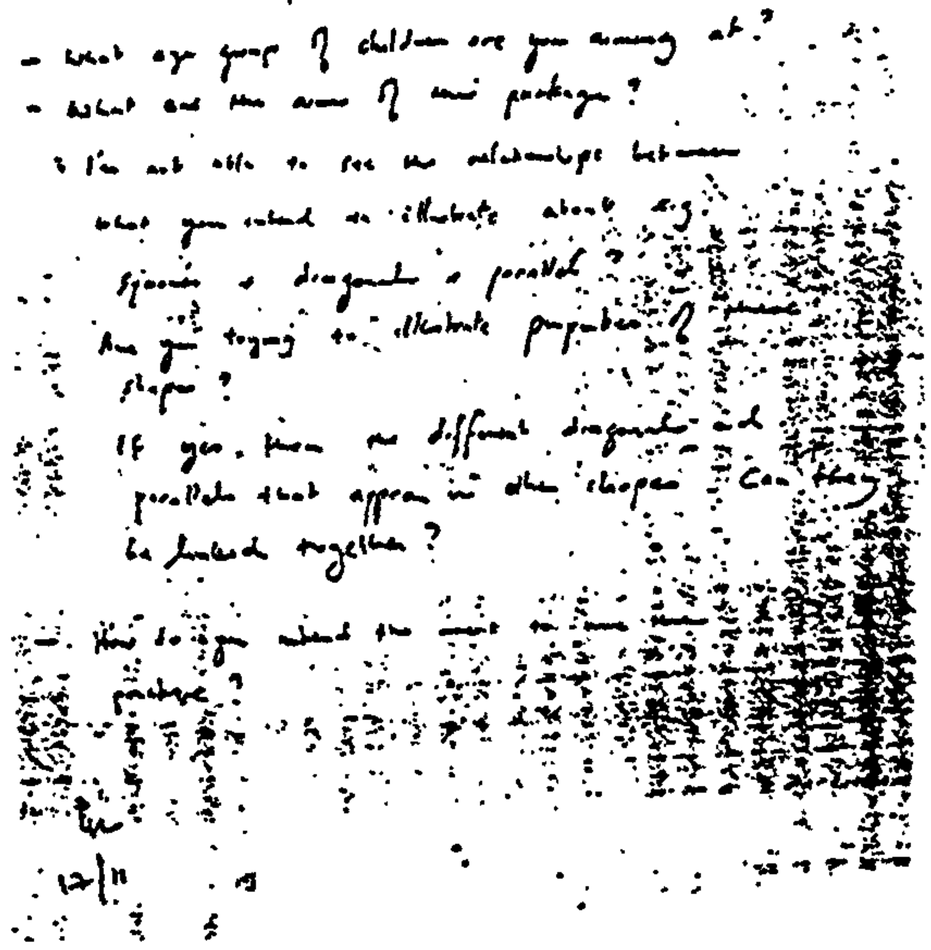
I think I understand what you are trying as structure, it looks fine.
In your try to improve the design, I would like to draw your attention to:

- Think about who your users are? which age group? Experience? Knowledge?
- Is it possible for users to move from page 5 to page 8, for example?
- Objectives of the package.
- How would you like the package to be used?

Student 3







Comments

You have given much thought to this assignment. well done. I agree with the comments you made in the questions and some points I would like to highlight:

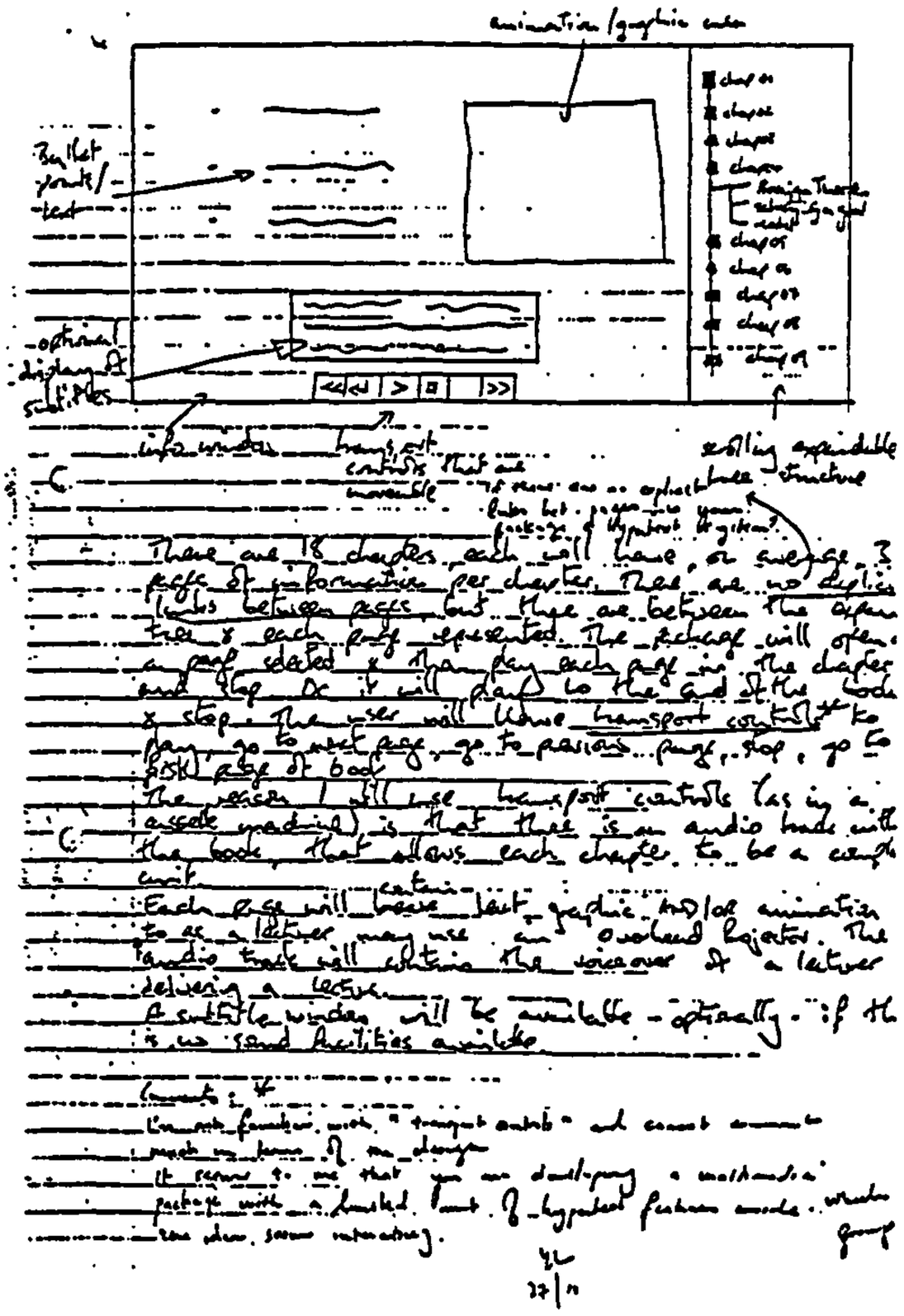
- what age group of children are you aiming at?
- what are the aims of your package?

I'm not able to see the relationships between what you intend to illustrate about eg. spheres & diagonals & parallels?

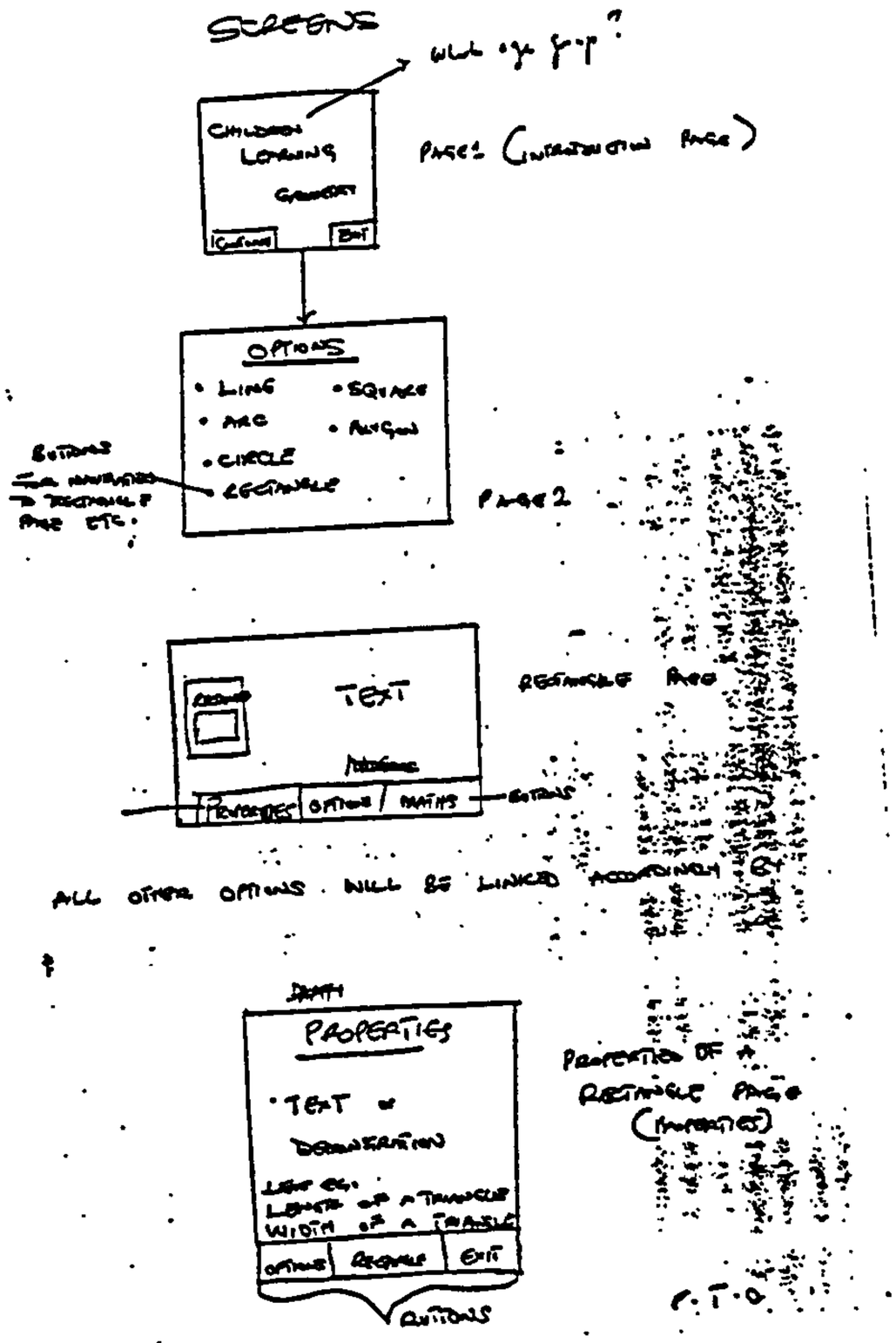
Are you trying to illustrate properties of shapes?

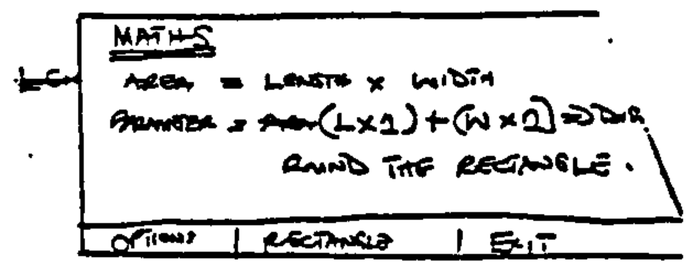
If you have the different diagonals and parallels that appear in other shapes can they be linked together?

Student 4


animation/graphic area
Bullet point/ text
optional display of subtitles
<< < > ■ >>
info window
transport controls that are invisible
scrolling expandable structure
There are 18 chapters each will have, on average, 3 pages of information per chapter. There are no duplicate links between pages but there are between the open tree & each page presented. The package will open a page related to the page each page in the chapter and stop & it will pause to the end of the book & stop. The user will have transport controls to play, go to next page, go to previous page, stop, go to first page of book
The reason I will use transport controls (as in a cassette machine) is that there is an audio track with the book that allows each chapter to be a complete unit
Each page will contain text, graphic and/or animation to as a lecturer may use an overhead projector. The audio track will contain the voice over of a lecturer delivering a lecture.
A subtitle window will be available - optionally - if the sound facilities available
It seems to me that you are developing a multimedia package with a limited set of hypertext features which may be slow when interacting.

Student 5



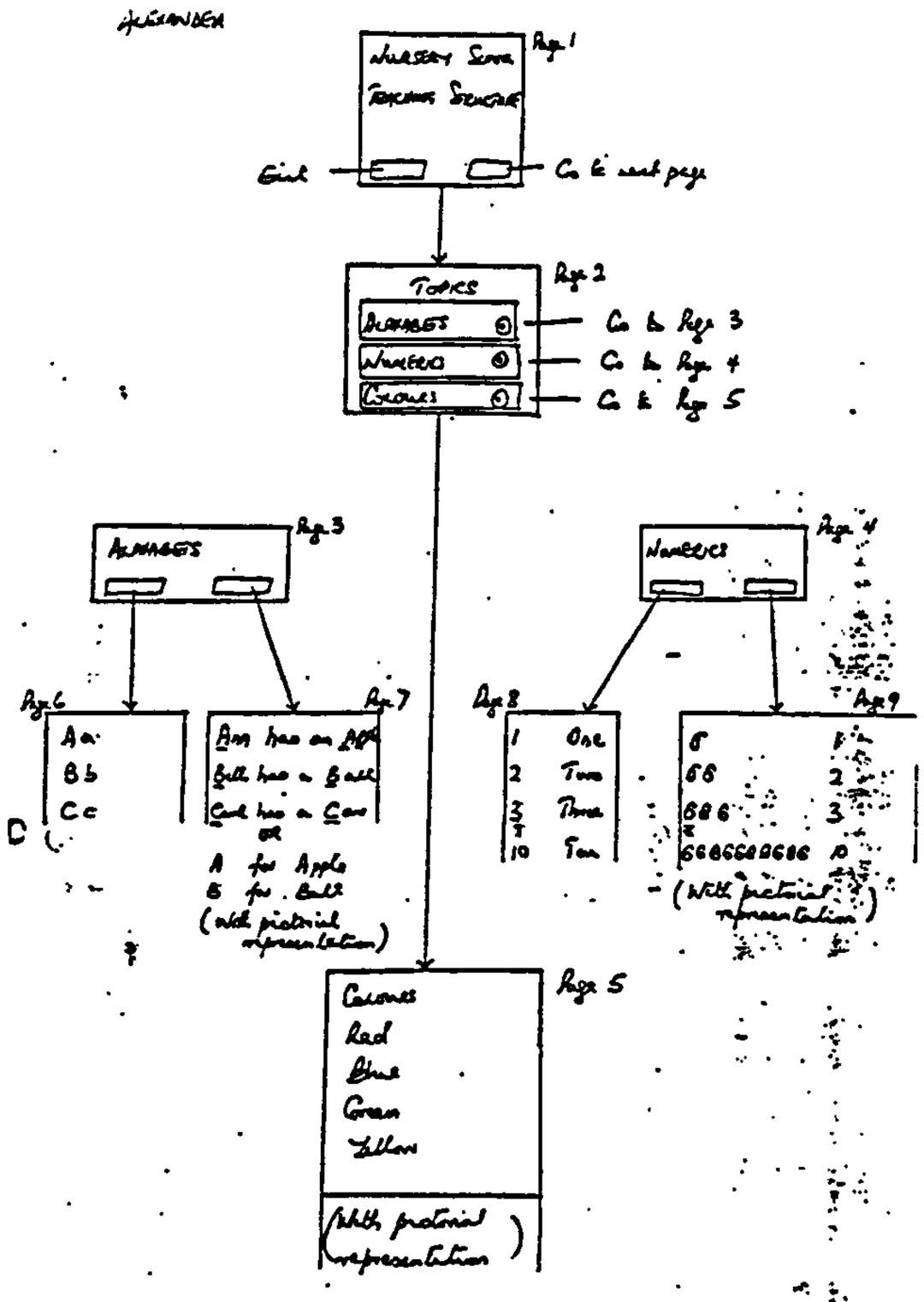


Comments

1. Structure : I'm not clear what the structure you have drawn - Are properties and mathematics common to all the different topics?
If they are not, should I expect to have an extra layer, then they should not be drawn into the same box as using a single box. However, you have identified something which is good because you realise that there are overlaps and your package should be able to link a user interested in the property of length to property of area without going back to the top of the structure again.

2. There is good potential in making it well-structured

Student 6



Pages 6 & 9.

It is likely these pages will be extended to five more pages because of the pictorial representation.

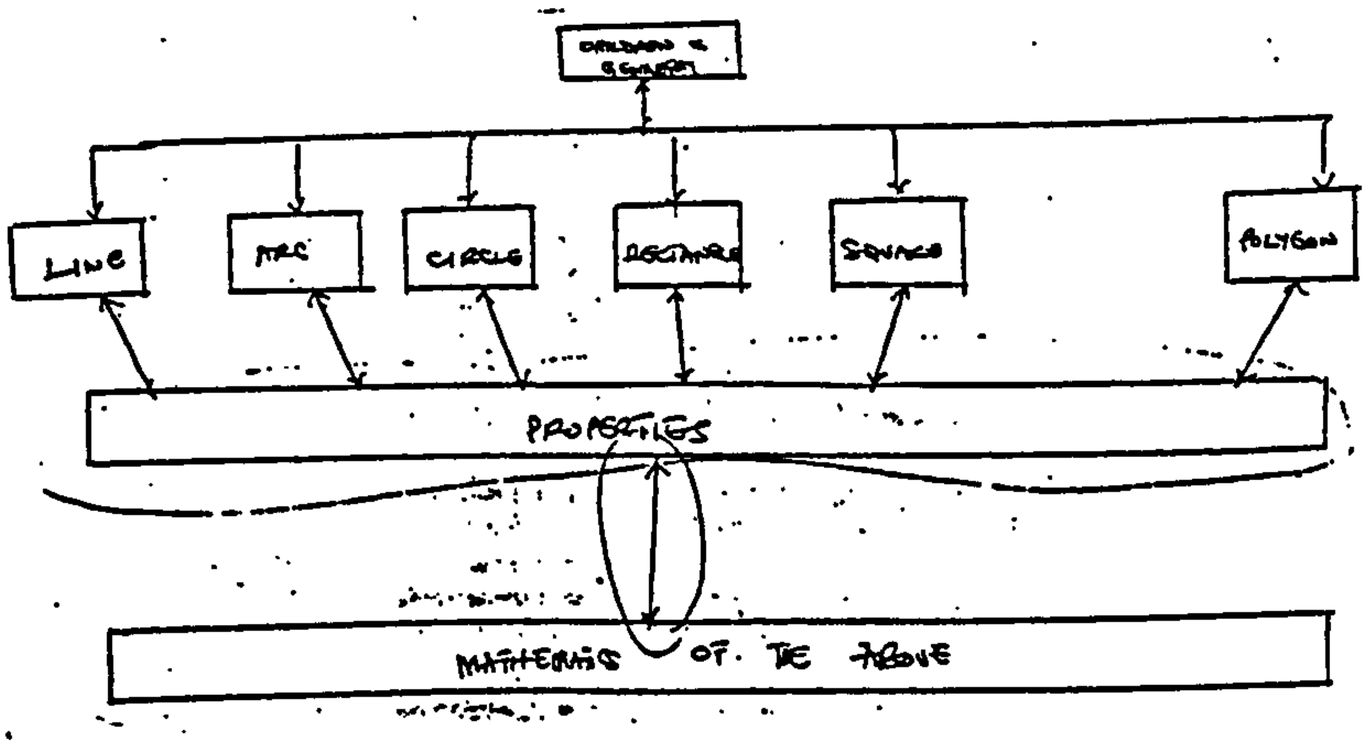
Page 6 - Explanation of the Capital & Small letters
Page 7 - Pictorial representation of apple, ball, car and combining the two parts
e.g. Aaa has an APPLE
Page 8 - Spelling and writing of 1-10
Page 9 - Pictorial representation

There is a possibility of introducing a story telling page

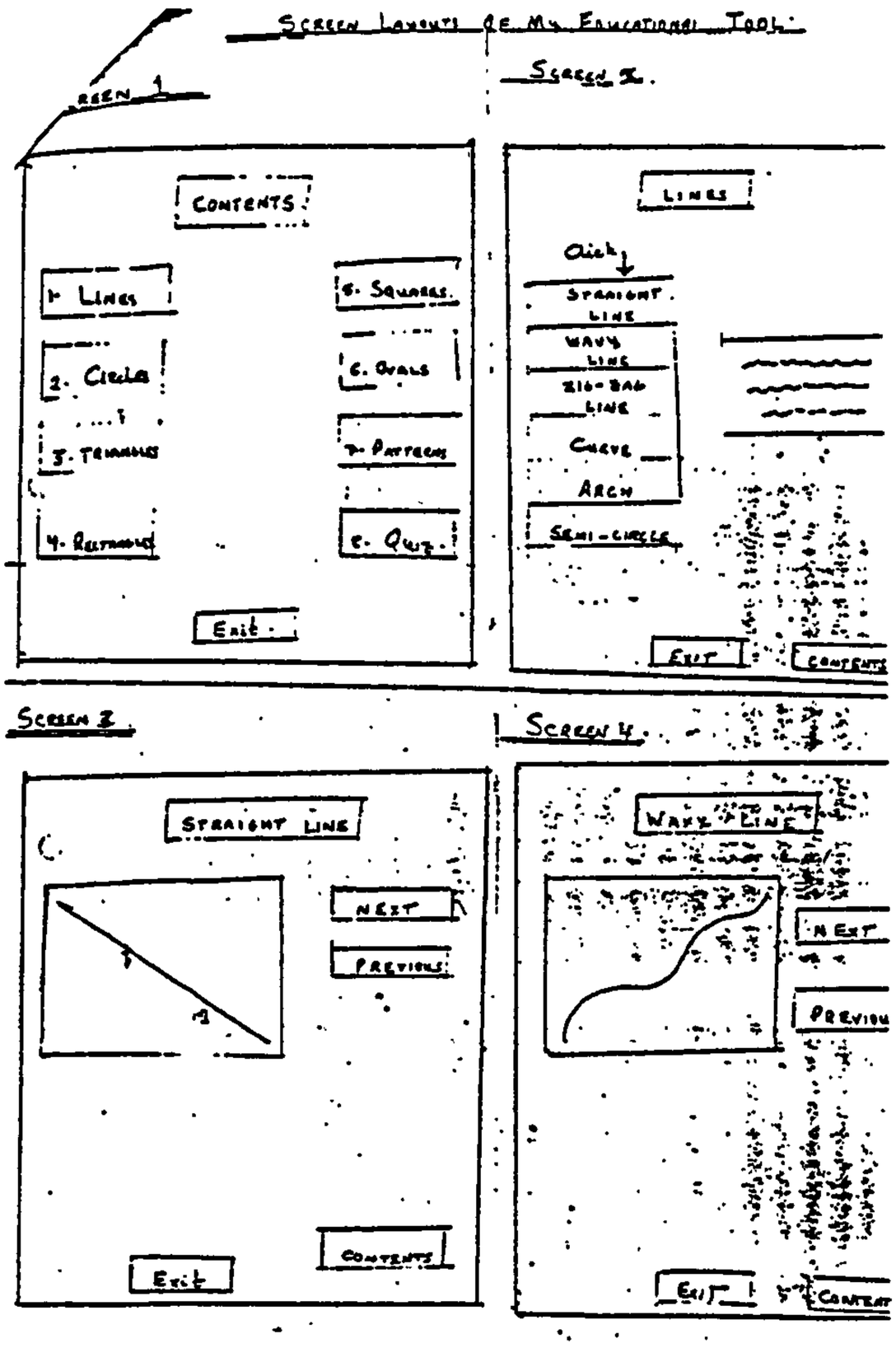
Comments

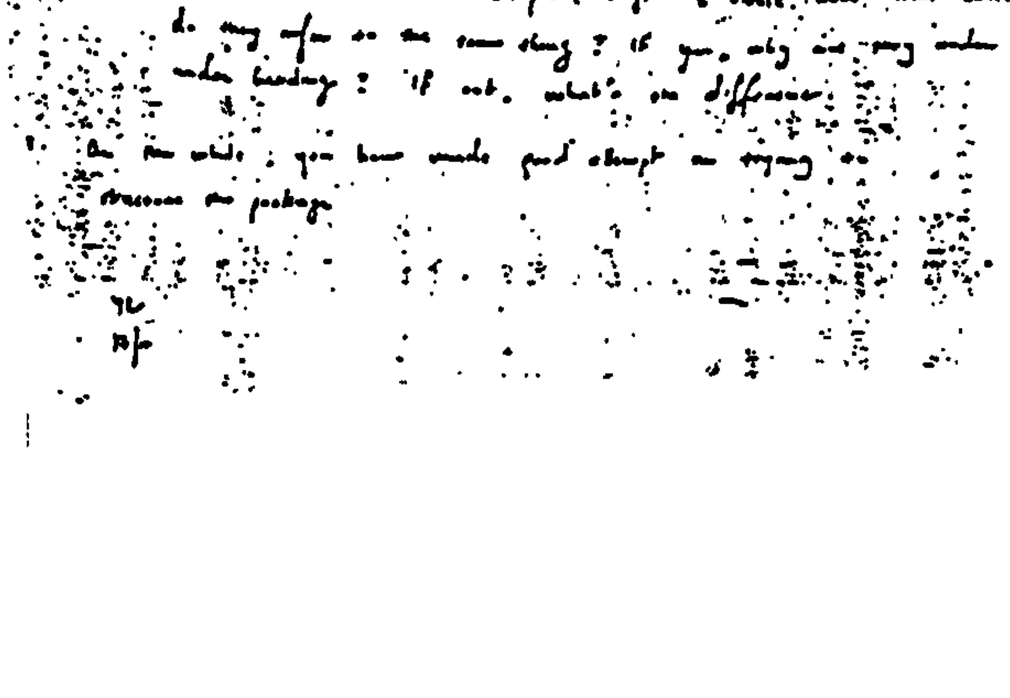
- You have done well by identifying who your users are and listed many good ideas you want to incorporate into your package. You may want to identify what the objectives of your package are? What are you wanting these children to learn? Games to be quite a few things - reading, writing, talking? You may want to narrow down the scope of what you can do a more complete design of your package?
- Are you also incorporating audio into your package?
- Structure: I'm not quite sure I understand how page 3 is linked to pages 6 & 7.

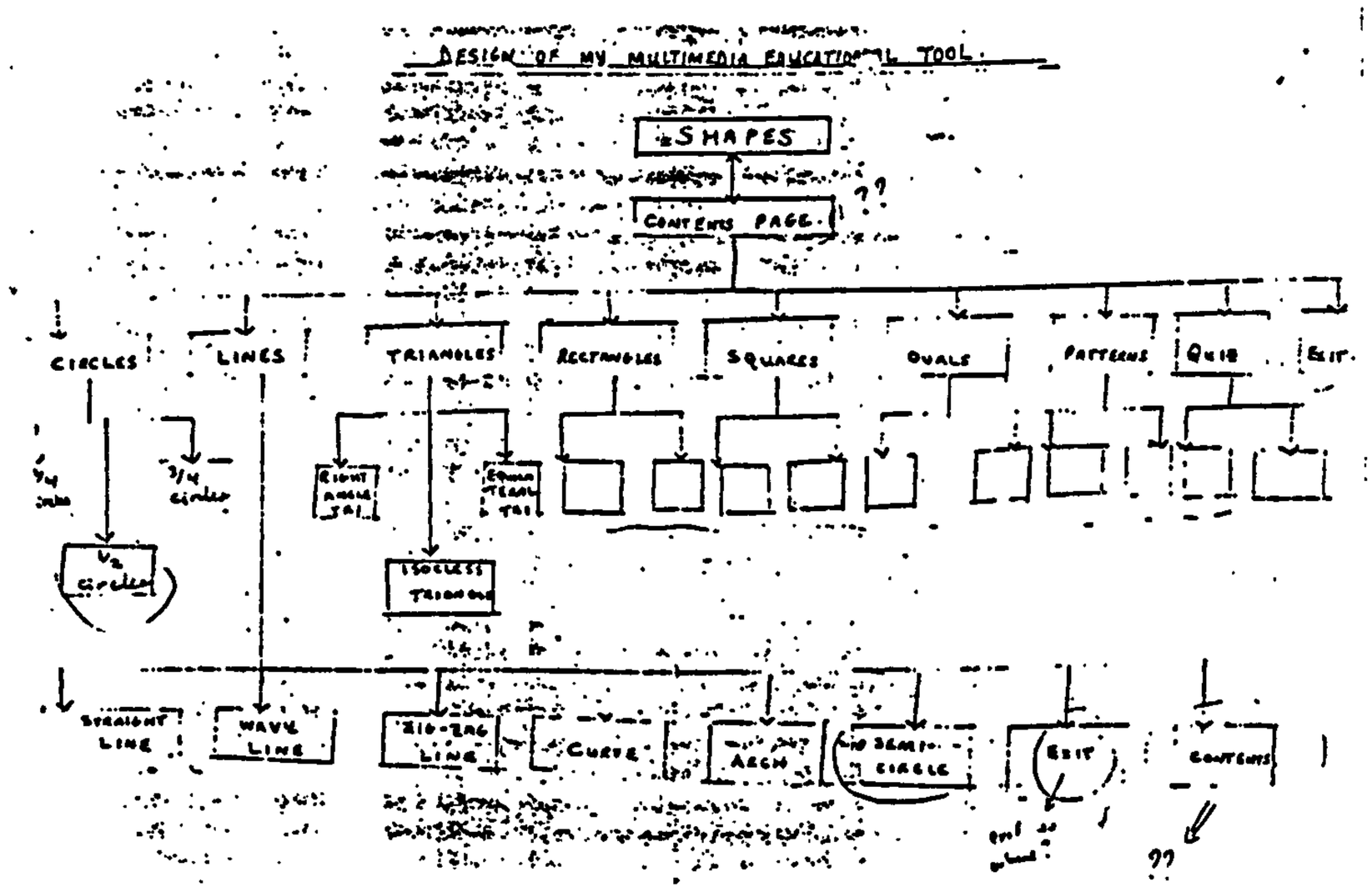
4
17/4



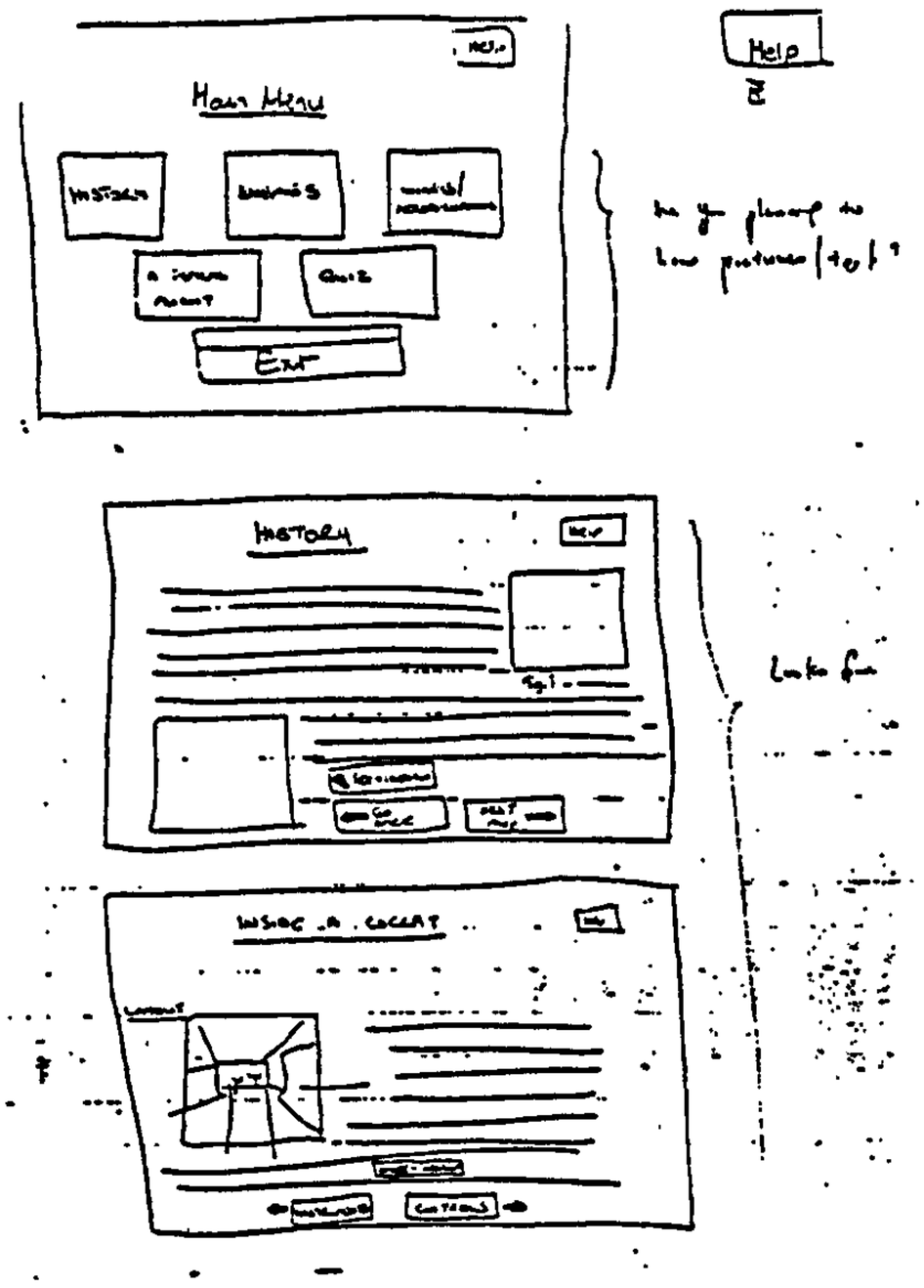
Student 7


SCREEN LAYOUTS OF MY EDUCATIONAL TOOL
SCREEN 1
SCREEN 2
SCREEN 3
SCREEN 4
CONTENTS
1. LINES
2. CIRCLES
3. TRIANGLES
4. RECTANGLES
5. SQUARES
6. OVALS
7. PATTERNS
8. QUIZ
EXIT
LINES
STRAIGHT LINE
WAVY LINE
ZIG-ZAG LINE
CURVE
ARCH
SEMI-CIRCLE
NEXT
PREVIOUS
CONTENTS



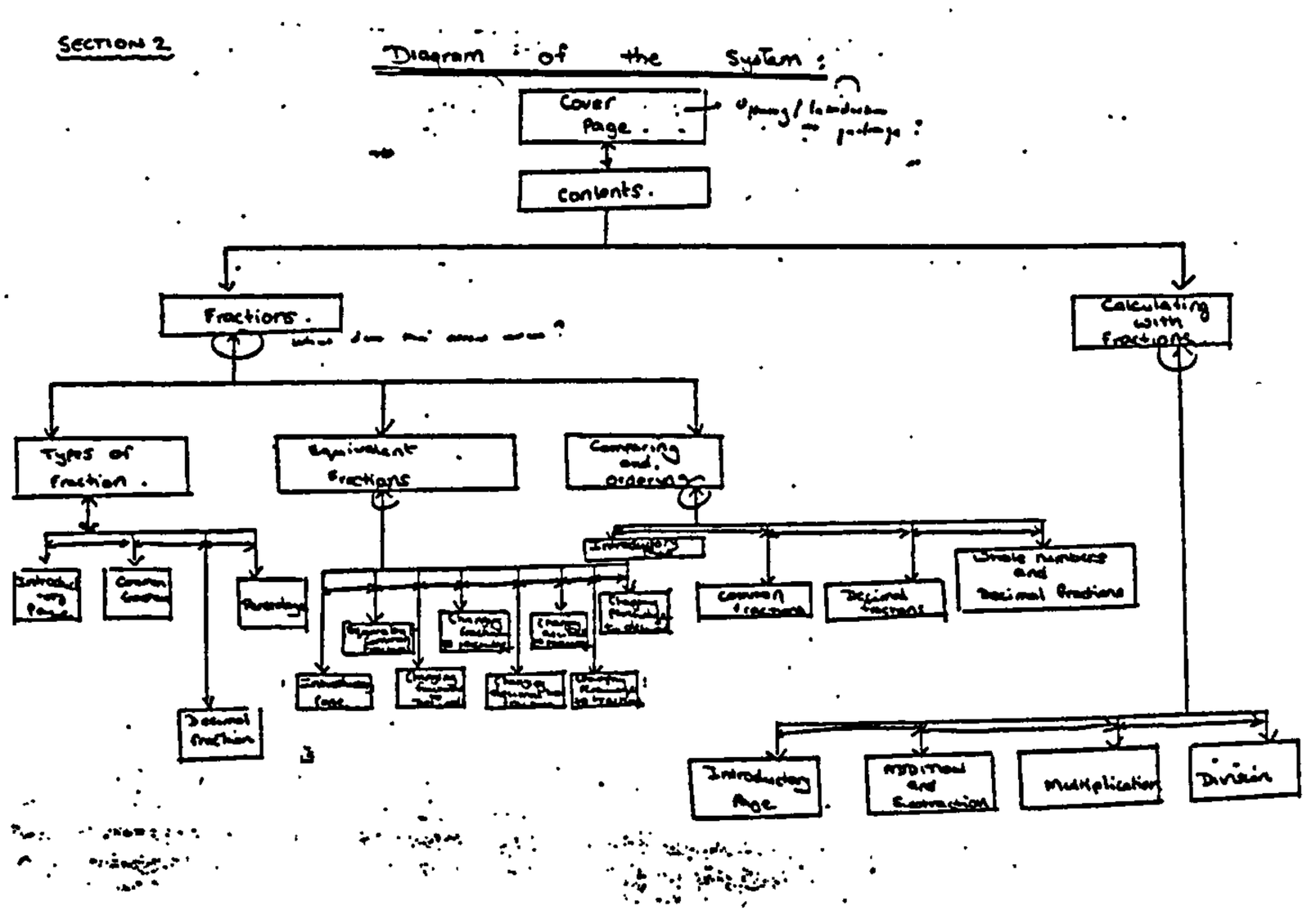

DESIGN OF MY MULTIMEDIA EDUCATIONAL TOOL
SHAPES
CONTENTS PAGE
CIRCLES
LINES
TRIANGLES
RECTANGLES
SQUARES
OVALS
PATTERNS
QUIZ
EXIT
STRAIGHT LINE
WAVE LINE
ZIG-ZAG LINE
CURVE
ARCH
SEMI-CIRCLE
CONTENTS

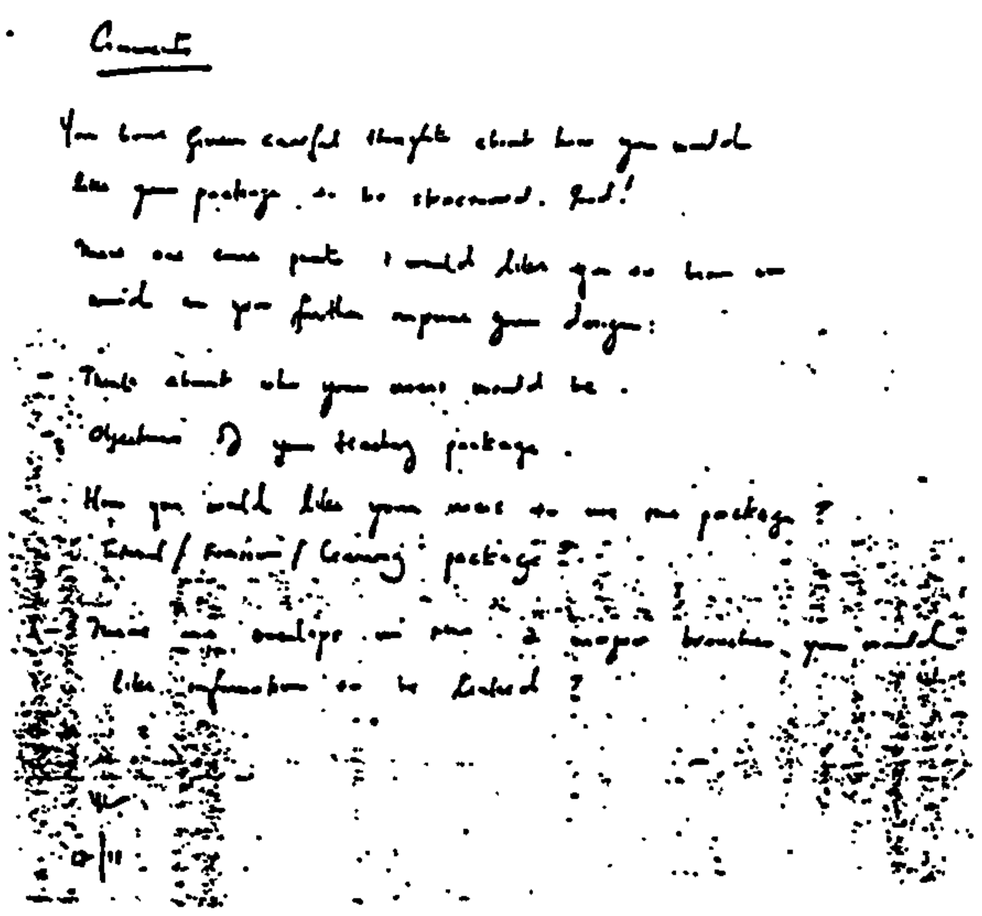
Student 8


Help
Main Menu
Exit
HISTORY

INDEX/STRUCTURE AIRCRAFT + FLIGHT
A TYPICAL FLIGHT
Quiz (HUMAN)
HELP SCREENS
Com3120
EDUCATIONAL MULTIMEDIA
DUTCH PRODUCTION

Student 9





Comments

You have given careful thought about how you would like your package to be structured. Good!

There are some points I would like you to bear in mind as you further improve your design:

- Think about who your users would be.
- Objectives of your teaching package.
- How you would like your users to use your package?
  Linear / random / learning package?
- There are overlaps in some of the chapter headings you would like information to be linked?

Appendix F

Sample programs

Appendix F1

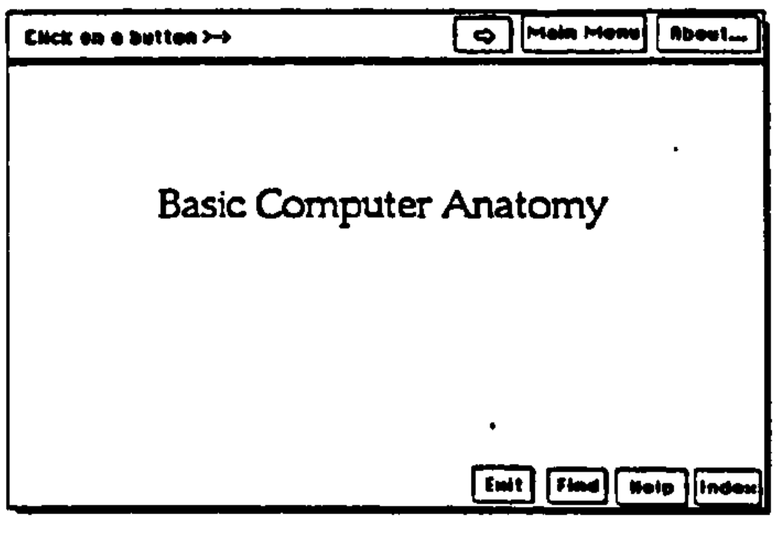
Appendix F1 shows all the cards in hypertext prototype1 and the corresponding HyperCard stack script.

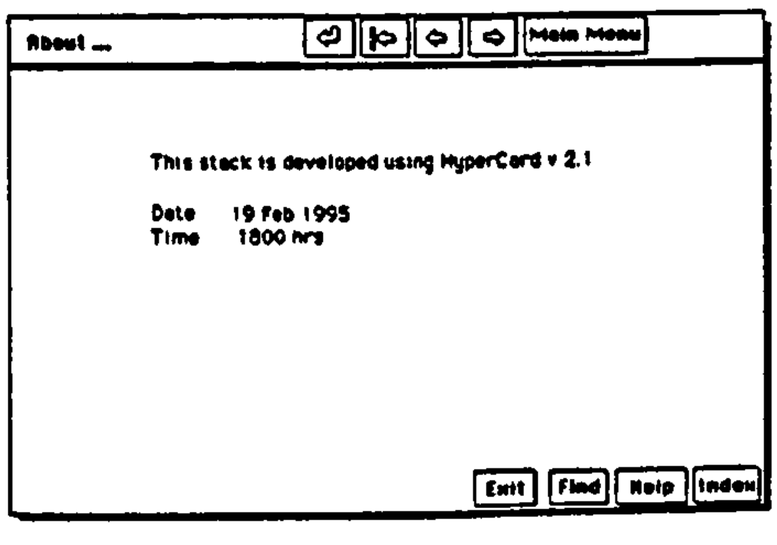
Appendix F2

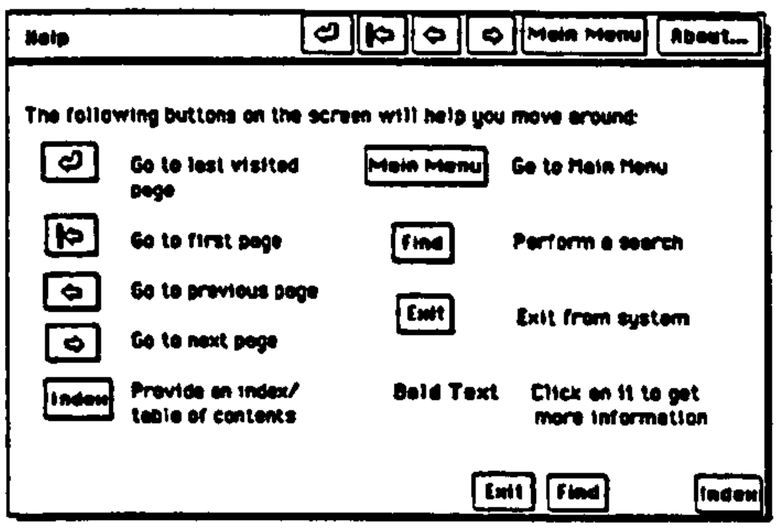
The entire code of HyperAT is not provided in this thesis since the emphasis of the thesis is not on the code but on the ideas and the approach it represents. Though Macintosh Common Lisp is a powerful and flexible language, it may not be the best choice for everyone interested to pursue the work, since many factors (for example, cost, technical support, *etc.*) could affect the decision. Therefore, only a sample Lisp program for the automatic conversion to HTML format is provided here to illustrate the implementation of one of the important features provided in HyperAT.

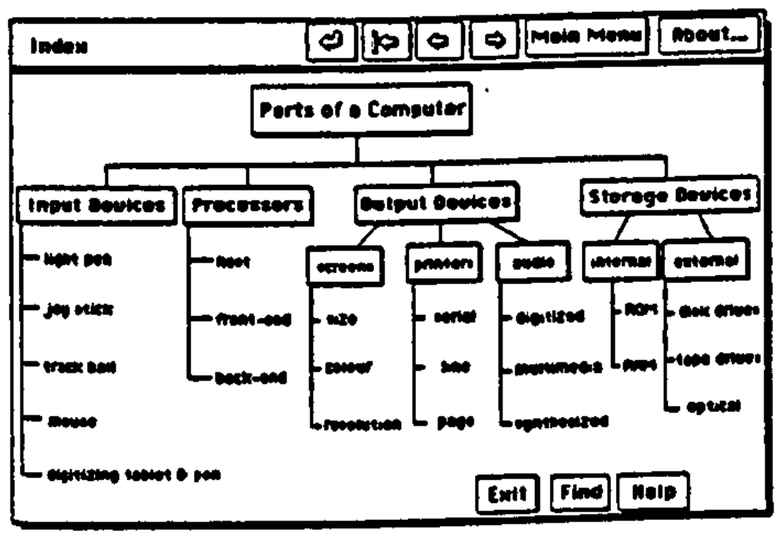
Appendix F1

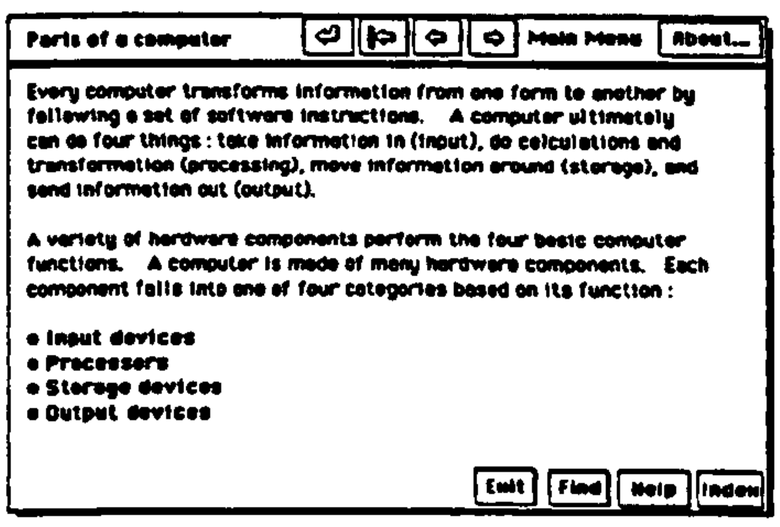
A sample hypertext prototype1 built using HyperCard (version 2.01)

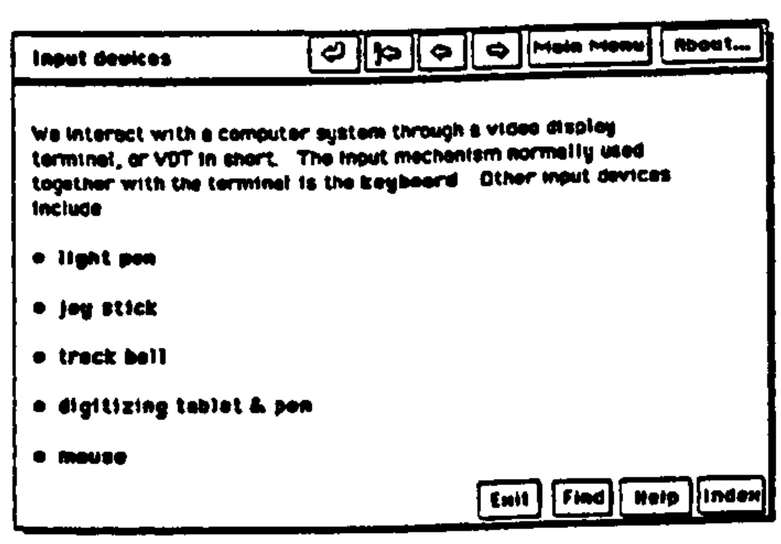


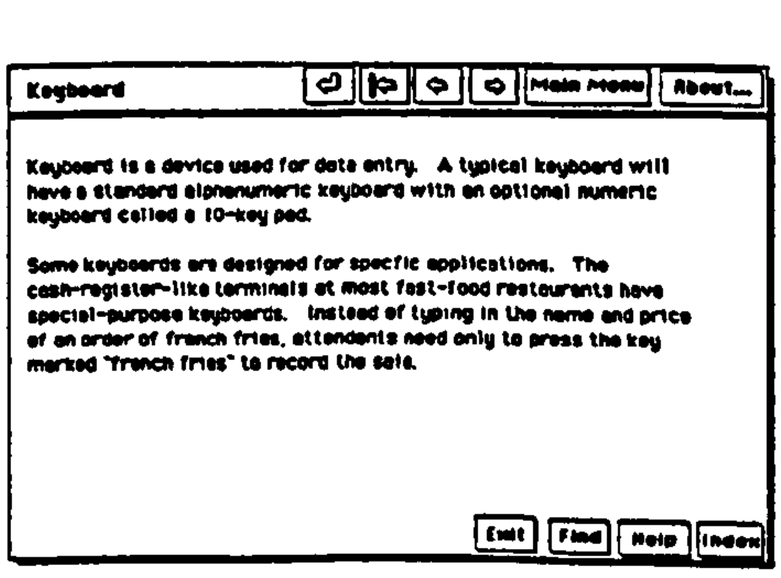


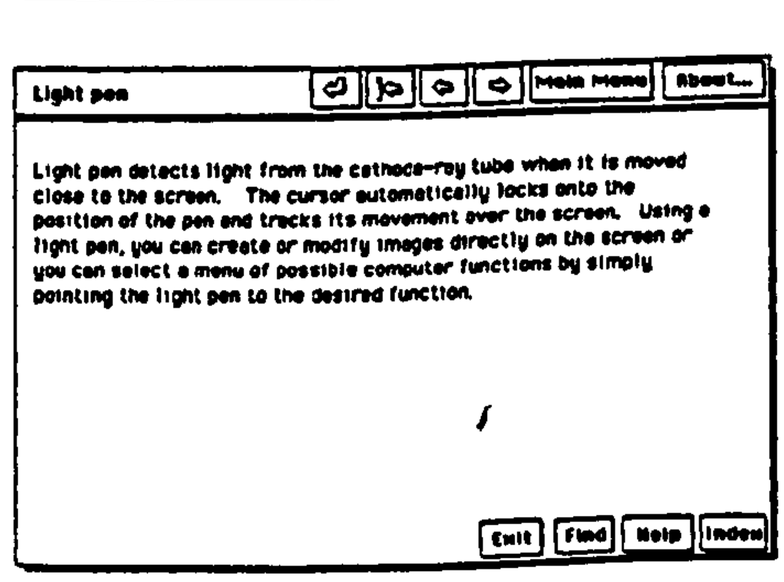


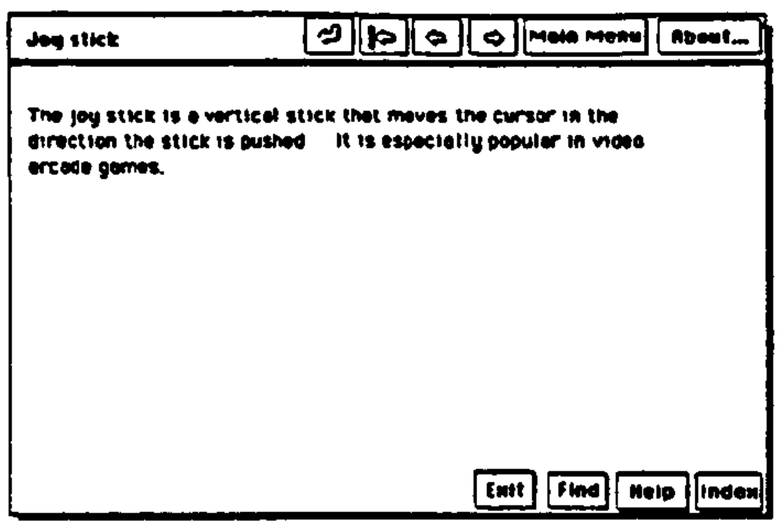


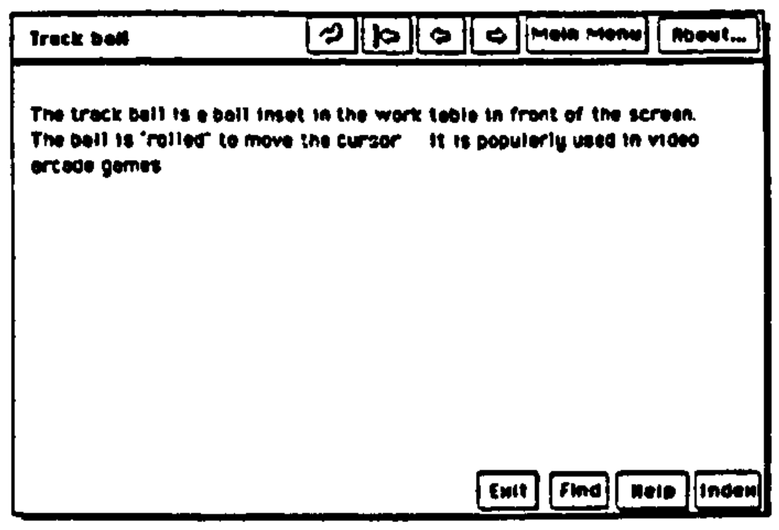


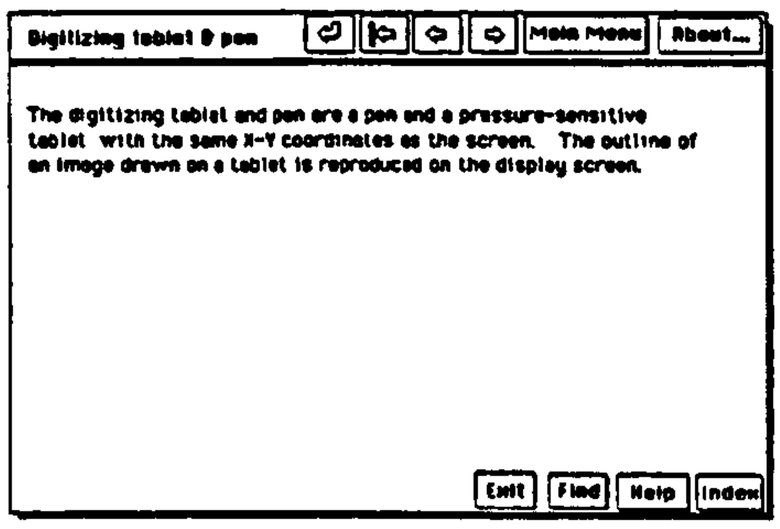


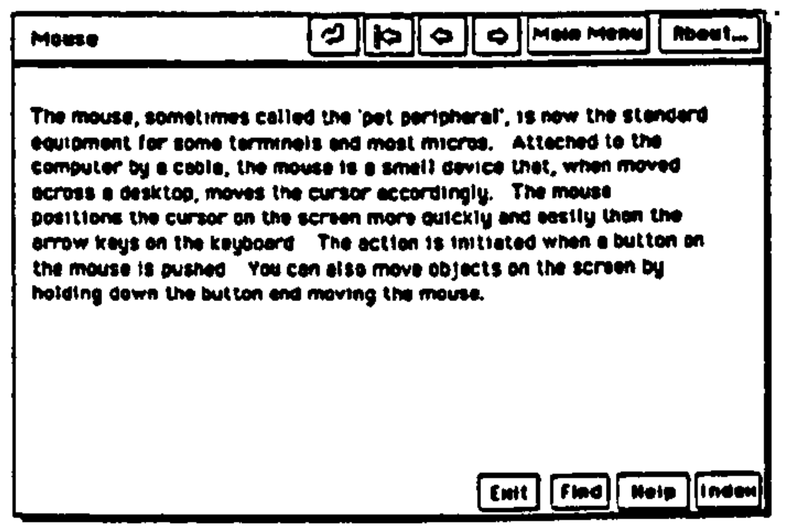


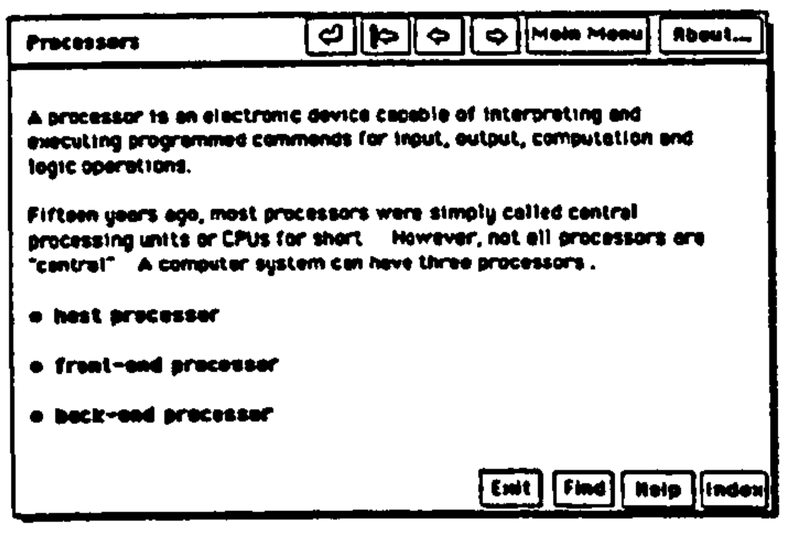


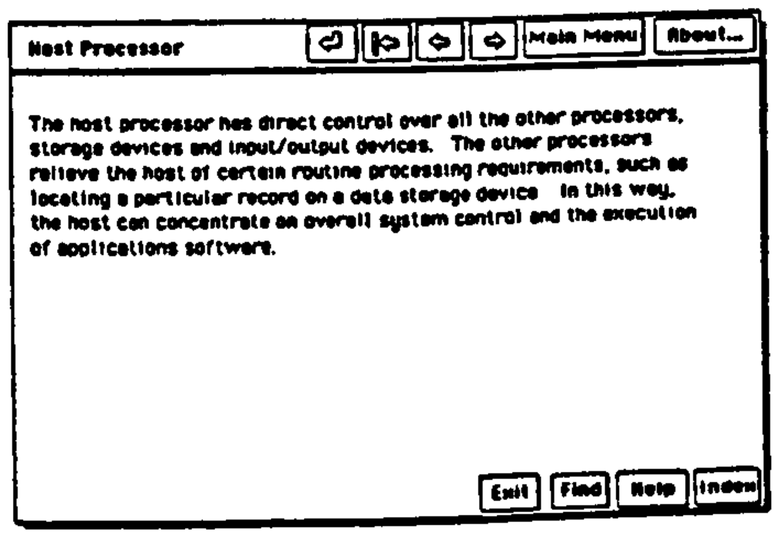

Joy stick
Main Menu
About...
The joy stick is a vertical stick that moves the cursor in the direction the stick is pushed It is especially popular in video arcade games.
Exit
Find
Help
Index

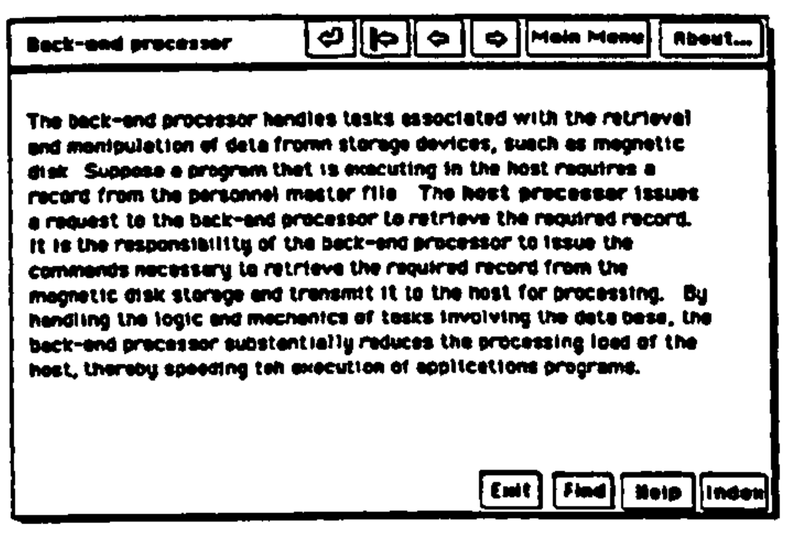

Track ball
Main Menu
About...
The track ball is a ball inset in the work table in front of the screen. The ball is "rolled" to move the cursor It is popularly used in video arcade games
Exit
Find
Help
Index

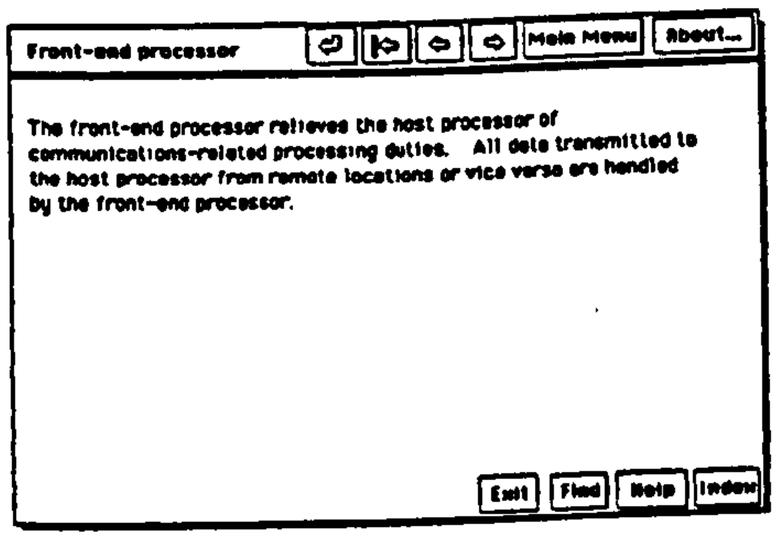

Digitizing tablet & pen
Main Menu
About...
The digitizing tablet and pen are a pen and a pressure-sensitive tablet with the same X-Y coordinates as the screen. The outline of an image drawn on a tablet is reproduced on the display screen.
Exit
Find
Help
Index

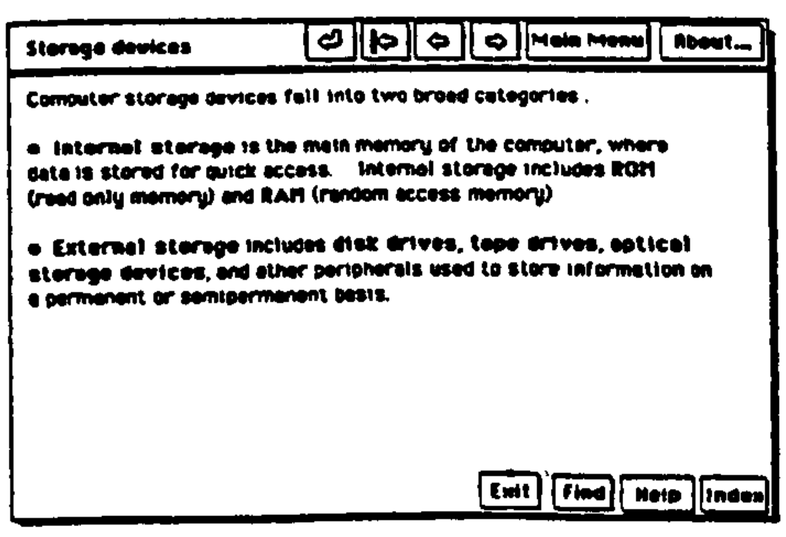

Mouse
Main Menu
About...
The mouse, sometimes called the 'pet peripheral', is now the standard equipment for some terminals and most micros. Attached to the computer by a cable, the mouse is a small device that, when moved across a desktop, moves the cursor accordingly. The mouse positions the cursor on the screen more quickly and easily than the arrow keys on the keyboard The action is initiated when a button on the mouse is pushed You can also move objects on the screen by holding down the button and moving the mouse.
Exit
Find
Help
Index

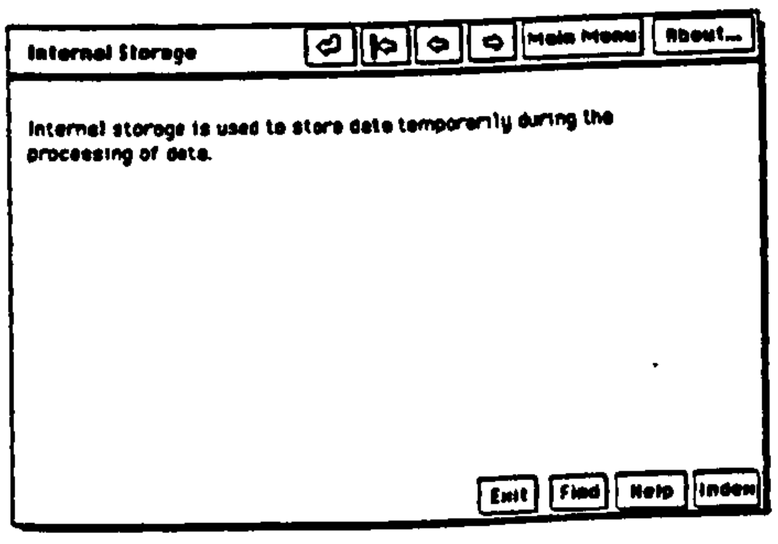

Processors
Main Menu
About...
A processor is an electronic device capable of interpreting and executing programmed commands for input, output, computation and logic operations.
Fifteen years ago, most processors were simply called central processing units or CPUs for short However, not all processors are "central" A computer system can have three processors .
• host processor
• front-end processor
• back-end processor
Exit
Find
Help
Index


Host Processor
Main Menu
About...
The host processor has direct control over all the other processors, storage devices and input/output devices. The other processors relieve the host of certain routine processing requirements, such as locating a particular record on a data storage device In this way, the host can concentrate on overall system control and the execution of applications software.
Exit
Find
Help
Index


Back-end processor
Main Menu
About...
The back-end processor handles tasks associated with the retrieval and manipulation of data fromm storage devices, such as magnetic disk Suppose a program that is executing in the host reauires a record from the personnel master file The host processor issues a request to the back-end processor to retrieve the required record. It is the responsibility of the back-end processor to issue the commands necessary to retrieve the required record from the magnetic disk storage and transmit it to the host for processing. By handling the logic and mechanics of tasks involving the data base, the back-end processor substantially reduces the processing load of the host, thereby speeding teh execution of applications programs.
Exit
Find
Help
Index


Front-end processor
Main Menu
About...
The front-end processor relieves the host processor of communications-related processing duties. All data transmitted to the host processor from remote locations or vice versa are handled by the front-end processor.
Exit
Find
Help
Index


Storage devices
Main Menu
About...
Computer storage devices fall into two broad categories .
• Internal storage is the main memory of the computer, where data is stored for quick access. Internal storage includes ROM (read only memory) and RAM (random access memory)
• External storage includes disk drives, tape drives, optical storage devices, and other peripherals used to store information on a permanent or semipermanent basis.
Exit
Find
Help
Index


Internal Storage
Main Menu
About...
Internal storage is used to store data temporarily during the processing of data.
Exit
Find
Help
Index

ROM

Main Menu | About...

ROM or Read Only Memory means that data can only be read but not written to.

Exit | Find | Help | Index

RAM

Main Menu | About...

RAM or Random Access Memory is the memory area in which all programs and data must reside before programs can be executed or data manipulated

Exit | Find | Help | Index

External Storage

Main Menu | About...

External storage refers to the storage of data outside the internal storage area of the computer system.

Exit | Find | Help | Index

Disk drives

Main Menu | About...

Disk drives refer to magnetic storage devices that record data on flat rotating disks.

Exit | Find | Help | Index

Tape drives

Main Menu | About...

Tape drives refer to the hardware devices that contain the read/write mechanism for the magnetic tape storage medium.

Exit | Find | Help | Index

Optical Storage Devices

Main Menu | About...

Optical storage devices refer to a read-only secondary storage medium that uses laser technology.

Exit | Find | Help | Index

Output Devices

Main Menu | About...

No amount of processing is of any value unless we have a way of getting the results out of the computer. A typical computer has three main forms of output.

- Monitor screens
- Printers
- Audio output

Exit | Find | Help | Index

Monitor screens

Main Menu | About...

A monitor screen is a televisionlike display for soft-copy output in a computer system. Alphanumeric and graphic output are displayed on the terminal's monitor. The three primary attributes of monitors are:

- size
- colour
- resolution

Exit | Find | Help | Index

Size

Main Menu | About...

The size of the screen varies. The common screen sizes are:

- 25 lines x 80 characters
- 32 lines x 80 characters
- 25 lines x 132 characters

The diagonal dimension of the display screen varies from 5 to 25 inches.

Exit | Find | Help | Index

Colour

Main Menu | About...

Monitors are either monochrome or colour. Monochrome monitors display images in a single colour, usually white, green or amber

Colour monitors add the dimension of the other colours, which draw attention to various aspects of the output. Take for example, colours can be used to highlight direction and type of fluid flow, size of pipelines for an oil refinery and so on.

Colour monitors can be used to present alphanumeric information more efficiently and effectively. For example, red can be used to highlight those items needing immediate attention.

Exit | Find | Help | Index

Resolution Main Menu About...

Resolution refers to the number of addressable points on the screen, that is, the number of points to which light can be directed under program control In short, resolution refers to the quality of output

A strictly alphanumeric terminal has about 65,000 such points whereas a terminal used primarily for computer graphics and computer-aided design may have over 16 million points. High-resolution monitors project extremely clear images that almost look like photographs

Exit Find Help Index

Printers Main Menu About...

Printers are used to prepare hard copy output and they are usually classified as

- serial printers
- line printers
- page printers

Printers are rated by their print speed Printer speeds for serial printers are measured in characters pers second (cps) whereas line and page printers are measured in lines per minute (lpm).

Printers are further categorised as impact or nonimpact.

Exit Find Help Index

Serial Printers Main Menu About...

Most serial printers are used with microcomputers They support terminal clusters and give users the flexibility to obtain hard copy output

The different types of serial printers are :

- dot-matrix
- daisy-wheel
- ink-jet
- thermal

Exit Find Help Index

Dot-matrix printer Main Menu About...

The dot-matrix printer arranges printed dots to form characters and all kinds of images. One or several vertical columns of small print hammers are contained in a rectangular "print head" The hammers are activated independently to form a dot character image as the print head moves horizontally across the paper

The quality of the printed output is directly proportional to the density of the dots in the matrix. High quality dot-matrix printers form characters that appear solid, and they can be used for business letters as well as for routine data processing output.

Exit Find Help Index

Daisy-wheel printer Main Menu About...

The daisy-wheel printer produces letter-quality (LQ) output for word processing applications. An interchangeable daisy wheel containing a set of fully formed characters is spun to the desired character. A print hammer strikes the embossed character on the print wheel to form the image. The print quality is at least equal to the best electric typewriters.

Although the daisy-wheel printers have the highest quality text output of serial printers, they are the slowest and cannot produce graphic output.

Exit Find Help Index

Ink-jet printer Main Menu About...

Ink-jet printers squirt dots of ink on paper to form images in much the same way dot-matrix printers do. The advantage ink-jet printers have over dot matrix printers is that they can produce multicolour output.

Exit Find Help Index

Thermal printer Main Menu About...

Heat elements produce dot-matrix images on heat-sensitive paper. The main disadvantage is the cost of the heat-sensitive paper. however, advantages include compactness, limited noise, and low purchase price. Some thermal printers are capable of colour output

Exit Find Help Index

Line Printers Main Menu About...

Line printers are impact printers that print one line at a time The most popular types of line printers are

- band printer
- chain printer
- matrix line printer

Exit Find Help Index

Band printer Main Menu About...

Band printer has a print hammer for each character position in the line of print (usually 132). several similar sets of fully formed characters are embossed on each link of the print chain. The paper is momentarily stopped and as the desired character passes over a given column, the hammer is activated, pressing the ribbon against the paper to form the image

Band printers are capable of printing on continuous-feed paper, cards and documents of varying sizes. Interchangeable bands and chains make it easy for operators to change the style of print.

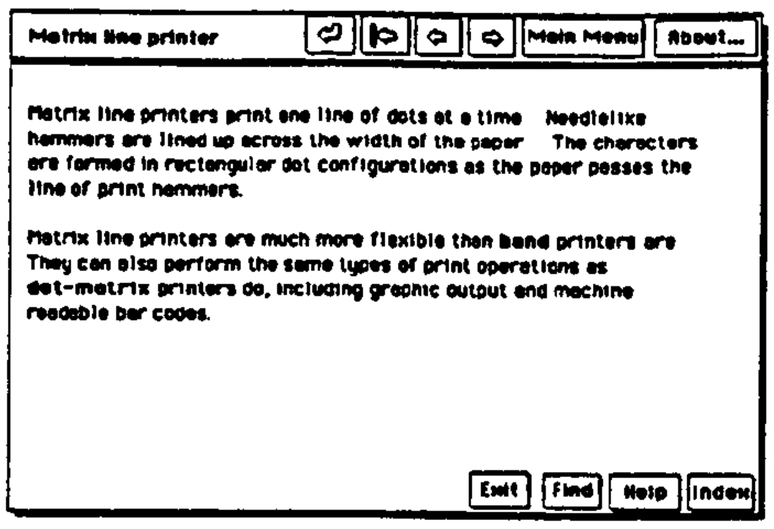
Exit Find Help Index

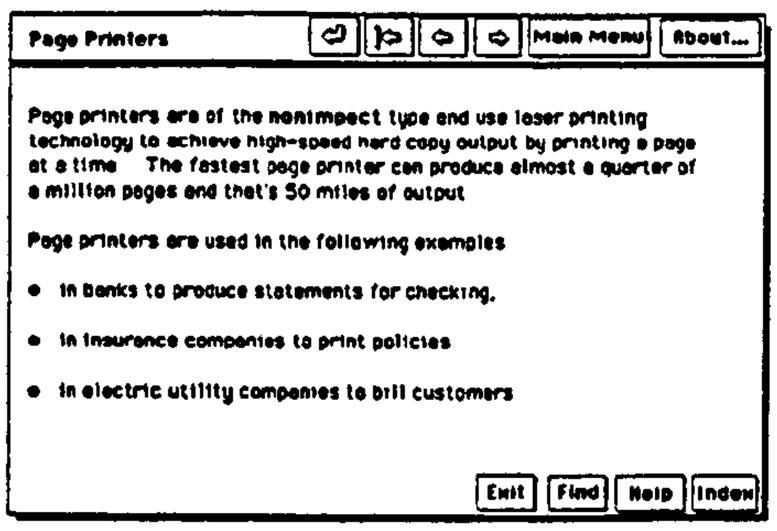
Chain printer Main Menu About...

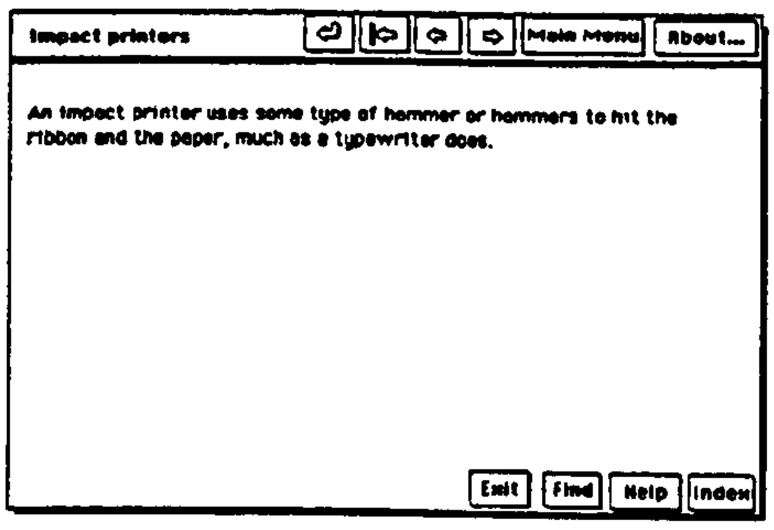
Chain printers have a print hammer for each character position in the line of print (usually 132). On a chain printer, the characters are embossed on each link of the print chain. Paper is momentarily stopped and as the desired character passes over a given column, the hammer is activated, pressing the ribbon against the paper to form the image

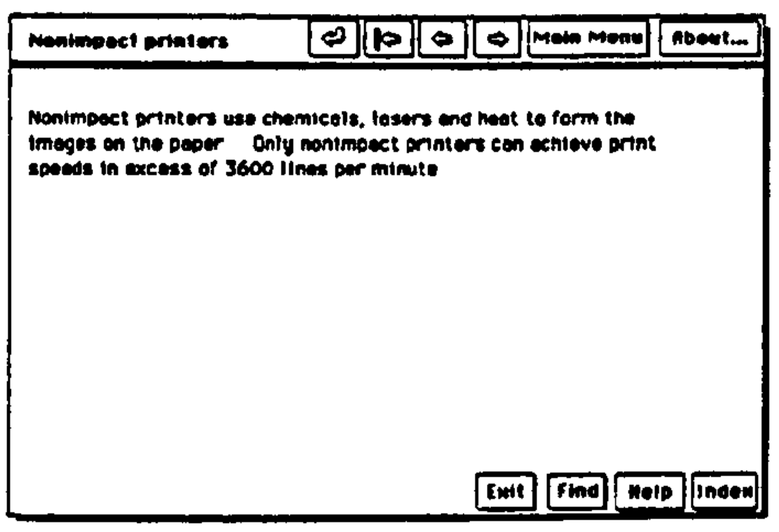
Chain printers are capable of printing on continuous-feed paper, cards, documents of varying sizes. Interchangeable chains make it easy for operators to change the style of print.

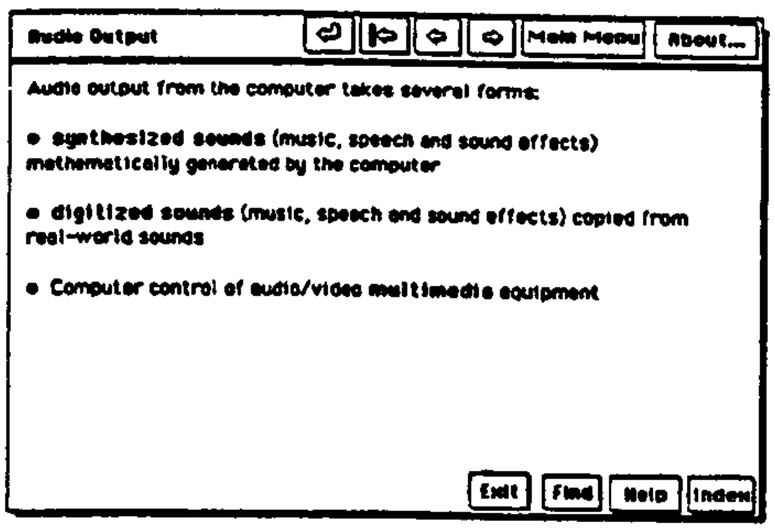
Exit Find Help Index

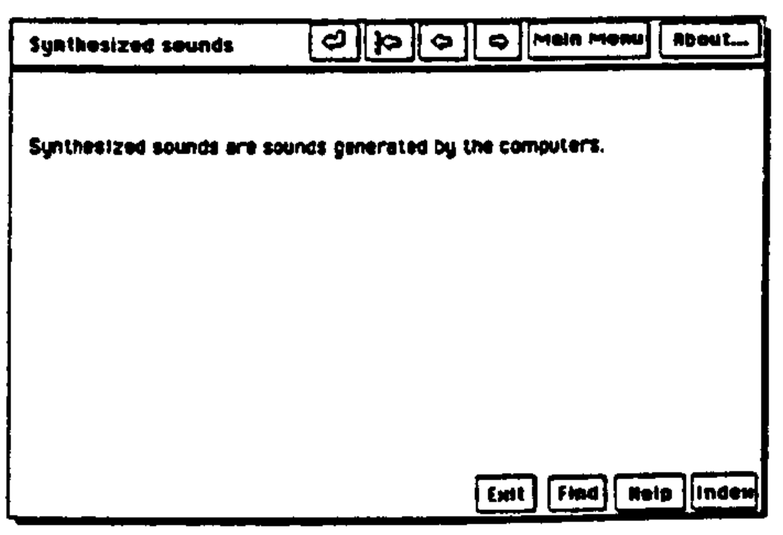

Matrix line printer
Main Menu
About...
Matrix line printers print one line of dots at a time Needlelike hammers are lined up across the width of the paper The characters are formed in rectangular dot configurations as the paper passes the line of print hammers.
Matrix line printers are much more flexible than band printers are They can also perform the same types of print operations as dot-matrix printers do, including graphic output and machine readable bar codes.
Exit
Find
Help
Index

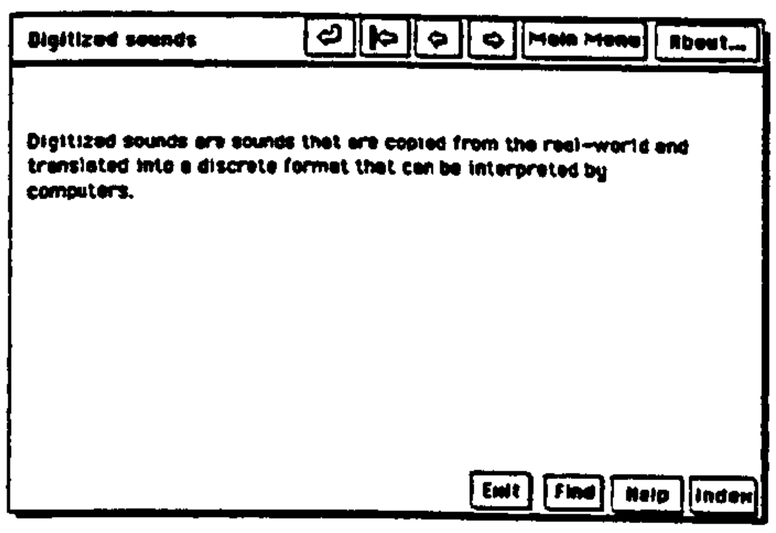

Page Printers
Main Menu
About...
Page printers are of the nonimpact type and use laser printing technology to achieve high-speed hard copy output by printing a page at a time The fastest page printer can produce almost a quarter of a million pages and that's 50 miles of output
Page printers are used in the following examples
• in banks to produce statements for checking.
• in insurance companies to print policies
• in electric utility companies to bill customers
Exit
Find
Help
Index

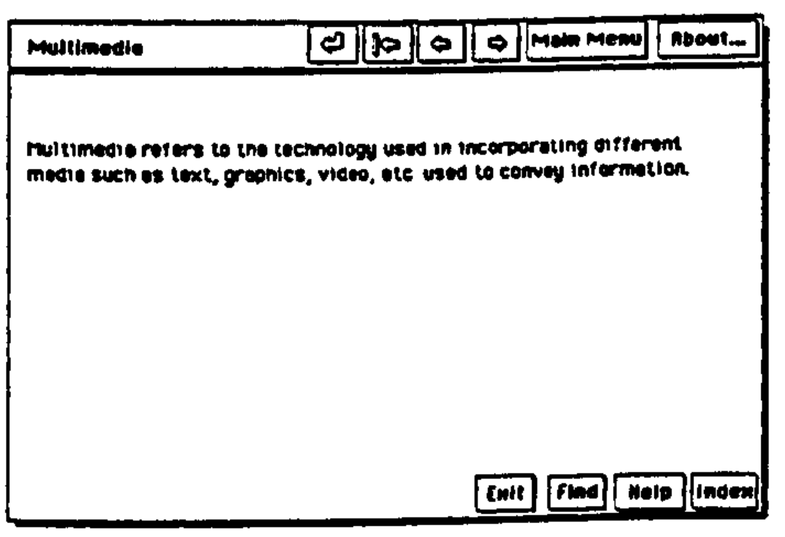

Impact printers
Main Menu
About...
An impact printer uses some type of hammer or hammers to hit the ribbon and the paper, much as a typewriter does.
Exit
Find
Help
Index

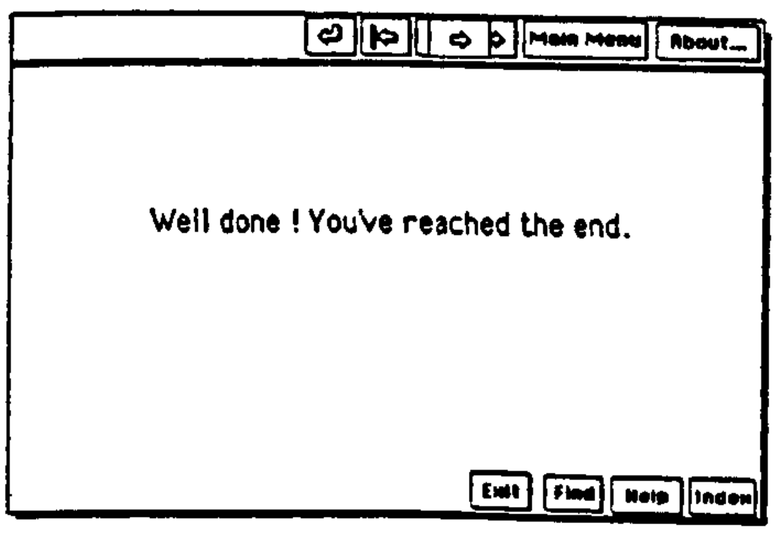

Nonimpact printers
Main Menu
About...
Nonimpact printers use chemicals, lasers and heat to form the images on the paper Only nonimpact printers can achieve print speeds in excess of 3600 lines per minute
Exit
Find
Help
Index


Audio Output
Main Menu
About...
Audio output from the computer takes several forms:
• synthesized sounds (music, speech and sound effects) mathematically generated by the computer
• digitized sounds (music, speech and sound effects) copied from real-world sounds
• Computer control of audio/video multimedia equipment
Exit
Find
Help
Index


Synthesized sounds
Main Menu
About...
Synthesized sounds are sounds generated by the computers.
Exit
Find
Help
Index


Digitized sounds
Main Menu
About...
Digitized sounds are sounds that are copied from the real-world and translated into a discrete format that can be interpreted by computers.
Exit
Find
Help
Index


Multimedia
Main Menu
About...
Multimedia refers to the technology used in incorporating different media such as text, graphics, video, etc used to convey information.
Exit
Find
Help
Index


Main Menu
About...
Well done ! You've reached the end.
Exit
Find
Help
Index

Corresponding sample HyperCard stack script

```
on openStack
  lock screen
  lowerUserLevel
  deleteOtherMenus
  disableGoMenuItems
  addNavigateMenu
  push card
  unlock screen
  hereweare
end openStack

on resumestack
  hereweare
end resumestack

on hereweare
  global mystacks
  put long name of this stack after last line of mystacks
  answer mystacks
end hereweare

on closeStack
  global oldUserLevel
  set userLevel to oldUserLevel
  hide card field "Thanks" of card 1
  reset menuBar
end closeStack

on lowerUserLevel
  global oldUserLevel
  get userLevel
  put It into oldUserLevel
  set userLevel to 1
end lowerUserLevel

--   MENU-HANDLING SCRIPTS   --

on addNavigateMenu
  show menubar
  create menu "Navigate"
```

```
  put NavigateItems() into menu "Navigate" ¬
  with menuMessages NavigateMessages()
end addNavigateMenu

on deleteOtherMenus
  delete menu "Edit"
  delete menu "File"
end deleteOtherMenus

on disableGoMenuItems
  disable menuItem "Recent" of menu "Go"
  disable menuItem "Help" of menu "Go"
end disableGoMenuItems

function NavigateItems
  return "Title Card, Main Menu, ---------------, Find, Exit"
end NavigateItems

function NavigateMessages
  return "doMenu First, goMain, empty, doFind, doExit"
end NavigateMessages

on goMain
  send mouseUp to bg button "Main Menu"
end goMain

on doFind
  send mouseUp to bg button "Find"
end doFind

on doExit
  send mouseUp to bg button "Exit"
end doExit
```

Appendix F2

A sample Lisp program for automatic conversion to HTML format

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : convert-html-process
;;; Arguments: nil
;;; Objective: To call functions (convert-html-single) and (convert-
;;;            html-part) to create html file as part or whole

(defun convert-html-process ()
;(print 'convert-html-process)
  (convert-html-part (reverse *nodes-created*))
  (setq *hypertext-indicator-5* 1)

)


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : convert-html-part
;;; Arguments: list of nodes (*node-store*)
;;; Objective: To read from data structure of nodes and convert them
;;;            into html format. Output is a hyperdocument in separate
;;;            pages. Will also incorporate within the facility to
;;;            print the hyperdocument into a single file.
;;;

 (defun convert-html-part (x)
   ;(print 'convert-html-part)
   (buffer-delete (fred-buffer *message-box*) 0)
   (fred-update *message-box*)
   (empty-window3)
    (find-hyperdocument)
    (get-designer-name)
    (write-function-content x)
    (converter x)
    (insert-char *message-box* "Updating complete...")
    (fred-update *message-box*)
    (window-close *window1*)
    (window-save *window3*)
    (window-close *window3*)

)
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : write-to-hyperdocument
;;; Arguments: nil
;;; Objective: To write the html codes into a hyperdocument.html that
;;;             will contain the codes for a single document
;;;


(defun write-to-hyperdocument ()
  ;(print 'write-to-hyperdocument)
   (paste *window3*)

   )


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : get-designer-name
;;; Arguments: nil
;;; Objective: Allow designer to input name for signing off
;;;
;;;

(defvar *name* nil)
(defun get-designer-name ()
  ;(print 'get-designer-name)
   (setf *name* (get-string-from-user "Enter your name."))

  )


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : converter
;;; Arguments: list of nodes
;;; Objective: Using a recursive function to convert the nodes into
;;;             separate html pages
;;;

 (defun converter (x)
   ;(print 'converter)
    (let ((y))

     (cond ((endp x) nil)
           (t
            (setf y (car x))
            (find-temp)
            (empty-temp *windowtemp*)
            (write-function-comment)
            (write-function-html y)
            (write-to-file y)
            (setf x (cdr x))
            (converter x)
            )
           )
      )
    )
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : empty-temp
;;; Arguments:'fred-window
;;; Objective: To delete the contents in the window
;;;
;;;


(defun empty-temp (window)
  ;(print 'empty-temp)
  (buffer-delete (fred-buffer window) 0)
  (fred-update window)

  )

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : find-temp
;;; Arguments: nil
;;; Objective: To find the file "temp.lisp" in the hard disk
;;;
;;;


(defvar *windowtemp* nil)

(defun find-temp ()
;(print 'find-temp)
(cond ((eq nil (find-window "temp.lisp {Hard Disk:lisp-datafile:test-
      data:}"))
         (setq *windowtemp* (make-instance 'fred-window
                              :filename (make-pathname :directory "Hard
                                       Disk:lisp-datafile:test-data:"
                                                      :name "temp"
                                                      :type "lisp")
                                  :window-layer 3))
           (set-mark (fred-buffer *windowtemp*) t))
        (t (set-mark (fred-buffer *windowtemp*) t))))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : Write-function-html
;;; Arguments: *windowtemp* 'fred-window
;;; Objective: To hard code ht into html format
;;;
;;;

(defun write-function-html (page)
  ;(print 'write-function-html)
  (set-part-color *windowtemp* :text *red-color*)
  (write-function-navigation-button page)
  (write-function-header page)
  (write-function-body page)
  (write-function-navigation-button page)
)
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : Write-function-comment
;;; Arguments: *windowtemp* 'fred-window
;;; Objective: To comment on the nodes written into the html file

(defun write-function-comment ()
  ;(print 'write-function-comment)
  (insert-char *windowtemp* "<!-- HTML format of hypertext -->")
  (insert-line *windowtemp*)
  (insert-char *windowtemp* "<html>")

  )

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : Write-function-signing-off
;;; Arguments: nil
;;; Objective: To sign off with name, date and time

(defun write-function-signing-off ()
  ;(print 'write-function-signing-off)

  (let ((time-list) (second) (minute) (hour) (date) (month)
        (year))

    (setf time-list (multiple-value-bind (a b c d e f)
                                          (get-decoded-time)
                                          (list a b c d e f)))
    (setq second (nth 0 time-list))
    (setq minute (nth 1 time-list))
    (setq hour (nth 2 time-list))
    (setq date (nth 3 time-list))
    (setq month (nth 4 time-list))
    (setq year (nth 5 time-list))
    (insert-char *windowtemp* "<hr>")
    (insert-char *windowtemp* "<l>")
    (insert-line *windowtemp*)
    (insert-char *windowtemp* "Copyright by ")
    (insert-char *windowtemp* *name*)
    (insert-line *windowtemp*)
    (insert-char *windowtemp* "<br>")
    (insert-char *windowtemp* (princ-to-string date))
    (insert-char *windowtemp* "/")
    (insert-char *windowtemp* (princ-to-string month))
    (insert-char *windowtemp* "/")
    (insert-char *windowtemp* (princ-to-string year))
    (insert-char *windowtemp* "  ")
    (insert-char *windowtemp* (princ-to-string hour))
    (insert-char *windowtemp* ":")
    (insert-char *windowtemp* (princ-to-string minute))
    (insert-char *windowtemp* ":")
    (insert-char *windowtemp* (princ-to-string second))
    (insert-line *windowtemp*)
    (insert-char *windowtemp* "</br>")
    (insert-char *windowtemp* "</l>")
    (insert-char *windowtemp* "</html>")
    )
  )
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : write-function-content
;;; Arguments: list of nodes
;;; Objective: To create the contents page using the information in
;;;              the root node

(defun write-function-content (x)
  ;(print 'write-function-content)
  (find-temp)
   (empty-temp *windowtemp*)
   (write-function-comment)
   (insert-char *windowtemp* "<title>")
   (insert-char *windowtemp* "Table of Contents")
   (insert-char *windowtemp* "</title>")
   (insert-char *windowtemp* "<h1>")
   (insert-char *windowtemp* "Table of Contents")
   (insert-char *windowtemp* "</h1>")
   (insert-char *windowtemp* "<hr>")
   (insert-char *windowtemp* "A ")
   (insert-char *windowtemp* "<a href=\"hyperdocument.html\">")
   (insert-char *windowtemp* "single ")
   (insert-char *windowtemp* "</a>")
   (insert-char *windowtemp* "document is available for easier
      printing.")
   (insert-char *windowtemp* "<br>")
   (insert-char *windowtemp* "<li>")
      (insert-char *windowtemp* "<a href=\"")
      (insert-char *windowtemp* (princ-to-string (car x)))
      (insert-char *windowtemp* ".html")
      (insert-char *windowtemp* "\"")
      (insert-char *windowtemp* ">")
      (insert-char *windowtemp* (princ-to-string (car x)))
      (insert-char *windowtemp* "</a>")
   (dolist (element x)
         (cond ((equal (hypertext-level-structure-parent-node-name
                   (symbol-value element)) nil)
           (display-further-info element))))
   (insert-char *windowtemp* "<hr>")
   (write-function-signing-off)
    (write-to-content))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : write-to-content
;;; Arguments: nil
;;; Objective: To write the contents from a temp file into a new file
;;;            for contents node
;;;

(defun write-to-content ()
  ;(print 'write-to-content)
  (select-all *windowtemp*)
  (copy *windowtemp*)
  (cond ((eq (find-window "contents.html {Hard Disk:lisp-datafile:
          test-data:}") nil)
      (setq *windowperm* (make-instance 'fred-window
                          :filename
      (make-pathname :directory "Hard Disk:lisp-datafile:test-data:"
                                   :name "contents"
                                   :type "html"))))

       (t (fred (make-pathname :directory "Hard Disk:lisp-datafile:
                          test-data:"
                          :name "contents"
                          :type "html"))))
   (empty-temp *windowperm*)
   (paste *windowperm*)
   (write-to-hyperdocument)
   (buffer-delete (fred-buffer *windowtemp*) 0)
   (fred-update *windowtemp*)
   (window-save (find-window "temp.lisp {Hard disk:lisp-datafile:test-
          data:}"))
   (window-close (find-window "temp.lisp {Hard disk:lisp-
      datafile:test-data:}"))
   (window-save (find-window "contents.html {Hard disk:lisp-
      datafile:test-data:}"))
   (window-close (find-window "contents.html {Hard disk:lisp-
      datafile:test-data:}")))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : Write-function-header
;;; Arguments: list
;;; Objective: To hardcode header into html format
;;;
;;;

(defun write-function-header (x)
  ;(print 'write-function-header)

  (let ((node))
    (insert-char *windowtemp* "<title>")
    (insert-char *windowtemp* (princ-to-string
                               (hypertext-level-structure-window-name-
      sub (symbol-value x))))
    (insert-char *windowtemp* "</title>")
    (header-table x)
    (insert-line *windowtemp*)
    (insert-line *windowtemp*)))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function  : Header-table
;;; Arguments : list
;;; Objective: To hardcode header into html format, calls from
;;;            function "Write-function-header"

(defun header-table (x)
  ;(print 'header-table)
  (insert-char *windowtemp* "<TABLE BORDER = 0 CELLPADDING=10
     CELLSPACING=10><TR><TD><IMG SRC=\"")
  (insert-char *windowtemp* (princ-to-string
                            (hypertext-level-structure-icon-name-sub
  (symbol-value x))))
  (insert-char *windowtemp* "\"")
   (insert-char *windowtemp* " WIDTH = 154 HEIGHT = 154 LOWSRC=\"")
   (insert-char *windowtemp* (princ-to-string
                             (hypertext-level-structure-icon-name-sub
   (symbol-value x))))
   (insert-char *windowtemp* "\"")
    (insert-char *windowtemp* " WIDTH = 154 HEIGHT = 154></TD>
                              <TD COL=2 ALIGN = TOP><PRE><FONT
                              SIZE=7>")
    (insert-char *windowtemp* "<B>")
    (insert-char *windowtemp* (princ-to-string
          (string-capitalize
               (hypertext-level-structure-window-name-sub (symbol-
          value x)))))
    (insert-char *windowtemp* "</FONT></B>")
     (insert-char *windowtemp* "</PRE><IMG SRC=\"")
     (insert-char *windowtemp* "grey3.gif")
     (insert-char *windowtemp* "\"")
    (insert-char *windowtemp* " WIDTH = 240 HEIGHT =15>
          </TD></TR></TABLE><P>"))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : parse-node-name-window
;;; Arguments: list-string (string), start-pos (number)
;;; Objective: To call function to get substrings from string

(defun parse-node-name-window (string start-pos)
  ;(print 'parse-node-name-window)
  (find-string-text)
  (buffer-delete (fred-buffer *windowstring*) 0)
  (fred-update *windowstring*)
  (insert-char *windowstring* string)
  (fred-update *windowstring*)
  (parse-string-word-window start-pos '())
  (window-save *windowstring*)
  (window-close *windowstring*))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : parse-string-word-window
;;; Arguments: start-pos (number), list-string (string)
;;; Objective: To get substrings from string

(defvar *convert-list* nil)

(defun parse-string-word-window (start-pos list-string)
  ;(print 'parse-string-word-window)
  (let ((hyphenpos)(wordstring)(find-file))
    (setf find-file (find-window (window-title *windowstring*)))
    (setf hyphenpos (buffer-string-pos (fred-buffer find-file) "-"
                                       :start start-pos))
    (setf wordstring
          (buffer-substring (fred-buffer find-file) start-pos
          hyphenpos))
    (setq *convert-list* (concatenate 'string *convert-list* " "
           wordstring))
    (cond ((equal hyphenpos nil) nil)
          (t (setf start-pos (+ 1 hyphenpos))
             (parse-string-word-window start-pos list-string)))))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : find-string-text
;;; Arguments: nil
;;; Objective: To find the window for putting the node-name string,
;;;            and then parsing it


(defvar *windowstring* nil)
(defun find-string-text ()
;(print 'find-string-text)
(cond ((eq nil (find-window "string.lisp {Hard Disk:lisp-datafile:}"))
         (setq *windowstring* (make-instance 'fred-window
                             :filename (make-pathname :directory "Hard
                                                Disk:lisp-datafile:"
                                                     :name "string"
                                                     :type "lisp")

                              :window-layer 3
                              :wrap-p t ))

         (set-mark (fred-buffer *windowstring*) t))
        (t (set-mark (fred-buffer *windowstring*) t))))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : parse-node-name
;;; Arguments: string, start-pos (number)
;;; Objective: Function to call functions to parse

(defun parse-node-name (string start-pos)
  ;(print parse-node-name)
  (find-string-text)
  (buffer-delete (fred-buffer *windowstring*) 0)
  (fred-update *windowstring*)
  (insert-char *windowstring* string)
  (fred-update *windowstring*)
  (parse-string-word start-pos '())
  (window-save *windowstring*)
  (window-close *windowstring*))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : parse-string-word
;;; Arguments: start-pos (number), list-string (string)
;;; Objective: To get substrings from string
;;;
;;;

(defvar *convert-list* nil)
(defun parse-string-word (start-pos list-string)
  ;(print 'parse-string-word)
  (let ((blankpos)(wordstring)(find-file))
    (setf find-file (find-window (window-title *windowstring*)))
    (setf blankpos (buffer-string-pos (fred-buffer find-file) " "
                                      :start start-pos))
    (setf wordstring
          (buffer-substring (fred-buffer find-file) start-pos
           blankpos))
     (setq *convert-string* (concatenate 'string *convert-string* "-"
           wordstring))
    (cond ((equal blankpos nil) nil)
          (t (setf start-pos (+ 1 blankpos))
           (parse-string-word start-pos list-string)))))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : Write-function-body
;;; Arguments: *windowtemp* 'fred-window
;;; Objective: To write the node text into *windowtemp*

(defun write-function-body (x)
  ;(print 'write-function-body)
  (insert-char *windowtemp* "<body>")
  (insert-line *windowtemp*)
  (insert-char *windowtemp* (princ-to-string
     (hypertext-level-structure-node-text-sub (symbol-value x))))
  (insert-line *windowtemp*)
  (insert-char *windowtemp* "<br><br>")
  (insert-line *windowtemp*)

  ;;; to insert the index for that level
  (cond ((equal (hypertext-level-structure-children-nodes (symbol-
        value x)) nil) nil)
       (t
        (insert-char *windowtemp* "<hr>")
        (insert-char *windowtemp* "Related information on ")
        (insert-char *windowtemp* (princ-to-string
             (hypertext-level-structure-node-name-sub
             (symbol-value x))))
        (insert-char *windowtemp* " ... ")
        (insert-char *windowtemp* "<br>")
        (insert-line *windowtemp*)
        (insert-line *windowtemp*)
        (display-further-info x)))
  (insert-char *windowtemp* "</ul>")
  (insert-line *windowtemp*)
  (insert-char *windowtemp* "<hr>")
  (insert-line *windowtemp*)
  (insert-char *windowtemp* "</body>")
  (insert-line *windowtemp*))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : display-further-info
;;; Arguments: a node
;;; Objective: To find all the descendants of a node
;;;

(defun display-further-info (x)
  ;(print 'display-further-info)
  (let ((children))
    (setq children (hypertext-level-structure-children-nodes (symbol-
          value x)))
    (if children (insert-char *windowtemp* "<ul>"))
    (dolist (child children)
      (insert-char *windowtemp* "<li>")
      (insert-char *windowtemp* "<a href=\"")
      (insert-char *windowtemp* (princ-to-string child))
      (insert-char *windowtemp* ".html")
      (insert-char *windowtemp* "\"")
      (insert-char *windowtemp* ">")
      (insert-char *windowtemp* (princ-to-string child))
      (insert-char *windowtemp* "</a>")
```

```
      (display-further-info child))
    (if children (insert-char *windowtemp* "</ul>"))))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : Write-function-navigation-button
;;; Arguments: *windowtemp* 'fred-window
;;; Objective: To hardcode the position of the navigation buttons
;;;            consisting of home, help, ->, <-, hierarchical
;;;            directory (index)
;;;


(defun write-function-navigation-button (x)
 ;(print 'write-function-navigation-button)
;;Contents button
        (insert-char *windowtemp* "<hr>")
        (insert-char *windowtemp* "<a href=\"")
        (insert-char *windowtemp* "contents.html\"")
        (insert-char *windowtemp* ">")
        (insert-char *windowtemp* "<img src=\"")
        (insert-char *windowtemp* "contents_motif.gif\"")
        (insert-char *windowtemp* " width=50 height=30 alt=\"")
        (insert-char *windowtemp* "Contents Page\"")
        (insert-char *windowtemp* ">")
        (insert-char *windowtemp* "</a>")
        (insert-line *windowtemp*)

;;Previous button should be generated, that is, parent node
        (cond ((equal (hypertext-level-structure-parent-node-name
               (symbol-value x)) nil))
              (t
               (insert-char *windowtemp* "<a href=\"")
               (insert-char *windowtemp* (princ-to-string
                      (car (hypertext-level-structure-parent-node-
                           name (symbol-value x)))))
               (insert-char *windowtemp* ".html")
               (insert-char *windowtemp* "\"")
               (insert-char *windowtemp* ">")
               (insert-char *windowtemp* "<img src=\"")
               (insert-char *windowtemp* "previous_motif.gif\"")
               (insert-char *windowtemp* " width=50 height=30
                     alt=\"")
               (insert-char *windowtemp* "Previous Page\"")
               (insert-char *windowtemp* ">")
               (insert-char *windowtemp* "</a>")
               (insert-line *windowtemp*)
               )
              )

;;Next button should be generated, that is, child node
 (cond ((equal (hypertext-level-structure-children-nodes
        (symbol-value x)) nil))
       (t (insert-char *windowtemp* "<a href=\"")
        (insert-char *windowtemp* (princ-to-string
           (car (hypertext-level-structure-children-nodes
                (symbol-value x)))))
        (insert-char *windowtemp* ".html")
        (insert-char *windowtemp* "\"")
        (insert-char *windowtemp* ">")
```

```
(insert-char *windowtemp* "<img src=\"")
(insert-char *windowtemp* "next_motif.gif\"")
(insert-char *windowtemp* " width=50 height=30 alt=\"")
(insert-char *windowtemp* "Next Page\"")
(insert-char *windowtemp* ">")
(insert-char *windowtemp* "</a>")
(insert-line *windowtemp*)))
(insert-char *windowtemp* "<br>")
(insert-line *windowtemp*)
(insert-char *windowtemp* "Contents: ")
(insert-char *windowtemp* "<a href=\"")
(insert-char *windowtemp* "contents.html")
          (insert-char *windowtemp* "\"")
          (insert-char *windowtemp* ">")
          (insert-char *windowtemp* "Table of Contents")
          (insert-char *windowtemp* "</a>")
 (cond ((equal (hypertext-level-structure-parent-node-name
          (symbol-value x)) nil))
        (t (insert-char *windowtemp* "Previous: ")
         (insert-char *windowtemp* "<a href=\"")
         (insert-char *windowtemp* (princ-to-string
                   (car (hypertext-level-structure-parent-node-name
                        (symbol-value x)))))
         (insert-char *windowtemp* ".html")
         (insert-char *windowtemp* "\"")
         (insert-char *windowtemp* ">")
         (insert-char *windowtemp* (princ-to-string
                                (car (hypertext-level-structure-
parent-node-name (symbol-value x))))))
         (insert-char *windowtemp* "</a>")))
(cond ((equal (hypertext-level-structure-children-nodes
          (symbol-value x)) nil))
        (t (insert-line *windowtemp*)
         (insert-char *windowtemp* "Next: ")
         (insert-char *windowtemp* "<a href=\"")
         (insert-char *windowtemp* (princ-to-string
                   (car (hypertext-level-structure-children-nodes
                   (symbol-value x)))))
         (insert-char *windowtemp* ".html")
         (insert-char *windowtemp* "\"")
         (insert-char *windowtemp* ">")
         (insert-char *windowtemp* (princ-to-string
                   (car (hypertext-level-structure-children-nodes
                   (symbol-value x)))))
         (insert-char *windowtemp* "</a>")))
(insert-char *windowtemp* "</br>")
(insert-char *windowtemp* "</br>")
(insert-char *windowtemp* "<hr>"))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : write-to-file
;;; Arguments: a node
;;; Objective: To write the contents from a temp file into a new file
;;;            for each node

(defvar *windowperm* nil)
(defun write-to-file (y)
  ;(print 'write-to-file)
  (select-all *windowtemp*)
  (copy *windowtemp*)
  (create-html-file y)
  (open-html-file y)
  (paste *windowperm*)
  (write-to-hyperdocument)
  (window-save *windowperm*)
   (window-close *windowperm*)
  (buffer-delete (fred-buffer *windowtemp*) 0)
   (fred-update *windowtemp*)
   (window-save (find-window "temp.lisp {Hard disk:lisp-datafile:
                 test-data:}"))
   (window-close (find-window "temp.lisp {Hard disk:lisp-datafile:
                 test-data:}")))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : create-html-file
;;; Arguments: a node
;;; Objective: Create a new file or open existing file
;;;

(defun create-html-file (y)
;(print 'create-html-file)
(with-open-file (ifile (make-pathname :directory "Hard Disk:lisp-
                                       datafile:test-data:"
                                      :name (princ-to-string y)
                                      :type "html")
                       :direction :io
                       :if-exists :supersede
                       :if-does-not-exist :create)
                 ifile))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; Function : open-html-file
;;; Arguments: a node
;;; Objective: To open the file that has been created by
;;;            create-html-file function into a fred-window

(defun open-html-file (y)
;(print 'open-html-file)
(cond ((eq nil (find-window "y.lisp {Hard Disk:lisp-datafile:
               test-data:}"))
       (setq *windowperm* (make-instance 'fred-window
           :filename (make-pathname :directory "Hard Disk:
                      lisp-datafile:test-data:"
                     :name (princ-to-string y)
                     :type "html")))
       (set-mark (fred-buffer *windowperm*) t))
      (t (set-mark (fred-buffer *windowperm*) t))))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : get-node-string-comma
;;; Arguments: string
;;; Objective: To parse the string

(defvar *convert-list* nil)
(defun get-node-string-comma (string)
  ;(print 'get-node-string-comma)
  (setf *convert-list* nil)
  (setf *parent-list* nil)
  (find-string-text)
  (buffer-delete (fred-buffer *windowstring*) 0)
  (fred-update *windowstring*)
  (parse-node-name-comma 0 string))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : parse-node-name-comma
;;; Arguments: integer, string
;;; Objective: To pick out parts of string before a comma

(defun parse-node-name-comma (start-pos string)
 ;(print 'parse-node-name-comma)
  (insert-char *windowstring* string)
  (fred-update *windowstring*)
  (parse-string-word-comma start-pos)
    (window-save *windowstring*)
  (window-close *windowstring*)
  (window-save *windowstring1*)
  (window-close *windowstring1*))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : parse-string-word-comma
;;; Arguments: integer
;;; Objective: To pick out parts of string before a comma

(defun parse-string-word-comma (start-pos)
  ;(print 'parse-string-word-comma)
  (let ((commapos)(wordstring)(find-file))
    (setf find-file (find-window (window-title *windowstring*)))
    (setf commapos (buffer-string-pos (fred-buffer find-file) ","
                                       :start start-pos))
    (setf wordstring
          (buffer-substring (fred-buffer find-file) start-pos
                     commapos))
     (find-string-text1)
     (buffer-delete (fred-buffer *windowstring1*) 0)
     (fred-update *windowstring1*)
     (insert-char *windowstring1* wordstring)
     (fred-update *windowstring1*)
     (setf *convert-list* nil)
    (parse-string-word-window 0)
    (setq *parent-list* (append (list (get-sexpr *convert-list*))
            *parent-list*))
    (cond ((equal commapos nil) nil)
          (t (setf start-pos (+ 1 commapos))
           (parse-string-word-comma start-pos)))))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : parse-string-word-window
;;; Arguments: integer
;;; Objective: To pick out parts of string before a space

(defvar *parent-list* nil)
(defun parse-string-word-window (start-pos)
  ;(print 'parse-string-word-window)
  (let ((hyphenpos)(wordstring)(find-file))
    (setf find-file (find-window (window-title *windowstring1*)))
    (setf hyphenpos (buffer-string-pos (fred-buffer find-file) " "
                                       :start start-pos))
    (setf wordstring
          (buffer-substring (fred-buffer find-file) start-pos
               hyphenpos))
    (setq *convert-list* (concatenate 'string *convert-list* "-"
               wordstring))
    (setq *convert-list* (string-trim "-" *convert-list*))
    (cond ((equal hyphenpos nil) nil)
          (t (setf start-pos (+ 1 hyphenpos))
             (parse-string-word-window start-pos)))))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Function : find-string-text1
;;; Arguments: none
;;; Objective: window to write string

(defvar *windowstring1* nil)
(defun find-string-text1 ()
  ;(print 'find-string-text1)
  (cond ((eq nil (find-window "string1.lisp {Hard Disk:
                lisp-datafile:}"))
    (setq *windowstring1* (make-instance 'fred-window
        :filename (make-pathname :directory "Hard Disk:lisp-datafile:"
        :name "string1"
        :type "lisp")
        :window-layer 3
        :wrap-p t))
         (set-mark (fred-buffer *windowstring1*) t))
        (t (set-mark (fred-buffer *windowstring1*) t))))
```

Appendix G

Generated data structures

```
(setf HOME
(make-hypertext-structure
:file-name 'childnet
:window-name '(childnet international)
:node-name 'HOME
:icon-name 'smile.gif
:node-text
"Promoting the interests of Children in International Communications.<p>Welcome to the home page of <B>Childnet International</B>. We are a new non-profit organisation concerned to enable children to benefit from all the changes in international communications, and to protect them from any negative influences <p>"
:children-nodes '(ATLASWWW NEW INDEX)
:neighbour-nodes 'NIL
:parent-node-name 'NIL
))


(setf INDEX
(make-hypertext-structure
:file-name 'childnet
:window-name '(childnet international - index)
:node-name 'INDEX
:icon-name 'boy3-ci.gif
:node-text
"<TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=0><TR>
<TD VALIGN=top ALIGN = RIGHT><IMG SRC=\"orgline.gif\" WIDTH = 130 HEIGHT =15> <a href =\"research.html\"><IMG SRC=\"smbut1.gif\" WIDTH = 28 HEIGHT=28 border =0></A></TD>
<TD COL=2><a href =\"research.html\"><FONT SIZE=6>Research</A></FONT><BR> our research plans and developments elsewhere</TD></TR><TR>
<TD ALIGN =RIGHT> <a href =\"projects.html\"><FONT SIZE=6>Positive projects</FONT></A><BR> linking children around the globe. Contribute details of your own project </TD>
<TD COL=2 VALIGN=top><a href =\"projects.html\"><IMG SRC=\"smbut2.gif\" WIDTH = 28 HEIGHT= 28 border = 0></A><IMG SRC=\"orgline.gif\" WIDTH = 120 HEIGHT =15></TD></TR><TR>
<TD VALIGN=top ALIGN =RIGHT><IMG SRC=\"orgline.gif\" WIDTH = 120 HEIGHT =15><a href =\"practice.html\"><IMG SRC=\"smbut3.gif\" WIDTH = 28 HEIGHT= 28 border = 0></A></TD>
<TD COL=2><a href =\"practice.html\"><FONT SIZE=6>Promoting good practice in the industry</FONT> </A><BR> do you have an example of child-friendly actions by a computer or communications company?</TD></TR><TR>
<TD ALIGN =RIGHT><a href =\"education.html\"> <FONT SIZE=6> Educating and informing</FONT> </A><BR> helping parents, educators, governments and us all get to grips with children's interests in international communications</TD>
<TD COL=2 VALIGN=top><a href =\"education.html\"><IMG SRC=\"smbut4.gif\" WIDTH = 28 HEIGHT= 28 border =0></A><IMG SRC=\"orgline.gif\" WIDTH = 140 HEIGHT =15></TD></TR><TR>
<TD VALIGN=top ALIGN =RIGHT><IMG SRC=\"orgline.gif\" WIDTH = 145 HEIGHT =15><a href =\"who.html\"><IMG SRC=\"smbut5.gif\" WIDTH = 28 HEIGHT= 28 border = 0></A></TD>
<TD COL=2><a href =\"who.html\"> <FONT SIZE=6>Who we are</FONT></A> <BR>and how to reach us</TD></TR><TR>
<TD ALIGN =RIGHT> <a href =\"new.html\"><FONT SIZE=6>What's new</FONT></A><BR> new information on this site </TD><TD COL=2 VALIGN=top><a href =\"new.html\"><IMG SRC=\"smbut2.gif\" WIDTH = 28 HEIGHT= 28 border = 0></A><IMG SRC=\"orgline.gif\" WIDTH = 120 HEIGHT =15></TD></TR></TABLE>"
:children-nodes '(ATLASWWW NEW WHO EDUCATION PRACTICE PROJECTS RESEARCH)
:neighbour-nodes '(NEW WHO EDUCATION PRACTICE PROJECTS RESEARCH)
:parent-node-name '(HOME)
))
```

```
(setf WHO
(make-hypertext-structure
:file-name 'childnet
:window-name '(who we are?)
:node-name 'WHO
:icon-name 'smile.gif
:node-text
"Childnet International is a non-profit organisation with its international headquarters in London, England. It is registered in the UK as a charity No 1053193.<P>Childnet International also operates in the USA as an independent non-profit organisation.<P>The Director of Childnet is Nigel Williams. A Cambridge University graduate with experience as a government official, the computer industry and the voluntary sector he has been a strong advocate of the interests of children, especially with reference to the impact of the media. <P>
Nigel has researched and written on the legal framework affecting the content of media communications. Married to Heather he has four computer literate children. <P>Nigel has travelled widely especially in the USA, Europe and Asia.<P>
Nigel can be contacted at:<BR>
Childnet International,<BR>
35 Piccadilly,<BR>
LONDON, W1V 9PB<BR>
Voice: 0171 525 9014<BR>
Fax : 0171 701 1418 <BR>
<IMG SRC=\"orgline.gif\" WIDTH = 340 HEIGHT =15><P>The US Director of Childnet International is Nuala Holowicki. She received an honors degree in law from Queen's University Belfast and a Master of Laws from Wayne State University in Detroit. She was admitted to the State Bar of Michigan in 1995.<P>Nuala has wide legal experience including some child abuse cases. Her husband Gerry works in the computer industry.<P>
Nuala can be contacted at: <BR>
317 South Division, #198<BR>
Ann Arbor, MI 48104<BR>
USA<P>
Voice and fax: (313) 572 4595<BR>"
:children-nodes '(NUALA NIGEL)
:neighbour-nodes 'NIL
:parent-node-name '(INDEX)
))
```

```
(setf NIGEL
(make-hypertext-structure
:file-name 'childnet
:window-name '(nigel's home page)
:node-name 'NIGEL
:icon-name 'man.gif
:node-text
"The Director of Childnet is Nigel Williams. A Cambridge University graduate with experience as a government official, the computer industry and the voluntary sector he has been a strong advocate of the interests of children, especially with reference to the impact of the media. <P>
Nigel has researched and written on the legal framework affecting the content of media communications. Married to Heather he has four computer literate children. <P> Nigel has travelled widely especially in the USA, Europe and Asia.<P>
Nigel can be contacted at:<BR>
Childnet International,<BR>
35 Piccadilly,<BR>
LONDON, W1V 9PB<BR>
Voice: 0171 525 9014<BR>
Fax : 0171 701 1418 <BR>"
:children-nodes 'NIL
:neighbour-nodes 'NIL
:parent-node-name '(WHO)
))
```

```
(setf NUALA
(make-hypertext-structure
:file-name 'childnet
:window-name '(nuala's home page)
:node-name 'NUALA
:icon-name 'woman.gif
:node-text
"The US Director of Childnet International is Nuala Holowicki. She received an honors degree in law from Queen's University Belfast and a Master of Laws from Wayne State University in Detroit. She was admitted to the State Bar of Michigan in 1995.<P>Nuala has wide legal experience including some child abuse cases. Her husband Gerry works in the computer industry.<P>
Nuala can be contacted at: <BR>
317 South Division, #198<BR>
Ann Arbor,  MI 48104<BR>
USA<P>
Voice and fax:  (313) 572 4595<BR>"
:children-nodes 'NIL
:neighbour-nodes 'NIL
:parent-node-name '(WHO)
))


(setf NEW
(make-hypertext-structure
:file-name 'childnet
:window-name '(what's new)
:node-name 'NEW
:icon-name 'smile.gif
:node-text
"<A HREF =\"safety.html\">CHILD SAFETY ON THE INTERNET</A><P></B>Summary of Video Conference held between Capitol Hill, Washington DC and Westminster, London on Wednesday 10 July 1996<P>"
:children-nodes '(SAFETY)
:neighbour-nodes '(INTRO)
:parent-node-name '(INDEX HOME)
))
```

Appendix H

Generated html files

Table of Contents

<!-- HTML format of hypertext -->
<html><title>Table of Contents</title><h1>Table of Contents</h1><hr>A <a
href="hyperdocument.html">single document is available for easier printing.
<a
href="HOME.html">HOME
ATLASWWW
<a
href="NEW.html">NEW
SAFETY
<a
href="PRINT.html">PRINT
INDEX
<a
href="ATLASWWW.html">ATLASWWW<li>NEW
<a
href="SAFETY.html">SAFETY
PRINT
<a
href="WHO.html">WHO
NUALA
<a
href="NIGEL.html">NIGEL
EDUCATION
<a
href="PRACTICE.html">PRACTICE<li>PROJECTS
<a
href="RESEARCH.html">RESEARCH
<hr><hr><l>
Copyright by peter

7/7/1997 13:33:41
</br></l></html>

Contents page

<!-- HTML format of hypertext -->
<html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents
Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous
Page"><img src="next_motif.gif" width=50 height=30 alt="Next
Page">
Contents: Table of ContentsPrevious: <a
href="HOME.html">HOMENext: <a
href="ATLASWWW.html">ATLASWWW</br></br><hr><title>childnet international -
index</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="boy3-
ci.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="smile.gif" WIDTH = 154 HEIGHT = 154></TD><TD COL=2
ALIGN = TOP><PRE><FONT SIZE=7><B>Childnet International - Index</FONT></B></PRE><IMG
SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P><body>
<TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=0><TR>
<TD VALIGN=top ALIGN = RIGHT><IMG SRC="grey3.gif" WIDTH = 130 HEIGHT =15> <a href
="research.html"><IMG SRC="smbut1.gif" WIDTH = 28 HEIGHT=28 border =0></A></TD>
<TD COL=2><FONT SIZE=6>Research</A></FONT>
 our research
plans and developments elsewhere</TD></TR><TR>
<TD ALIGN =RIGHT> <FONT SIZE=6>Positive projects</FONT></A>
 linking
children around the globe. Contribute details of your own project </TD> <TD COL=2 VALIGN=top><a href
="projects.html"><IMG SRC="smbut2.gif" WIDTH = 28 HEIGHT= 28 border = 0></A><IMG
SRC="orgline.gif" WIDTH = 120 HEIGHT =15></TD></TR><TR>
<TD VALIGN=top ALIGN =RIGHT><IMG SRC="grey3.gif" WIDTH = 120 HEIGHT =15><a href
="practice.html"><IMG SRC="smbut3.gif" WIDTH = 28 HEIGHT= 28 border = 0></A></TD>
<TD COL=2><FONT SIZE=6>Promoting good practice in the industry</FONT>
</A>
 do you have an example of child-friendly actions by a computer or communications
company?</TD></TR><TR><TD ALIGN =RIGHT> <FONT SIZE=6> Educating and
informing</FONT> </A>
 helping parents, educators, governments and us all get to grips with children's
interests in international communications</TD><TD COL=2 VALIGN=top><IMG
SRC="smbut4.gif" WIDTH = 28 HEIGHT= 28 border =0></A><IMG SRC="orgline.gif" WIDTH = 140 HEIGHT
=15></TD></TR>TR><TD VALIGN=top ALIGN =RIGHT><IMG SRC="grey3.gif" WIDTH = 145 HEIGHT
=15><IMG SRC="smbut5.gif" WIDTH = 28 HEIGHT= 28 border = 0></A></TD> <TD
COL=2> <FONT SIZE=6>Who we are</FONT></A>
and how to reach
us</TD></TR><TR><TD ALIGN =RIGHT> <FONT SIZE=6>What's
new</FONT></A>
 new information on this site </TD><TD COL=2 VALIGN=top><a href
="new.html"><IMG SRC="smbut2.gif" WIDTH = 28 HEIGHT= 28 border = 0></A><IMG SRC="orgline.gif"
WIDTH = 120 HEIGHT =15></TD></TR></TABLE>

<hr>Related information on index ...
ATLASWWW
<a
href="NEW.html">NEW
SAFETY
<a
href="PRINT.html">PRINT
WHO
<a
href="NUALA.html">NUALA
NIGEL
<a
href="EDUCATION.html">EDUCATION<li>PRACTICE
<a
href="PROJECTS.html">PROJECTS
RESEARCH
<hr></body><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents
Page"><a href="HOME.html"><img src="previous_motif.gif" width=50 height=30 alt="Previous

```
Page"></a><a href="ATLASWWW.html"><img src="next_motif.gif" width=50 height=30 alt="Next
Page"></a><br>
Contents: <a href="contents.html">Table of Contents</a>Previous: <a href="HOME.html">HOME</a>
Next: <a href="ATLASWWW.html">ATLASWWW</a></br></br><hr>
```

Research page

```
<!-- HTML format of hypertext --><html><hr><a href="contents.html"><img src="contents_motif.gif"
width=50 height=30 alt="Contents Page"></a><a href="INDEX.html"><img src="previous_motif.gif"
width=50 height=30 alt="Previous Page"></a><br>Contents: <a href="contents.html">Table of
Contents</a>Previous: <a href="INDEX.html">INDEX</a></br></br><hr><title>research</title><TABLE
BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="smile.gif" WIDTH = 154 HEIGHT
= 154 LOWSRC="smile.gif" WIDTH = 154 HEIGHT = 154></TD><TD COL=2 ALIGN = TOP><PRE><FONT
SIZE=7><B>Research</FONT></B></PRE><IMG SRC="grey3.gif" WIDTH = 240 HEIGHT
=15></TD></TR></TABLE><P><body>

Childnet believes that international communications systems offer great potential to benefit children, and we
all need to understand better the impact that such systems can have.<P>We are seeking funding for a research
programme which will evaluate the positive (and negative influences) of the Internet and other international
communication systems on children.<P>Much debate is currently centred on protecting children from
undesirable content. Childnet has proposed that the various approaches suggested to help this (eg blocking and
filtering software, firewalls, self rating and third party rating ) do need to be independently evaluated. We are
seeking funding for a research study to be conducted jointly with Middlesex University Computer Science
Department to undertake such an evaluation.<P>The last year has seen the development of new tools to help
parents control access to the Net. Childnet commends such initiatives. Two wide ranging tools currently under
development are:<P><A HREF="http://www.w3.org/pub/WWW/PICS">PICS</A> - a system allowing various
rating systems to be accessed by the user. When loading a web page, the pre-selected rating system will be
searched in parallel, to give the rating for that page. This is being developed at the Massachussetts Institute of
Technology. <P><A HREF="http://www.rsac.org">RSACi</A> The Recreational Software Advisory Council (of
America) has to date been involved in rating computer games on four criteria (Violence, Nudity, Language and
General Suitability for children). They are currently seeking to apply their ratings to commonly accessed web
sites. These ratings could be used in conjunction with PICS and proprietary blocking software eg
Cyberpatrol.<P></B></TD></TR></TABLE><FONT SIZE =4>We are interested in learning of other research
initiatives <P></FONT><br><br></ul><hr></body><hr><a href="contents.html"><img
src="contents_motif.gif" width=50 height=30 alt="Contents Page"></a><a href="INDEX.html"><img
src="previous_motif.gif" width=50 height=30 alt="Previous Page"></a><br>Contents: <a
href="contents.html">Table of Contents</a>Previous: <a href="INDEX.html">INDEX</a></br></br><hr>
```

Projects page

```
<!-- HTML format of hypertext --><html><hr><a href="contents.html"><img src="contents_motif.gif"
width=50 height=30 alt="Contents Page"></a><a href="INDEX.html"><img src="previous_motif.gif"
width=50 height=30 alt="Previous Page"></a><br>Contents: <a href="contents.html">Table of
Contents</a>Previous: <a href="INDEX.html">INDEX</a></br></br><hr><title>positive
projects</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG
SRC="smile.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="smile.gif" WIDTH = 154 HEIGHT = 154></TD><TD
COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>Positive Projects</FONT></B></PRE><IMG
SRC="grey3.gif" WIDTH = 240 HEIGHT=15></TD></TR></TABLE><P>
<body>Childnet wishes to encourage positive educational and social contacts using communications systems,
between children from different countries. We will act as a facilitator, manager and help find funding for such
projects. <P>There are many advantages of using technology to link children in this way:<P>
        broadening horizons<BR>
        encouraging research<BR>
        sharing differences in background and language<BR>
        overcoming historic cultural prejudice and suspicion<BR>
        joint educational projects - learning together<BR>
        for children of migrants discovering more of their own cultural heritage<P>

The internet is cheap, fast, graphic and available. Childnet is particularly concerned to further that availability
by encouraging computer and communications companies to give some of their resource for children in
countries where technology is less available. <P>One example of a positive project is <A
HREF="http://www.kidlink.org/">Kidlink</A> which encourages global dialogue between children in the age
range 10-15. Another is Kids Space which publishes children's artwork, stories and encourages chat.<P>
We are establishing a directory of existing projects which link children in different countries. We will publish
summaries of projects on our web site. Your project can be included if:<P>
```

it is international (i.e. involves children in two or more countries)
it involves children and young people up to 18 years
it has a positive social or educational purpose<P>
Projects can be open to any child or limited to a particular age group, location(s), or club. Our aim is to encourage those organising projects to learn from one another.

<hr></body>
<hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: INDEX</br></br><hr>

Practice page

<!-- HTML format of hypertext --><html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: INDEX</br></br><hr><title>promoting good practice</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="smile.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="smile.gif" WIDTH = 154 HEIGHT = 154></TD><TD COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>Promoting Good Practice</FONT></B></PRE><IMG SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P>

<body>Existing communications systems are designed with adults in mind first. The Internet is still a relatively young method of "nation speaking unto nation".<P>Internet providers, software and hardware companies and telecommunications companies are now waking up to the opportunities for children to use systems.<P>In the UK a major project is underway backed by the Department for Education and Employment to evaluate how "superhighway" developments can benefit children's education. Many international companies including Microsoft, IBM and Apple are involved. See <A HREF="http://www.open.gov.uk/dfee/dfeehome.htm">DFEE</A> Web Page for details.<P>In the USA major companies have backed new developments like <A HREF="http://www.w3.org/pub/WWW/PICS">PICS</A> and <A HREF="http://www.rsac.org">RSACi</A> to offer tools to parents in controlling internet access for their children.<P>Childnet will be encouraging the industry to adopt positive initiatives and to prevent undesirable content being seen by children.

<hr></body>

<hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: INDEX</br></br><hr>

Education page

<!-- HTML format of hypertext --><html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: INDEX</br></br><hr><title>educating and informing</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="smile.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="smile.gif" WIDTH = 154 HEIGHT = 154></TD>

<TD COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>Educating And Informing</FONT></B></PRE><IMG SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P><body>Childnet is networking with child welfare and educational groups, governments and international agencies to provide information on how children can benefit from (and be protected in using) international communications systems like the Internet.

<P>We want to facilitate discussion among all the different interest groups involved internationally in these issues.<P>We are as a first step looking at encouraging contact between politicians with a concern for these issues. While there has been intense debate about regulating the Internet in the USA and some concern in other countries there has been little international discussion.<P>We will give further details of our work in this area as it develops.

<hr></body><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: INDEX</br></br><hr>

Who page

<!-- HTML format of hypertext --><html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page"><img src="next_motif.gif" width=50 height=30 alt="Next Page">
Contents: Table of ContentsPrevious: INDEXNext: NUALA</br></br><hr><title>who we are?</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="smile.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="smile.gif" WIDTH = 154 HEIGHT = 154></TD><TD COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>Who We Are?</FONT></B></PRE><IMG SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P><body>Childnet International is a non-profit organisation with its international headquarters in London, England. It is registered in the UK as a charity No 1053193.<P> Childnet International also operates in the USA as an independent non-profit organisation.<P> The Director of Childnet is Nigel Williams. A Cambridge University graduate with experience as a government official, the computer industry and the voluntary sector he has been a strong advocate of the interests of children, especially with reference to the impact of the media. <P>Nigel has researched and written on the legal framework affecting the content of media communications. Married to Heather he has four computer literate children. <P>Nigel has travelled widely especially in the USA, Europe and Asia.<P>
Nigel can be contacted at:
Childnet International,
35 Piccadilly,
LONDON, W1V 9PB
Voice: 0171 525 9014
Fax : 0171 701 1418
<IMG SRC="orgline.gif" WIDTH = 340 HEIGHT =15><P>The US Director of Childnet International is Nuala Holowicki. She received an honors degree in law from Queen's University Belfast and a Master of Laws fromWayne State University in Detroit. She was admitted to the State Bar of Michigan in 1995.<P>Nuala has wide legal experience including some child abuse cases. Her husband Gerry works in the computer industry.<P>
Nuala can be contacted at:
317 South Division, #198
Ann Arbor, MI 48104
USA<P>
Voice and fax: (313) 572 4595

<hr>Related information on who ...
NUALA
NIGEL
<hr></body>
<hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page"><img src="next_motif.gif" width=50 height=30 alt="Next Page">

Contents: Table of ContentsPrevious: INDEX
Next: NUALA</br></br><hr>

Nigel's home page

<!-- HTML format of hypertext --><html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: WHO</br></br><hr><title>nigel's home page</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="man.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="man.gif" WIDTH = 154 HEIGHT = 154></TD>
<TD COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>Nigel'S Home Page</FONT></B></PRE><IMG SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P><body>The Director of Childnet is Nigel Williams. A Cambridge University graduate with experience as a government official, the computer industry and the voluntary sector he has been a strong advocate of the interests of children, especially with reference to the impact of the media. <P>Nigel has researched and written on the legal framework affecting the content of media communications. Married to Heather he has four computer literate children.<P>
Nigel has travelled widely especially in the USA, Europe and Asia.<P>
Nigel can be contacted at:
Childnet International,
35 Piccadilly,
LONDON, W1V 9PB
Voice: 0171 525 9014
Fax : 0171 701 1418

<hr></body>
<hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: WHO</br></br><hr>

Nuala's home page

<!-- HTML format of hypertext --><html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: WHO</br></br><hr><title>nuala's home page</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="woman.gif" .WIDTH = 154 HEIGHT = 154 LOWSRC="woman.gif" WIDTH = 154 HEIGHT = 154></TD>
<TD COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>Nuala'S Home Page</FONT></B></PRE><IMG SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P><body>The US Director of Childnet International is Nuala Holowicki. She received an honors degree in law from Queen's University Belfast and a Master of Laws from Wayne State University in Detroit. She was admitted to the State Bar of Michigan in 1995.<P>Nuala has wide legal experience including some child abuse cases. Her husband Gerry works in the computer industry.<P>
Nuala can be contacted at:
317 South Division, #198
Ann Arbor, MI 48104
USA<P>
Voice and fax: (313) 572 4595

<hr></body>
<hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: WHO</br></br><hr>

New page

<!-- HTML format of hypertext --><html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page"><img src="next_motif.gif" width=50 height=30 alt="Next Page">
Contents: Table of ContentsPrevious: INDEXNext: SAFETY</br></br><hr><title>what's new</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="smile.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="smile.gif" WIDTH = 154 HEIGHT = 154></TD><TD COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>What'S New</FONT></B></PRE><IMG SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P><body><A HREF ="intro.html">CHILD SAFETY ON THE INTERNET</A><P></B>Summary of Video Conference held between Capitol Hill, Washington DC and Westminster, London on Wednesday 10 July 1996<P>

<hr>Related information on new ...
SAFETY
PRINT
<hr></body><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page"><img src="next_motif.gif" width=50 height=30 alt="Next Page">
Contents: Table of ContentsPrevious: INDEXNext: SAFETY</br></br><hr>

Safety page

<!-- HTML format of hypertext --><html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page"><img src="next_motif.gif" width=50 height=30 alt="Next Page">
Contents: Table of ContentsPrevious: NEWNext: PRINT</br></br><hr><title>child safety</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="smile.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="smile.gif" WIDTH = 154 HEIGHT = 154></TD>
<TD COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>Child Safety</FONT></B></PRE><IMG SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P><body><B>Childnet International arranged the <FONT COLOR ="#0515C9">first ever video conference between US and UK legislators</FONT COLOR> on Wednesday 10 July. The subject under discussion was how children could use the Internet safely, and be protected from undesirable content.</B><P> Topics covered in the conference included how further educational contact between children in different countries could be promoted, the extent to which material harmful to children (eg pornography) was currently available, possible protective measures including law enforcement, rating systems and blocking software and the need for further international cooperation.
<P></BLOCKQUOTE><IMG SRC="orgline.gif">
 <DD> <FONT COLOR ="#0515C9"><B>UK</B></FONT COLOR> participants were:<P><B>Baroness Brenda Dean</B> (Labour - Chair of the telephone watchdog ICSTIS><A HREF="http://www.icstis.org.uk">http://www.icstis.org.uk</A><P><B>Lord Renwick</B> (Con - member of the board of the National Council of Educational Technology) <A HREF="http://www.ncet/csv/warwick.ac.uk">http://www.ncet/csv/warwick.ac.uk</A><P><B>Stephen Timms

MP</B> (Labour - much experience in the computer industry) <P><B>Ann Winterton MP</B> (Con - who has raised concerns about children and computer pornography)<P><DD><FONT COLOR ="#0515C9"><B>US</B></FONT COLOR> participants were:<P><B> Senator Patrick Leahy</B> (Democrat)- a champion of freedom of speech known as the "cyber senator" and represents the State of Vermont.<P><B>Representatives Chris Cox</B> (Republican - Southern California) co-sponsor of a House bill to encourage the use of software controls to prevent pornography being seen by children.<P><B>Representative Rick White</B> (Republican - Washington State) - helped found the Internet Caucus in the House, represents the district which is home to Microsoft.<P>The conference was chaired by <B>Bridget Kendall</B>, BBC Washington Correspondent. <P><IMG SRC="grey3.gif"><P> Equipment and technical support was generously provided by
<B>Picturetel Ltd</B> and <B>Mercury Communications</B>.<P><IMG SRC="orgline.gif">
 A single screen version of the <A HREF ="print.html">report</A> for easy printing.</body></html>

<hr>Related information on safety ...
PRINT
<hr></body>
<hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page"><img src="next_motif.gif" width=50 height=30 alt="Next Page">

Contents: Table of ContentsPrevious: NEW Next: PRINT</br></br><hr>

AtlasWWW page

<!-- HTML format of hypertext --><html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: INDEX</br></br><hr><title>atlaswww's home page</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="crayon1.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="crayon1.gif" WIDTH = 154 HEIGHT = 154></TD><TD COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>Atlaswww'S Home Page</FONT></B></PRE><IMG SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P>
<body>this site was designed by Jake Tilson at
<DD> <A HREF="//www.ruskin-sch.ox.ac.uk/atlas">ATLASWWW Site Design & Planning Services</A><P></FONT>jake@tilson.demon.co.uk<P>
<FONT SIZE = 5><DD><A HREF="//www.ruskin-sch.ox.ac.uk/~jake/design/design3.html">other sites</A> designed by ATLASWWW<P><DD><A HREF="//www.ruskin-sch.ox.ac.uk/~jake/re/re1.html">ATLASWWW</A> Research Area<P></FONT><IMG SRC="crayon2.gif"><P>

<hr></body><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page">
<img src="previous_motif.gif" width=50 height=30 alt="Previous Page">

Contents: Table of ContentsPrevious: INDEX</br></br><hr>

Print page

<!-- HTML format of hypertext --><html><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: SAFETY</br></br><hr><title>child safety on the internet</title><TABLE BORDER = 0 CELLPADDING=10 CELLSPACING=10><TR><TD><IMG SRC="smile.gif" WIDTH = 154 HEIGHT = 154 LOWSRC="smile.gif" WIDTH = 154 HEIGHT = 154></TD>
<TD COL=2 ALIGN = TOP><PRE><FONT SIZE=7><B>Child Safety On The Internet</FONT></B></PRE>
<IMG SRC="grey3.gif" WIDTH = 240 HEIGHT =15></TD></TR></TABLE><P><body>
Summary of Video Conference held between Capitol Hill, Washington DC and Westminster, London on Wednesday 10 July 1996<P>
INTRODUCTION: IAN TAYLOR MBE MP, UK MINISTER FOR SCIENCE AND TECHNOLOGY<P>the Internet is very exciting and can do enormous good it provides access to information and libraries that were previously unavailable however there are also people who put on information which can be offensive and distasteful to people who come across it the purpose of today is to discuss the effects of this information on children voluntary proposals for self-restraint by industries is one solution caution against a leap into legislation as the Internet is designed to be resistant to government legislation and therefore legislation could be totally ineffective<P>

SENATOR PATRICK LEAHY
the Internet opens huge opportunities for people and we don't want to limit these the Internet has grown because of governments not trying to hinder it it can also damage children who have not yet formed their own judgement we have to figure out software that enables parents to block particular sites from their children the legislative route may be impossible technology might be the best way to regulate the Internet<P>

LORD RENWICK
the Internet has huge educational potential a pilot study of seven schools shows that it is a tremendous motivator in education teachers need to be trained and a networking system needs to be developed there is nothing but good in the opportunity to access a wide range of information as long as it is relevant to the national curriculum and it aids and not distracts the teacher<P>

ANN WINTERTON MP
there is a downside to the Internet we all realise that there is a problem with child pornography in the world no civilised society can sit back and not take action against it research carried out in the UK by Professor Thimbleby has shown that half of the uses of a common search engine were related to pornography we need to co-operate together to encourage and educate parents and make them aware of the problem<p>

CONGRESSMAN CHRIS COX
there is a lot on the World Wide Web that we don't want children to see and there will be more tomorrow there are two basic questions governments must confront in order to address the problem:
Will Internet standards be determined by what is acceptable for children?
If yes, does anyone derive any meaningful protection?
Whatever standards are set by government, they will be woefully inadequate for the protection of children we have to think of ways to utilise the technology so that the content will be suitable for the intended recipient<P>

SENATOR PATRICK LEAHY
we have to understand that government won't be able to set standards acceptable to all parents we've got to give parents a tool to make it their own responsibility we have to work with teachers no matter what we do there will still be objectionable material on the Internet<P>

ANN WINTERTON MP
there has been a disintegration of the family unit over the past two or three decades and therefore there are children out there who have access to the Internet and don't have parental guidance the Internet also provides an opportunity for children to meet and communicatewith others and this can include paedophiles who can pass themselves off as children the service provider should also take some responsibility as they like to be considered as large responsible companies governments should play a part in voluntary regulation<P>

CONGRESSMAN RICK WHITE
we are confronting the old problem of pornography in a different medium the fundamental problems are no different the Internet creates a problem but also provides us with the tools to solve the problem as you can program your computer to determine what you do and don't want to receive there is a role for some government regulation the heart of the problem is how do we deal with materials which are legally obscene and are on the Internet<P>

BARONESS DEAN
we are now looking at something very different as the Internet can be accessed in your own living room there is no simple answer, instead a menu of answers the government has a role the service providers themselves have a role; they could introduce a rating system parents have a role, but they need to know what is available and how they can block it there is an international consensus on what we don't want children to see i.e.

child pornography strict statutory regulation won't work we need to have an effective sanction with which to deal with services on the Internet which aren't acceptable the debate which we should be having is how we can still provide the opportunities the Internet offers whilst also protecting vulnerable children<P>

CONGRESSMAN CHRIS COX
the current models of regulation are inadequate in dealing with the Internet the notion that we are going to regulate service providers and make them responsible is a serious one content providers are also responsible if service providers are left to set up a screening system, it is going to be an imperfect screen service providers need incentives; putting them in jail won't helpscreening software is getting more and more sophisticated and is the key to the problem<P>

SENATOR PATRICK LEAHY
we should not feel helpless because we did not grow up with the Internet parents have a strong sense of responsibility and there are techniques which can help parents and teachers there are laws which can help throw child abusers on the Internet into jail we should all agree that child pornography is unacceptable and can be acted against<P>

STEPHEN TIMMS MP
technology is emerging to allow parents and others to control the Internet on their child's screen somebody rates the WWW screen and the software determines how much material gets through this is only part of the solution as there are problems with the software screening approach e.g.
how can we cope with the vast

amount of information being added to the Internet daily? can we depend on the publishers to rate the material themselves?
We should encourage the service providers to be responsible and encourage them to remove offensive material question to the US:'The Communications Decency Act was fiercely criticised. Why was the legislation introduced given the ferocity of the response?'<P>

SENATOR PATRICK LEAHY
I was in the small minority that voted against the legislation as was convinced that the courts would find it unconstitutional people want to protect children and therefore people voted for the Act if they voted against it, it would look as if they were in favour of child pornography and abuse<P>

CONGRESSMAN CHRIS COX
the bottom line is that once legislation gets to the floor on child pornography or problems with harm to minors, then no-one wants to vote against it good samaritan protection where people who are screening almost perfectly are protected against being sued, is very important<P>

CONGRESSMAN RICK WHITE
the Communications Decency Act missed the balance we will be able to write a law once we know how the constitution works there is a strong feeling in the public that there is a role for the government we have to make it consistent with constitution but also not damaging the Internet unnecessarily<P>

ANN WINTERTON MP
the Communications Decency Act was followed closely in Britain as it would have an international impactis semi-assured by the objectives and aims of the legislation the answer to the problem is to get the legislators, the service providers, and all people associated with the Internet together in conferences so they can discussthe problem together they could hammer out an acceptable policylegislation is not the only way forward service providers should not be fined straight away but instead reminded that child pornography is illegal and encouraged to be more responsible<P>

LORD RENWICK
it is definitely possible to build an international consensus on what the problem is and how we can solve it an incentive is necessary to both service providers and content providers this incentive could not only come from legal activity but also from technology the ability to screen software i.e. setting up exclusion files, works best in schools but parents can screen software aswell, removing undesirable content technology is going to come down on the side of control and good conduct<P>

BARONESS DEAN
this is what has been happening in Britain over the last 12 months on Internet pornography accessed by premium rate numbers self-regulation works there are arrangements between an independent committee and the network operator, and then the service provider has a contract with the network operator in 1995 on Internet services accessed by the premium rate number; there were very explicit sexual pictures on 25 services this was a breach of code and when asked, the service providers removed them co-operation does work and services can be taken off we must get that kind of consensus with the service providers and the content providers, on a voluntary basis<P>

SENATOR PATRICK LEAHY
is hesitant in introducing new legislation must wait until we know what the Supreme Court rules must see what the software companies come up with in allowing screening by parents, schools and servers the combination of the court ruling and what software is produced will determine the legislation<P>

CONGRESSMAN RICK WHITE
we have to wait to hear from the court and when we do we'll be back to the drawing board the technology will go much further than we imagine and the issues may solve themselves as we become more comfortable with the technology our goal is not to go too far too fast<P>

CONGRESSMAN CHRIS COX
regulations are stultifying development as people cannot move beyond the scope of regulation things change so rapidly that regulations slow down development<P>

FINAL QUESTION:'IS FURTHER INTERNATIONAL DISCUSSION USEFUL?'
<p>BARONESS DEAN
the ITU committee are discussing this later this year and there is a call in Europe for the EU to discuss it is absolutely essential to talk to one another and learn from each other in the international market and these kind of discussions should continue<P>

STEPHEN TIMMS MP
things move too fast to draw up a comprehensive international treaty it is important to work internationally as the Internet is by its very nature international a treaty is not necessarily the right approach a great deal of international discussion is needed<P>

SENATOR PATRICK LEAHY
discussions are very useful the industry itself will find itself working with governments in trying to do a lot of it voluntarily it is virtually impossible at this stage to form an international treaty which will reflect the needs and laws of every single country international co-operation will develop between the services as it becomes more widespread<P><blockquote>BRIEF SUMMARY: BRIDGET KENDALL, MODERATOR
everybody is concerned about the possibilities of pornography which could harm children there are reservations about restrictions the need not to go too far too fast is acknowledged we need to look at incentives and self-regulation and evolving a means of control people must continue to exchange information</blockquote>

CONCLUSION: NIGEL WILLIAMS, DIRECTOR, CHILDNET INTERNATIONAL
thanked all those who participated in unique event the Internet is international and there are children in every country of the world we need to develop the tremendous educational potential of the Internet while also providing safe access for children there are concerns and children do need protection there is common agreement that laws against child pornography need to be rigorously enforced on the Internet. The international community must come up with an effective consensus about how children can use the Internet safely there will be different opinions about the precise approach we have taken the first step in promoting international debate and we need to involve educators, parents and the communications industry in international discussions about protecting children on the Internet <hr>Back to summary.

<hr></body><hr><img src="contents_motif.gif" width=50 height=30 alt="Contents Page"><img src="previous_motif.gif" width=50 height=30 alt="Previous Page">
Contents: Table of ContentsPrevious: SAFETY</br></br><hr>

Appendix I

Generated trace file by CUM-DesTool

HD CMR:trace 19/4/96 18:16:02

****

activation of : BROWSING

BROWSING
State : TRUE
Excitation frequency : 1

BROWSING => STARTING

STARTING
State : TRUE
Excitation frequency : 1

STARTING => CLICK-OK

/CLICK-OK/
State : TRUE
Excitation frequency : 1

****

CLICK-OK has been performed and then deactivated

/CLICK-OK/
State : UNDETERMINED
Excitation frequency : 2

BROWSING=> FINDING-TOPIC

FINDING-TOPIC
State : TRUE
Excitation frequency : 1

FINDING-TOPIC=> LOOKING-FOR-TABLE-OF-CONTENTS

LOOKING-FOR-TABLE-OF-CONTENTS
State : TRUE
Excitation frequency : 1

LOOKING-FOR-TABLE-CONTENTS=>CLICK-TABLE-OF-CONTENTS

/CLICK-TABLE-OF-CONTENTS/
State : TRUE
Excitation frequency : 1

****

activation of: FINDING-TOPIC

FINDING-TOPIC
State : TRUE
Excitation frequency : 2

FINDING-TOPIC => USING-SEARCH-MECHANISM

USING-SEARCH-MECHANISM
State : TRUE
Excitation frequency : 1

USING-SEARCH-MECHANISM=> CLICK-FIND

/CLICK-FIND/
State : TRUE
Excitation frequency : 1

****

CLICK-FIND has been performed and then deactivated

/CLICK-FIND/
State : UNDETERMINED
Excitation frequency : 2

****

activation of : CLICK-FIND

/CLICK-FIND/
State : TRUE
Excitation frequency : 3

****

TYPE-QUERY has been performed and then deactivated

/TYPE-QUERY/
State : UNDETERMINED
Excitation frequency : 1

****

activation of: FINDING-TOPIC

FINDING-TOPIC
State : TRUE
Excitation frequency : 3

FINDING-TOPIC => USING-SEARCH-MECHANISM

USING-SEARCH-MECHANISM
State : TRUE
Excitation frequency : 2

USING-SEARCH-MECHANISM => CLICK-FIND

/CLICK-FIND/
State : TRUE
Excitation frequency : 4

****

CLICK-FIND has been performed and then deactivated

/CLICK-FIND/
State : UNDETERMINED
Excitation frequency : 5

****

The network is initialised

****

activation of: FINDING-TOPIC

FINDING-TOPIC
State : TRUE
Excitation frequency : 4

FINDING-TOPIC => USING-SEARCH-MECHANISM

USING-SEARCH-MECHANISM
State : TRUE
Excitation frequency : 3

USING-SEARCH-MECHANISM => CLICK-FIND

/CLICK-FIND/
State : TRUE
Excitation frequency : 6

****

CLICK-FIND has been performed and then deactivated

/CLICK-FIND/
State : UNDETERMINED
Excitation frequency : 7

FINDING-TOPIC=> LOOKING-FOR-TABLE-OF-CONTENTS

LOOKING-FOR-TABLE-OF-CONTENTS
State : TRUE
Excitation frequency : 2

LOOKING-FOR-TABLE-OF-CONTENTS=>CLICK-TABLE-OF-CONTENTS

/CLICK-TABLE-OF-CONTENTS/
State : UNDETERMINED
Excitation frequency : 2

****
The network is initialised
****
activation of : BROWSING

BROWSING
State : TRUE
Excitation frequency : 2

BROWSING => STARTING

STARTING
State : TRUE
Excitation frequency : 2

STARTING => CLICK-OK

/CLICK-OK/
State : TRUE
Excitation frequency : 3

****
CLICK-OK has been performed and then deactivated

/CLICK-OK/
State : UNDETERMINED
Excitation frequency : 4

BROWSING=> FINDING-TOPIC

FINDING-TOPIC
State : TRUE
Excitation frequency : 5

FINDING-TOPIC => USING-SEARCH-MECHANISM

USING-SEARCH-MECHANISM
State : TRUE
Excitation frequency : 4

USING-SEARCH-MECHANISM => CLICK-FIND

/CLICK-FIND/
State : TRUE
Excitation frequency : 8

****

CLICK-FIND has been performed and then deactivated

/CLICK-FIND/
State : UNDETERMINED
Excitation frequency : 9

FINDING-TOPIC=> LOOKING-FOR-TABLE-OF-CONTENTS

LOOKING-FOR-TABLE-OF-CONTENTS
State : TRUE
Excitation frequency : 3

LOOKING-FOR-TABLE-OF-CONTENTS=>CLICK-TABLE-OF-CONTENTS

/CLICK-TABLE-OF-CONTENTS/
State : UNDETERMINED
Excitation frequency : 4

Appendix J

Papers published

J1. **Theng, Y.L.** (1995), "Lost in hyperspace? - A look at 4 viable approaches," *HCI'95 Adjunct Proceedings*, pp. 180-181, U.K.

J2. **Theng, Y.L., Thimbleby, H. and Jones, M.** (1995), "Designer tools for hypertext authoring," *IEE Colloquium on "The Authoring and Application of Hypermedia-Based User Interfaces,"* Digest No: **95/202**, U.K.

J3. **Theng, Y.L., Thimbleby, H. and Jones, M.** (1995),"Reducing information overload: A comparative study of hypertext systems," *IEE Colloquium on "Information Overload,"* Digest No: **95/223**, U.K.

J4. **Theng, Y.L., Rigny, C., Thimbleby, H., and Jones, M.** (1996), "Cognitive task graphs and executable user models for better hypertext," *APCHI'96*, pp. 421 – 433, Singapore.

J5. **Theng, Y.L., Thimbleby, H. and Jones, M.** (1996), "'Lost in hyperspace': Psychological problem or bad design?", *APCHI'96*, pp. 387 – 396, Singapore.

J6. **Theng, Y.L., Rigny, C., Thimbleby, H. and Jones, M.** (1996), "Improved conceptual design for better hypertext," *HCI'96*, pp. 181 – 188, U.K.

J7. **Rigny, C., Theng, Y.L. and Thimbleby, H.** (1996), "Cognitive user models as design aids," *HCI'96*, pp. 139 – 149, U.K.

J8. **Thimbleby, H., Jones, M. and Theng, Y.L.** (1997), "Is 'lost in hyperspace' lost in controversy?" *Hypertext'97*, poster presentation, Southampton, U.K.

J9. **Theng, Y.L., Rigny, C., Thimbleby, H., and Jones, M.** (1997), "HyperAT:HCI and web authoring," *HCI'97*, pp. 359 – 378, U.K.

'Lost in hyperspace?' - A look at four viable approaches

Yin Leng Theng (yin2@mdx.ac.uk)
School of Computing Science
Middlesex University

Hypertext systems, structuring traditional text into non-sequential text, has provided us with a valuable mechanism and flexibility for non-linear reading and retrieval [Conklin 1987]. In hypertext, users not only benefit from the information they read but also from the richly, interconnected network of nodes and links present in hypertexts. Nodes are documents which can contain text, graphics, audio, video, as well as source code or other forms of data whereas links are the active references that allow the user to move to other parts of the database as required. Though hypertext systems promise freedom of reading and information retrieval, they are not without problems. Ironically, it is precisely this freedom that gives rise to many of these problems. One of the problems that seems inherent within hypertext systems is that users tend to lose their way in the maze of information within hypertext systems. This is commonly referred to as the 'lost in hyperspace' problem, which we will abbreviate as LIH. Much work has been done to address the LIH problem which involves the use of graphical browsers and query/search mechanisms [e.g., Conklin 1987, Simpson and McKnight 1990, etc.]. These examples represent a *reactive* approach, that is, doing remedial work to correct the deficiencies within the hypertext systems because they are poorly designed and built. What if we do things right from the start? What if solutions are being sought for an avoidable problem?

The subsequent discussion is primarily concerned with four viable approaches which we believe will help address the LIH problem in hypertexts:

- ensure that good hypertext design principles and guidelines are incorporated into the building of the hypertext systems in the first place;
- examine designer tools to create hypertext documents with a reduced tendency for the LIH problem to arise;
- define carefully the task and structure of hypertext systems; and
- employ AI techniques to optimise users' performance and behaviour.

Approach 1: Incorporate good hypertext design principles and guidelines

Just as early programmers lacked experience, so do hypertext authors. Brown conducted an assessment on the quality of hypertext systems designed by seventy student authors [Brown 1990]. The prevalent faults found were: poor design of the visual appearance of material; overuse of technology with lots of clever effects but no attempt made to re-design for the new medium; lack of a coherent overall structure and presentation style. In fact, this whole business of designing and producing hypertext systems is still a relatively new discipline. It is no wonder that some of the problems in hypertexts are due to poor designs caused by bad authorship. To test our hypothesis, we conducted an experiment with a group of four subjects to evaluate three hypertext systems: state-of-the-art Dorling Kindersley's *The Ultimate Human Body*, *The Way Things Work* and ACM's *Hypertext-on-Hypertext*. Two of the subjects were experienced hypertext users while the other two had little or no experience with hypertexts. They were asked to use each hypertext system to perform tasks that involved browsing, information search, seeking references and revision. After that, they were asked to complete a post-questionnaire commenting on how pleased they were with the hypertext systems in helping them complete the tasks successfully. Based on responses from the subjects and other research findings [e.g., Hardman and Sharratt, etc.], important design principles for hypertext authoring are: consistency in presentation; minimal mental load of users to remember objects, actions and codes; balance between the ease of learning and ease of use; and compatibility of structure of hypertext and users' tasks. Design guidelines should be considered for the following areas: information and screen design, dialogue input, navigation aids and on-line assistance. Graphical document browsers, maps, table of contents, index, references, metaphors, etc. were identified by subjects as essential and helpful navigation aids to ameliorate the LIH problem.

Approach 2 : Examine designer tools

Designing hypertexts is a complex process. A study by Wright [Wright 1989] highlighted some problems faced by hypertext authors: authors often discovered that links are created that go nowhere, or information exists that has not been linked into the text. Therefore, hypertext authors need support tools to assist them reduce the complexity of designing hypertext systems. Though present hypertext authoring tools (MacroMind Director, HyperCard, Toolbook, CD-i, etc.) greatly reduce the time needed for creating hypertexts, authors still find design options restricted by the tools themselves. We conducted a survey among a group of eight hypertext authors. In the questionnaire, they were asked to comment on how satisfied they were with present authoring tools, what were some of the problems encountered in authoring, and what essential and helpful features hypertext authoring tools should provide. Seven out of eight authors were only partially satisfied with

present authoring tools. Some of the problems faced by authors when using these tools were: limited/no integrity checking mechanisms, not able to analyse user interactions, poor graphics/colour/programming support, etc. For hypertext designer tools to be effective and useful, the authors highlighted support in these areas: easy creation of nodes and hypertext links; links with external programs and peripherals, facilities to keep track on what authors have done, toggle between author and user mode.

Approach 3 : Define the task and structure of the hypertext system

Many diverse hypertext systems have been designed and implemented to meet a variety of functions and needs: tutorials like Allinson and Hammond's *Learning Support Environment* (1988); access to large, encyclopaedic information sources like Dorling Kindersely's *The Ultimate Human Body* (1994), Frisse's *Hypertext Medical Handbook* (1988), and the *Oxford English Dictionary* (1988); support collaboration among a number of individuals like Engelbart's NLS/Augment (1963); and production of on-line manual like ZOG (1987). In order to define the task and structure of the hypertext system, hypertext authors need to do a user-centred system design, that is, start with the needs of the users. The needs of the users should dominate the design of the interface, and the needs of the interface should dominate the design of the rest of the hyypertext system, in particular the links and nodes in the hypertext database [Norman 1986]. If the hypertext author's view of the world does not concur with the user's, then the links that the author have created will be of no use to the user. Dan Diaper's Task Analysis for Knowledge Descriptions (TAKD), a well-established task analysis method [Diaper 1990], perhaps can be used to ensure that the tasks and their relationships between the tasks are accurately represented.

To ascertain whether a mismatch between user's mental model of the hypertext system and the actual structure of the hypertext system will lead to LIH problem, we asked seven subjects to evaluate the ACM's *Hypertext-on-Hypertext*. In the questionnaire, they were asked to indicate whether they were 'lost' while performing the tasks that involved browsing, information search, seeking references and revision. They also drew what they perceived the structure of the hypertext system was. We made two interesting observations pertaining to LIH: (1) if the user's mental model does not match the actual structure of the hypertext system, he experiences LIH regardless whether he is an experienced or novice user; (2) the more experienced the user, the less time he spends understanding the structure of the hypertext, and therefore, he experiences LIH. As for the performance of the user, the following were observed: (i) there is no significant difference in the performance of the user (novice or experienced) if he understands the structure of the hypertext system; (ii) there is a degradation in the performance if user is 'lost'.

Approach 4 : Use AI techniques

As hypertext becomes more sophisticated, the hypertext database also grows bigger and becomes more complex. This has inadvertently given rise to fresh problems confronting designers in design issues and affecting users in navigational behaviour. Therefore, we need more intelligent means to help users orientate and navigate within hypertext systems. Query search mechanisms and AI techniques have been suggested to deal with the problem of finding information in large hypertext documents [Conklin 1987, Woodhead 1991, Carlson 1989]. Woodhead sees AI and hypertext in a symbiotic relationship - what AI has to offer hypertext is automated strategies and heuristics with the means to make large and complex knowledge domains more tractable whereas what hypertext has to offer AI is a user-centred style of dialog and decision-making [Woodhead 1991]. Carlson suggests that the innate features of a hypertext system can augment traditional search techniques such as keyword, cross-referencing, string and pattern-matching [Carlson 1989]. To solve the LIH problem, we are in the process of re-designing the ACM's *Hypertext-on-Hypertext* using HyperCard for effective front-end and a logic query language like Prolog to provide the powerful back-end data extraction and manipulation. We hope to provide intelligent help supported by Prolog in these few areas: easy-to-use and intelligent search with an on-line thesarus to broaden and narrow a search; ensure extraction of all relevant links based on user needs and browsing pattern; automatically generate return paths of unidirectional links; and automatically generate a map showing the nodes and links in the hypertext database.

References

Brown, P.J., "Assessing the quality of hypertext documents," *Proceedings of the ECHT'90*, pp. 1-12, U.K., 1990.

Carlson, P.A., "Hypertext and Intelligent Interfaces for Text Retrieval," *The Society of Text : Hypertext, Hypermedia and the Social Construction of Information*, pp. 59-76, MIT Press, U.K., 1989.

Conklin, J., "Hypertext : An Introduction and Survey," *IEEE Computer*, pp. 17-41, Sep. 1987.

Diaper, D., "Analysing Focused Interview Data with Task Analysis for Knowledge Descriptions (TAKD)," *Human-Computer Interaction : INTERACT '90*, Elsevier Science, Holland, 1990.

Hardman, L. and Sharratt, B.S., "User-centred hypertext design : the application of HCI design principles and guidelines," *Hypertext : State of the Art*, Intellect Books, pp. 252-259, 1990.

Norman, D., "Cognitive Engineering," *User Centred System Design : New Perspectives on Human-Computer Interaction*, pp. 31-61, Lawrence Erlbaum Associates, London, 1986.

Simpson, A. and McKnight, C., "Navigation in Hypertext : structural cues and mental maps," *Hypertext : State of the Art*, Intellect Books, pp. 73-83, 1990.

Woodhead, N., *Hypertext and Hypermedia : Theory and Applications*, Sigma Press, U.K., 1991.

Wright, P., "Interface Alternatives for Hypertexts," *Hypermedia*, Vol. **1(2)**, 1989.

DESIGNER TOOLS FOR HYPERTEXT AUTHORING

Y L Theng, M Jones and H W Thimbleby

Key Words : Hypertext, Designer tools, Authoring, 'Lost in hyperspace'

1.0 Introduction

Just as early programmers lacked experience, so do hypertext authors [Brown 1990]. In fact, this whole business of designing and producing hypertext systems is still a relatively new discipline. Creating hypertexts is complex because of the richness of interconnectivity that exists among nodes and links in hypertexts. As such, the demands placed on hypertext authors in authoring hypertexts cannot be underestimated. Hypertext authors have to perform many balancing acts: (1) ensure that the design and structure of the hypertext system is 'best' according to its function; (2) ensure that all the nodes and links created in the hypertext database correspond directly to the windows and links in the display screen so that there should be no redundant/missing links or nodes; and (3) incorporate good design guidelines for screen and information display, dialogue design, navigation aids and on-line assistance.

Hypertext authors don't make good hypermedia documents because it is difficult to do so. They are faced with a vast range of potential structures and an astronomically large number of choices when creating a hypertext document [Thimbleby 1995]. Just as users are often 'lost' while navigating in hypertexts, so are hypertext authors themselves in authoring hypertext systems!

The question we want to ask is: could it be possible that because hypertext authors themselves are 'lost' in the process of designing and authoring hypertext systems, they inadvertently contribute to poorly designed hypertext systems, which in turn leads to users often being 'lost in hyperspace'? If so, there are many new answers for the following question: How can authors then be helped in designing well-structured hypertext system?

2.0 Essential and helpful features for hypertext authoring tools

To investigate hypertext authors' expectations of what good hypertext designer tools should provide, we conducted a survey among a group of eight hypertext authors. Four of them were amateur authors and the other four were experienced authors. In the questionnaire, we asked the authors to rate how satisfied they were with present hypertext authoring tools. Of the eight surveyed, seven indicated they were only partially satisfied with what present hypertext systems can offer. Many problems encountered in authoring were highlighted, but the main problems can be attributed to poor support in: (i) integrity checking of nodes and links; (ii) capturing user requirements; (iii) analysing user interactions; and (iv) importing facilities.

Based on responses obtained from the questionnaire and research findings made by other researchers [e.g., Shneiderman and Kearsley 1989, Wright 1989, Conklin 1987, etc.], the essential and helpful features designer tools should provide fall under the following functional areas (in order of importance):

- *Creation of nodes and links*

  To help hypertext authors cope with the vast number of nodes and hypertext links, tools should make it easy for authors to create and label nodes and links so that designers can concentrate on design guidelines and principles. Coming up with good node and link names can be demanding for hypertext authors so Conklin suggests having an authoring tool that supports immediate recording of the idea but deferring the creation and labelling of the link/node until after thought is captured [Conklin 1987]. Authoring tools should support the range of interface design options that are helpful in hypertexts. There should be functions for node and link listing, node information and tools for annotating nodes. There should be provided an indicator to show where specific links go to and automatic connecting of certain terms. Tools should be provided for link checking. An automatic generation of return paths of unidirectional links free authors of the 'burden' of creating return links.

- *Generation of overall map and structure*

  To aid users in navigating hypertext, authoring tools should automatically generate maps, contents, bookmarks, etc. in hypertext systems.

- *Support for capturing and representing users' needs*

  Tools could be incorporated with "cognitive user models" that can help designers capture and represent users' needs more accurately during the design and development of hypertext systems. There should also be facilities to check user entry and traversal. This can be useful information to make the hypertext system more adaptive to users' changing needs.

- *Provision of a full range of editing facilities*

  Like all other electronic tools, a comprehensive range of editing facilities such as copying, moving, insertion, deletion, formatting, etc. should be provided for users. To ensure that the hypertext systems are not just electronic page-turners, a rich library of graphics and colours should be available. Facilities should also be provided to import/export text and/or graphics files without much hassle.

- *Support for tracking and checking*

  As the hypertext database grows in size, it will be difficult for authors to keep track of the nodes and links. It will be a tremendous help to authors if the tools are able to keep track of what the authors have done so far and capture the hypertext database with its nodes and links in a graphical form with the choice of viewing it on the screen or be printed out.

- *Support for testing and evaluation*

  To test and evaluate hypertext systems, there should be a means of recording users' use of them and their feedback in an analytical way. During the authoring process, a very essential feature an authoring tool should have is the capability to toggle between author and user mode so that authors can test ideas.

- *Creation of external links*

  Tools should allow links to be established with external facilities with ease. This will encourage and enrich associations of links outside the hypertext system.

3.0 Better support tools for authoring

A study by Wright [Wright 1989] highlighted some problems faced by hypertext authors during the development of hypertexts: authors often discovered that links are created that go nowhere, or information exists that has not been linked into the text and so is completely inaccessible to the reader. The table below compares some popular hypertext authoring tools (e.g. MacroMind Director, HyperCard, SuperCard and Toolbook) based on the *functional features* discussed in Section 2.0. It also highlights whether these functional features are implementable by vendors and users of the tools.

Table: Functional support provided by hypertext authoring tools and implications for implementation
by vendors and users of the tools. This table is provided to promote debate (the higher the number, the better).

Functional support	Hypertext authoring tools											
	MacroMind Director			HyperCard			SuperCard			Toolbook		
	FA	IV	IU	FA	IV	IU	FA	IV	IU	FA	IV	IU
Creation of nodes and links	1	2	3	1	2	3	1	2	3	1	2	3
Generation of map & structure	0	1	1	0	1	1	0	1	1	0	1	1
Capturing and representing users' needs	0	2	1	0	2	1	0	2	1	0	2	1
Full range of editing facilities	2	3	3	1	3	3	2	3	3	2	3	3
Support for tracking and checking	0	2	1	0	2	1	0	2	1	0	2	1
Support of testing and evaluation	0	2	1	0	2	1	0	2	1	0	2	1
Creation of external links	2	2	2	2	2	2	2	2	2	2	2	2

Key
FA: Feature available (2: yes; 0: no; 1: partly)
IV: For vendors to modify the current authoring tool (3: easy; 1: hard; 2: yes; 0: no)
IU: For users to implement using the current authoring tool (3: easy; 1: hard; 2: yes; 0: no)

As reflected in the table above, popular authoring systems are not providing the essential functional support yearned by hypertext authors to aid them build effective and efficient hypertext systems. Hypertext authors need better support tools than present hypertext systems can offer [Thimbleby 1995]. With good support tools, the complexity of hypertext authoring can be reduced, thus freeing the hypertext authors to concentrate on the design and structure of the hypertext systems.

4.0 Hypermedia style generators

Thimbleby describes in his paper a new style authoring tool, called *hypermedia style generators*, that writes hypermedia to a set-formula [Thimbleby 1995]. This sort of style generator enables the structure of the hypertext system be maintained consistently regardless of changes made due to author's modification, users' requirements, etc. Gentml, his demonstration authoring tool, makes the task of author's design problem much easier by retaining a guaranteed structure. It has the following characteristics:

- it imposes a customisable, but uniform visual style on any size document
- it imposes a clear and uniform logical style
- it supports the author's understanding of the document
- it supports hypermedia research

Gentml has been applied to different documents with success [Thimbleby 1995]. This tool provides authors with a guaranteed flexibility to create well-structured hypertext document.

5.0 Concluding remarks

This paper summarises results on essential and helpful features hypertext authoring tools should have to provide better support and automated help to hypertext authors. It also discusses a successful implementation of a new hypermedia design tool that makes the complex authoring process simpler and guarantees generation of well-structured hypertext documents.

References

Brown, P.J., "Assessing the quality of hypertext documents," *Proceedings of ECHT'90*, pp. 1-12, U.K., 1990.

Conklin, J., "Hypertext : An Introduction and Survey," *IEEE Computer*, pp. 17-41, Sep. 1987.

Shneiderman, B. and Kearsley, G., *Hypertext Hands-On! An Introduction to a New Way of Organising and Accessing Information*, Addison-Wesley, U.S.A., 1989.

Thimbleby, H., "Authoring consistent hypermedia without getting lost," *HCI'95 Adjunct Proceedings*, U.K., 1995.

Wright, P., "Interface Alternatives for Hypertexts," *Hypermedia*, Vol. 1(2), 1989.

REDUCING INFORMATION OVERLOAD: A COMPARATIVE STUDY OF HYPERTEXT SYSTEMS

Y L Theng, M Jones and H W Thimbleby

Instructions on how to read this paper

To illustrate the essence of hypertext, we have structured this paper as a hyperdocument. For readers who want to get a gist of what is covered quickly, follow the appropriate 'jumps' in the form of a 'Go to Para #' tag to the relevant paragraph. See also footnote '[Go to [1]Footnote]'. However, readers can also read this paper linearly.

Introduction

1. When Vannevar Bush envisioned his hypertext 'memex' in 1945 [Bush 1945], he dreamed of a personal system that would help him tackle the problem of information overload at that time. Today, with the advent of more and more powerful computers, we are still struggling to cope with the vast amount of information that bombards us daily. Perhaps hypertext technology, structuring traditional text into non-sequential text, is a viable solution to help us cope with the problem of information overload? *[Go to Para 3].* In hypertext systems, we are provided with a powerful, flexible mechanism to browse non-sequentially via a network of richly connected nodes and links, and extract only that part of the information that is relevant to us. The same usability issues arise in multimedia, hypermedia and World Wide Web, and we will call them all "hypertexts" for conciseness. *[Go to Para 2].*

2. Hypertext systems are fast becoming more complex than they used to be. Take the World Wide Web, for instance. It is a massive hypertext system that runs the danger of having too much information. Though hypertext technology may help us overcome the problem of information overload: it is information itself that can contribute to worse overload. *[Go to Para 3].*

3. Many hypertext systems are, however, poorly designed and built in terms of how information is structured and displayed. *[Go to Para 4].* When Brown conducted an assessment on the quality of hypertext systems designed by seventy student authors [Brown 1990], poor design of the visual appearance of material, overuse of technology with lots of clever effects but no attempt made to re-design for the new medium, lack of a coherent overall structure and presentation style were some prevalent faults identified. Just as early programmers lacked experience, so do hypertext authors. It is, therefore, not surprising that some of the problems in hypertexts are due to poor designs caused by bad authorship. To overcome this problem, hypertext authors need help [Thimbleby 1995]. They need to know what good design principles and guidelines are, so that they can incorporate them into the building of hypertext systems in the first place [Nielsen 1995, Signore 1995, Theng 1995]. Ironically, hypertext systems in trying to tackle the problem of information overload often confuse users further. *[Go to Para 4].*

4. Perhaps we need to re-examine the way hypertext systems are designed and built. Can we reduce information overload within hypertext systems by ensuring that good design guidelines and principles are incorporated into the building of hypertext systems in the first place? If so, then what are these guidelines and principles? (Other approaches to cope with information overload involve filtering

and retrieving relevant information [Belkin and Croft 1992], which we will not elaborate in this paper.) *[Go to Para 5].*

Hypertext design guidelines and principles to deal with information overload: A survey

5. The need for good design guidelines and principles has spurred us to do a survey of other research efforts done in this *[Go to [1]Footnote]* area for general interactive systems and in particular, hypertext systems. Much of the work done so far concentrated only on the interface design issues. We think that design guidelines and principles go beyond just providing an attractive interface. We have, therefore, classified design guidelines and principles into two broad areas that are concerned with interface issues *[Go to Para 5.1]* and structural interaction issues *[Go to Para 5.2].*

5.1 ***Interface Issues***

By 'interface issues', we refer to the information channel that allows the hypertext system to explain the internal structure and representation of nodes and links to user in the simplest and most effective way, and for the user to communicate his intentions and obtain the answer to his intentions. According to Hardman and Sharratt [Hardman and Sharratt 1990], the interface design principles applicable to hypertext authoring are to ensure: (i) consistency of presentation; and (ii) minimal mental overload for users to remember objects, actions and codes to navigate the hypertext system. Other researchers in their independent research work [e.g., Fillion and Boyle 1991, McKnight, Dillon and Richardson 1991, Hammond and Allinson 1989, Conklin 1987, etc.] identified interface design principles and guidelines into the following areas:

- *Navigation aids.* At any point in the hypertext, there should be sufficient information to orientate users such as the use of title, subtitle, page numbers, etc. Conklin suggests using graphical document browsers to display the structure of the document and allow users to assess what is there [Conklin 1987]. This is because much of the problem in navigation and in particular, disorientation is the lack of understanding of the document structure and the inability of users to assess the amount and size of information available. An experiment by Hammond and Allinson confirms that users' lack of overview information or wrong understanding of the hypertext structure make them more prone to losing their orientation [Hammond and Allinson 1989]. This suggests that in order to have more 'reader-friendly' hypertext systems, care should be taken to ensure that users do not feel 'lost'. Apart from browsers, maps of various kinds and functions have been used in many hypertext systems such as Intermedia [Conklin 1987, McKnight, Dillon and Richardson 1991]. 'Global maps' or 'overview diagrams' have been introduced to give users an overview of the hypertext system and the documents and links between them. 'Local maps' or 'document finders' are used to provide users with information on what documents are linked to current user-defined document. These maps aim to provide users with as much context information as possible so that users are able to answer questions like : 'Where am I?'; 'Where do I go from here?'; 'How much information is there?'; 'How did I arrive here?', etc.

  Fillion and Boyle suggest providing users with reference points, orientation cues and some notion of direction. Other information like the structure of the document, size of the document and the presentation of the nodes and the way they respond to given functions may have great impact on the way users orient themselves [Fillion and Boyle 1991]. Therefore table of

[1]Footnote: If you don't know about what "this" refers to, you haven't been following links properly!

contents, index and references have also been included in many hypertext systems like the *Semistructured Intelligent Navigation System* [Boyle and Snell 1990], *Book Emulator* [Benest 1990] and ACM's *Hypertext-on-Hypertext* [ACM 1989]. Metaphors of all kinds have been introduced in hypertext systems such as the travel metaphor in the *Learning Support System* [Allinson and Hammond 1989] and the book metaphor *Book Emulator* [Benest 1990]. To address the problem of returning to known locations, Monk suggests a 'personal browser' to capture information of previously visited node [Monk 1990]. Others like HyperCard provides a built-in Recent Command, a backtracking facility, to provide users with a list of miniature screen dumps of the 42 most recently visited nodes. In the design of *Hypergate*, Bernestein describes the introduction of personalised features like bookmarks and margin notes for users [Bernestein 1988].

- *Screen and information display.* The display of information should be kept short and simple with only the necessary information displayed. Information should also be grouped to reflect relationships and ordered to capture the hierarchical structure. Critical information should be highlighted though this should be used with discretion and only for a small proportion of information on the screen. As the links within hypertexts can be of different types, these should be made apparent to users before they are actioned. Visual coding for different types of information in the form of graphics, colour, brightness, flashing or combinations of them can be used. Care should be taken however not to use too many codes to prevent overloading the user. There should be consistency in the use of terminology, wording and format. Graphics can be used to illustrate ideas in a more interesting and appealing manner compared to just plain text.

- *Dialogue design.* In asking for user action or input, the position of the link on the screen should be obvious, for example, highlighting the link(s). But with a large number of links, this may clutter up the screen and an alternative is to define areas containing links instead.

- *On-line assistance.* Help should always be available to the user at the general level, for example, displaying a help icon on the screen. Help should also be available specific to the user's current position.

- *Fun to use.* Carroll's 1982 paper "The adventure of getting to know a computer" [Carroll 1982] advocates the game metaphor to present users with an exploratory environment and turn obstacles such as disorientation into challenges. Perhaps we can exploit the game metaphor in making hypertext systems easier to learn and more fun to use as well.

[Go to Para 5.2].

5.2 *Structural Interaction Issues*

Because the very essence of hypertext system lies in the associative relationships between nodes and links, it is vital that they are correctly captured and represented. Very often, designers of hypertext systems consider hypertext design to be a "creative" task and invest much effort in making it look good "technically" without much careful consideration given to the organisation of information and the modelling of associations of nodes and links [Signore 1995]. It is still open to debate, but we will define 'structural interaction issues' to refer to the formal methods of ensuring that the structural design of the hypertext system (in the form of computer programs) [Thimbleby 1990]: (i) is 'best' according to its function; (ii) matches user mental models; and (iii) is simple with suitable modularising of documents into nodes and appropriate links connecting the nodes.

Not only should designers be concerned with the structure of hypertext systems, they should pay attention to the type of nodes and links they are creating. Nodes are documents which contain text, graphics, audio, video, as well as source code or other forms of data and they can be typed, semistructured or composite. Most users of hypertext favour using nodes which express a single concept or idea [Conklin 1987]. Links are active references that allow the user to move to other parts of the hypertext databases and can be of many types: explicit (hard-coded links) or implicit (keyword links or links that emulate the human mind's associative mechanism); hierarchical (organisational links) or non-hierarchical (referential links) [Conklin 1987, Signore 1995]. Besides ensuring compatibility of the structure of hypertext system and users' tasks, Hardman and Sharratt also highlighted that it is important that hypertext systems should be adaptable to the different types and levels of users and adaptive to their browsing patterns [Hardman and Sharratt 1990]. This means that users should be made aware of promising links that exist among the various nodes. This can perhaps be achieved by examining the degree of affinity between nodes [Signore 1995].
[Go to Para 6].

Experiments & results

6. To investigate whether the above design guidelines and principles *[Go to Para 5]* are congruent with users' expectations and needs, we conducted an experiment with a group of four subjects to evaluate three hypertext systems: the much-publicised, state-of-art Dorling Kindersley's *The Ultimate Human Body* (1994), *The Way Things Work* (1994), and ACM's *Hypertext-on-Hypertext* (1987). Two of the subjects had little or no experience with hypertext systems while the other two were experienced hypertext users. They were asked to use each hypertext system by answering questions that involved tasks like browsing, seeking references, information search and revision. After that, they were asked to complete a questionnaire commenting on how satisfied they were with the design and structure of the system in helping them complete the tasks successfully. *[Go to Para 7].*

7. Of the three systems *[Go to Para 6]* evaluated, three of the four subjects found *The Way Things Work* to be the most effective in helping them complete the tasks. Not surprisingly, it was also picked as the one that gave these three subjects the most satisfaction. They commented that they preferred hypertext system with interface that is simple and easy to use. It would be added bonus if the hypertext system is interesting and fun with clever use of colours, sound and graphics. However, one of the subjects rated *The Way Things Work* the worst because the interaction and interface elements did not fit the perception of the tasks he had to perform. Though the three systems were not too poorly rated, subjects generally felt that they could be improved by incorporating the following features: (i) history list to trace the paths taken; (ii) guide to use the system; (iii) map to show where subject is, where he has been and where he can go; (iv) better help facility; and (v) intelligent recommendation facility/context-sensitive index. *[Go to Para 8].*

8. In another experiment, a group of four hypertext authors (two of whom were involved in the first experiment) were asked to complete a questionnaire what they think (in detail) are the three most important qualities effective hypertext systems should possess. From the responses obtained, authors' expectations of what an effective hypertext system should be do not differ from users' expectations. The following were listed as important qualities effective hypertext systems should possess: (i) ease of use; (ii) fun to use; (iii) good tools to navigate; (iv) information structured and organised into nodes in the most effective manner; (v) effective linking of information/nodes; and (vi) consistency in the use of icons and metaphors. *[Go to Para 9].*

9. The question we now want to ask is: why then are hypertext authors not able to design systems that they want, or users need? Can it be the case of 'they do not want to do the things they do, but they do it because they have no choice or help'? *[Go to Para 10].*

Concluding remarks and future work

10. In this paper, we have highlighted good design guidelines and principles *[Go to Para 5]* that should be incorporated into hypertext systems in order to reduce information overload. Though not statistically significant, the results from the two experiments *[Go to Para 6]* were suggestive in confirming that good design guidelines and principles are similar both from the users' and designers' perspectives. Besides using good design guidelines and principles, hypertext authors should ensure that users' needs are accurately captured and represented. Currently, we are looking into building user models for common tasks such as browsing, information search, seeking references and revision to help hypertext authors in the design and development of hypertext systems. This work is currently being carried out by Cécile Rigny and Yin Leng Theng at Middlesex University.

References

ACM, "ACM Hypertext-on-Hypertext," *ACM Press Database and Electronic Products Series*, 1989.

Allinson, L. and Hammond, N., "A learning support environment: The Hitchhikers' Guide," *Hypertext: Theory into practice*, Intellect Books, pp. 62-74, 1989.

Belkin, N.J., and Croft, W.B., "Information filtering and information retrieval: Two sides of the same coin?" *Communications of the ACM* **35**, pp. 29-38, 1992.

Benest, I., "A hypertext system with controlled hype," *Hypertext: State of the Art*, Intellect Books,
pp. 52-63, 1990.

Bernestein, M., "The bookmark and the compass: Orientation tools for hypertext users," *ACM SIGOIS Bulletin*, Vol. 9(4), pp. 34-45, 1988.

Boyle, C. and Snell, J., "Intelligent navigation for semistructured hypertext documents," *Hypertext: State of the Art*, Intellect Books, pp. 34-45, 1988.

Brown, P.J., "Assessing the quality of hypertext documents," *Proceedings of ECHT'90*, pp. 1-12, U.K., 1990.

Bush, V., "As We May Think," *Atlantic Monthly*, Vol. 7, pp. 101-108, 1945.

Carroll, J.M., "The adventure of getting to know a computer," *IEEE Computer*, Vol. **15**, No. **11**, pp. 1982.

Conklin, J., "Hypertext : An Introduction and Survey," *IEEE Computer*, pp. 17-41, Sep. 1987.

Fillion, F.M. and Boyle, C.D.B., "Important issues in hypertext documentation usability," *Proceedings of the 9th ACM Annual International Conference on Systems Documentation*,
SIGDOC'91, U.S.A., 1991.

Hammond, N. and Allinson, L., "Extending hypertext for learning: An investigation of access and
guidance tools," *Proceedings of People and Computers V*, pp. 293-304.

Hardman, L. and Sharratt, B.S., "User-centred hypertext design : the application of HCI design
principles and guidelines," *Hypertext : State of the Art*, Intellect Books, pp. 252-259, 1990.

McKnight, C., Dillon, A. and Richardson, J., *Hypertext in Context*, pp. 65-104, Cambridge University Press, U.K., 1991.

Monk, A., "Getting to known locations in hypertext," *Hypertext: State of the Art*, Intellect Books,
pp. 20-27, 1990.

Nielsen, J., *Multimedia and Hypertext: The Internet and Beyond*, AP Professional, U.S.A., 1995.

Signore, O., "Issues on Hypertext Design," *DEXA'95 Conference Proceedings*, London, 1995.

Thimbleby, H., *User Interface Design*, ACM Press, U.S.A., 1990.

Thimbleby, H., "Authoring consistent hypermedia without getting lost," *HCI'95 Adjunct Proceedings*, U.K., 1995.

Theng, Y.L., "Lost in hyperspace? A look at 4 viable approaches," *HCI'95 Adjunct Proceedings*, U.K., 1995.

Cognitive task graphs and executable user models for better hypertext

Yin Leng Theng, Cécile Rigny, Harold Thimbleby & Matthew Jones
School of Computing Science, Middlesex University
Bounds Green Road,London N11 2NQ
Tel: +44 181 362 5000
Email: {y.theng, c.rigny, h.thimbleby, m.jones}@mdx.ac.uk

Abstract
Authoring hypertext systems is complex and difficult. To help designers build quality hypertext systems, designers should understand users' behaviour and the common tasks they perform when navigating hypertext systems, and, further, designers should build in accurate representations of user models. This is uncontentious wisdom, but so far, not easy. This paper shows that common tasks can be broken down into simpler subtasks, taking into consideration users' reasoning and learning. The results of this cognitive task-based approach can then be used to design the interface and functionality of a hypertext system. In particular we show that task trees can be automated, leading to 'executable user models.' Executable user models can be used efficiently as substitute users to support design and evaluation purposes. We discuss the task browsing as a concrete example.

Keywords
Task graph, hypertext, browsing, information search, seeking references, recall, navigation, cognitive user model

1. Introduction

Building effective and well-structured hypertext systems is a difficult and complex process — due to the richness of associative links that exist among nodes, and to the combinatorial explosion of design possibilities. Designers have to cope with a vast range of potential structures to build a hypertext system, and with a large number of choices to create links. When designers are themselves 'lost' in how to manage the complexity of the design process, producing well-structured hypertext systems becomes a matter of chance. It is, therefore, not surprising that few good hypertext systems are created. This, of course, has inadvertently generated a set of problems associated with navigation issues within hypertext systems, of which the 'lost in hyperspace' problem is but one example. How can designers be helped in creating well-structured, user-centred hypertext systems, appropriate to the users' tasks?

2. Need to understand users and the tasks they perform

To help designers build well-structured hypertext systems, we need to know users and their needs by analysing the common tasks they want to perform or try to perform. In fact, the importance of understanding users and their tasks has never been in question, at least in the research community: ranging from Hansen's "know the user" (Hansen, 1971), to Lewis and Rieman's "task-centred design" (Lewis and Rieman, 1995). By trying to make sense of what users should do or what they actually do, designers should stand a better chance of producing user-centred hypertext systems that will meet users' needs more effectively. However, it is also well-known that designers often design for themselves unless they are trained to realise that people are diverse, and that users are unlikely to be like them (Thimbleby, 1990; Landauer, 1995). It is also imperative that designers build into hypertext systems an accurate enough representation of user models when performing these tasks, taking into consideration users' reasoning and learning processes (Newman and Lamming, 1995).

Unfortunately the conventional exhortations do not help make much impact on hypertext design problems. The hypertext design space is astronomically large (Thimbleby, 1995). Iterative design, that usually helps improve systems, has problems in that users involved in the process experience too little of a system to help make significant design contributions, and, secondly, once most hypertext systems have reached the point of user testing, commercial pressures would have them shipped and sold to uncritical customers.

In this paper, we present ways of understanding users' behaviour and navigation issues in hypertext by giving them more structure, and using this structure for improved hypertext design, thus ameliorating the usability problems more effectively.

3. Browsing in hypertext

To develop an accurate understanding of users, it is essential that representative tasks which provide reasonably complete coverage of a system's intended functionality are chosen (Lewis and Rieman, 1995). Therefore, to identify the representative tasks in hypertext, we considered the kinds of support hypertext systems generally provide: applications support (e.g., tutorial and educational needs); support for collaboration among a number of individuals and production of on-line manuals; functional support (e.g., browsing and authoring); and cognitive support (e.g., reading, annotating, collaborating and learning). Based on the kinds of support provided by hypertext systems, we identified four representative tasks users want to perform (or try to perform) when using hypertext. These tasks are browsing, information search, seeking references, and recall.

As an illustration, we have selected one out of these four: the task we call *browsing* to illustrate how cognitive task graphs can be built. (Cognitive task graphs for the other tasks can be developed similarly.) Generally, browsing refers to reading and navigating in hypertext without a definite or explicit goal for the user to accomplish. However, in this paper, we include focussed browsing, where users have an idea of what they want to do to accomplish a goal. They may explore the system's interface to discover actions useful in helping them accomplish their current goal. Based on a match between what they are trying to do and the interface's description of actions, users then select actions they think will help them accomplish their current goal. In deciding what action to do

next, users will then try to understand system responses based on the action they have just performed.

In the following sections, we describe how the task *browsing* in hypertext is broken down into simpler, unit tasks using cognitive task graphs. We explain the rationale behind this task analysis approach. We then describe how cognitive task graphs are validated and refined by firstly, conducting video protocols with real users, and secondly, using computerised users in the form of executable user models. The results of our analysis are then used to make principled recommendations for the design of a prototype hypertext system.

4. Building a cognitive task graph

The concept of a task is central to user-centred design. Many task analysis (TA) techniques have been developed that focus on different aspects of tasks, such as the ease of learning a task, the knowledge users require in order to accomplish a task, or the task structure. These different aspects generally refer to the cognition, practice or logic of the task (Payne and Green, 1989). Therefore, to break the task *browsing* into unit tasks, we need to represent it in terms of how it is usually done (practice), the flow of actions (logic) and the cognitive processes (cognition).

To gather inputs for building the cognitive task graph for *browsing*, we performed the following activities: started with a rough cut of the task graph for *browsing* based on known facts about *browsing* and our experience with hypertext systems; and conducted video protocols to observe two "typical" users browsing a hypertext system. By "typical," we refer to a user who has general knowledge about navigation with hypertext systems.

Because we wanted to model the practice and logic aspects of the task *browsing*, we adapted the task decomposition (TD) technique to break down the task *browsing* into subtasks users need to have in order to perform browsing, and then describe *browsing* in terms of a flowchart of actions to achieve goals. Goals are the desired state(s) of the system and actions are the lowest level units of behaviour. Figure 1 shows the task graph for *browsing* and the possible subtasks that are associated to it: (1) starting; (2) getting general information; (3) understanding the system; (4) navigating; (5) finding out what topics or areas are covered in the system; (6) finding a particular topic; and (7) quitting.

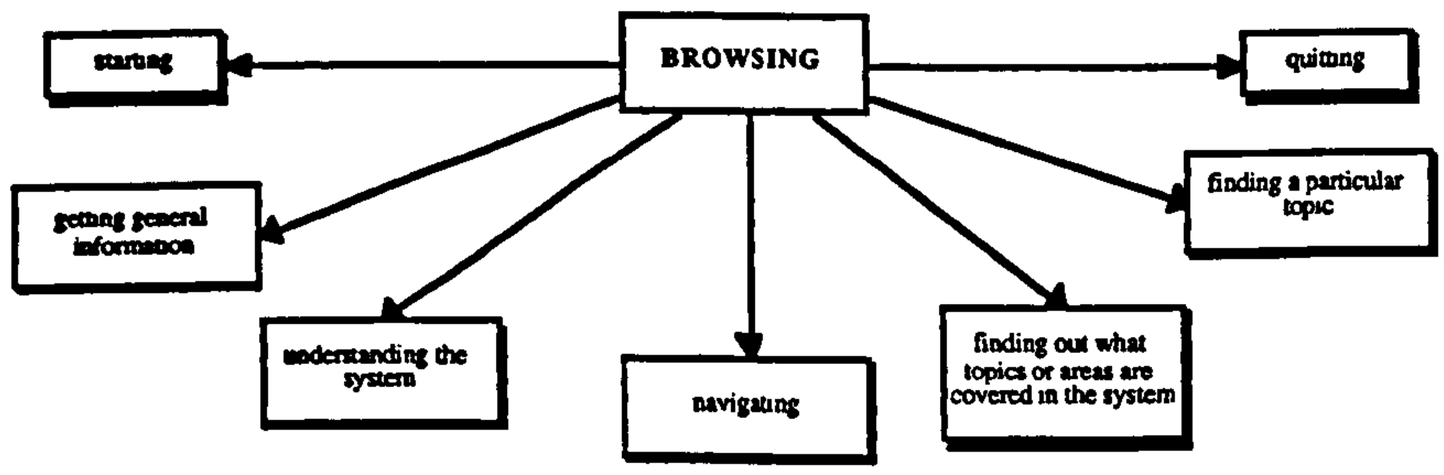


Figure 1. Task graph for browsing

Except for subtasks (4) and (5), the rest of the subtasks are goal-directed, that is, users have a goal in mind when executing them. Each subtask is then further broken down into

simpler, executable subtasks or actions. As examples, we have only included subtask graphs for *navigation* (e.g., unfocussed browsing) and *finding a particular topic* (e.g., focussed browsing) as shown in Figures 2 and 3.

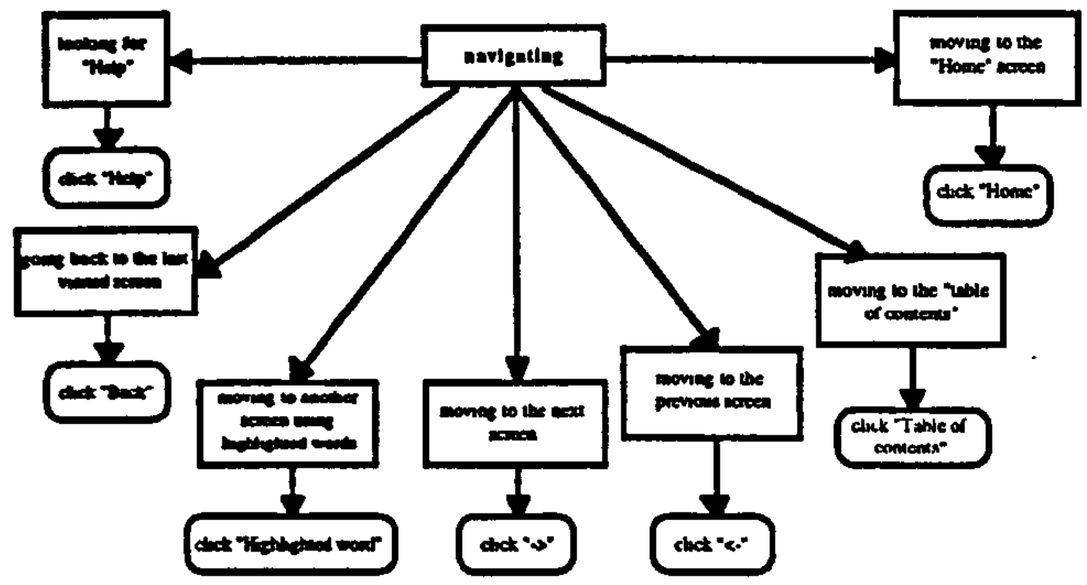


Figure 2. Task graph for navigating

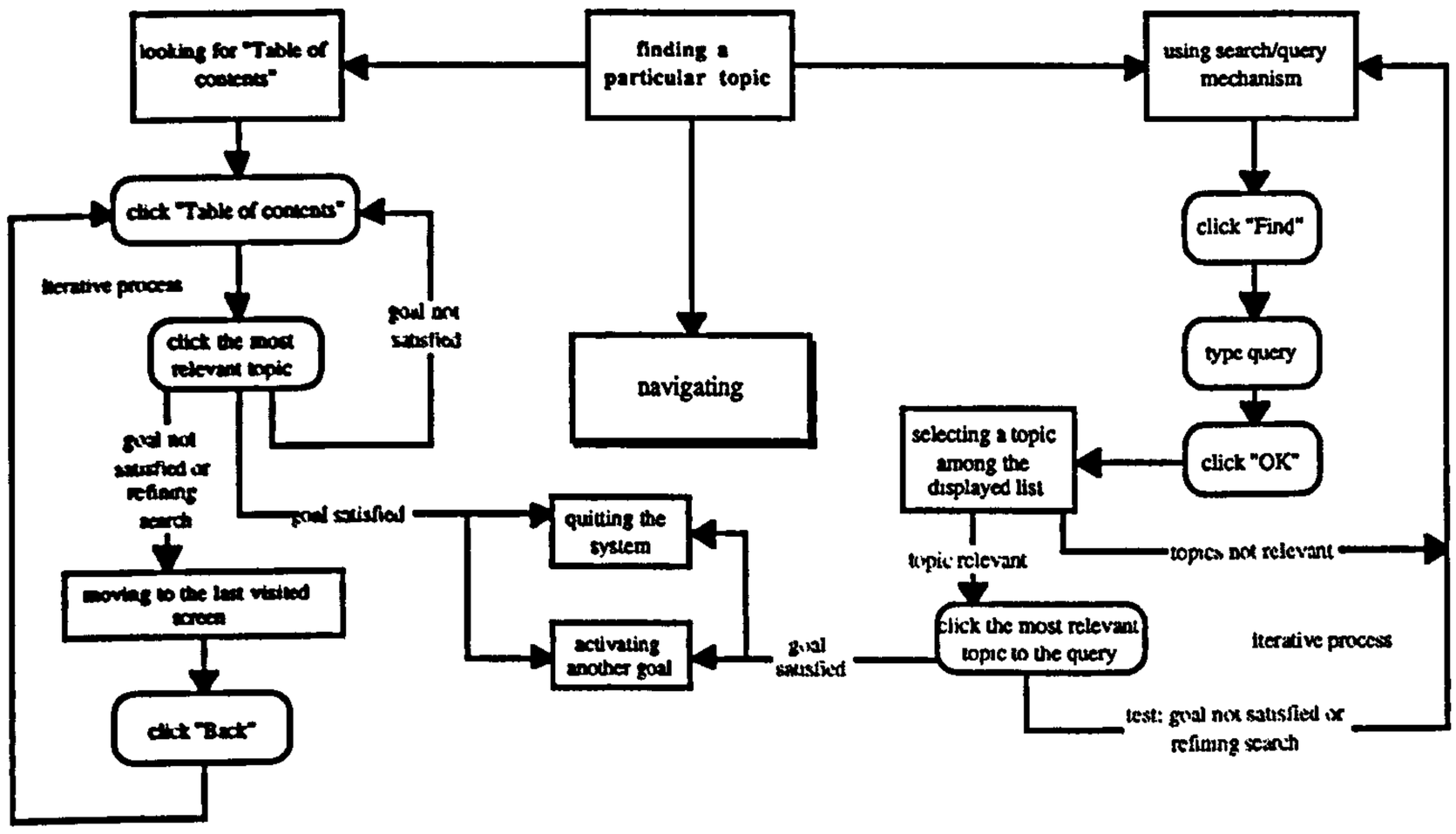


Figure 3. Task graph for finding a particular topic

Although subtasks are interrelated, the order in which they are performed depends on several parameters: users' domain knowledge of hypertext systems; users' domain knowledge of the contents covered in a particular hypertext system; users' intention or goal of using the system; and the context of users' tasks based on situated actions described in (Suchman, 1987). Unlike TD and related techniques that tend to focus on what actually happens, we also build into the task graphs subtasks that describe what *should* happen. We consider this an important inclusion since *browsing* is dynamic and therefore, hypertext systems supporting *browsing* should be adaptive as well as predictive.

Because *browsing* in hypertext is a cognitive activity, it is also essential that we take into consideration two aspects of cognition that are important to users browsing hypertext: reasoning and learning. Established TA techniques that are concerned with describing some aspects of the cognitive characteristics of users' tasks include the Model Human Processor, GOMS (Goals, Operations, Methods and Selections rules), Cognitive Complexity Theory, Task Knowledge Structures — there are many proposals. These cognitive TA techniques, however, are difficult to use and implement (Preece et al, 1994). Instead, because reasoning and learning are two important aspects of cognition that do need to be captured, we based our cognitive task graphs on Polson and Lewis' exploratory learning method (Wharton, Rieman and Polson, 1994). We made the assumption that learning is achieved when users are able to understand, induct, and subsequently, build links and concepts, as illustrated by the dotted links in the task graph for *trying out buttons options* (see Figure 4), which is one of the further subtasks associated to the subtask of *understanding the system*.

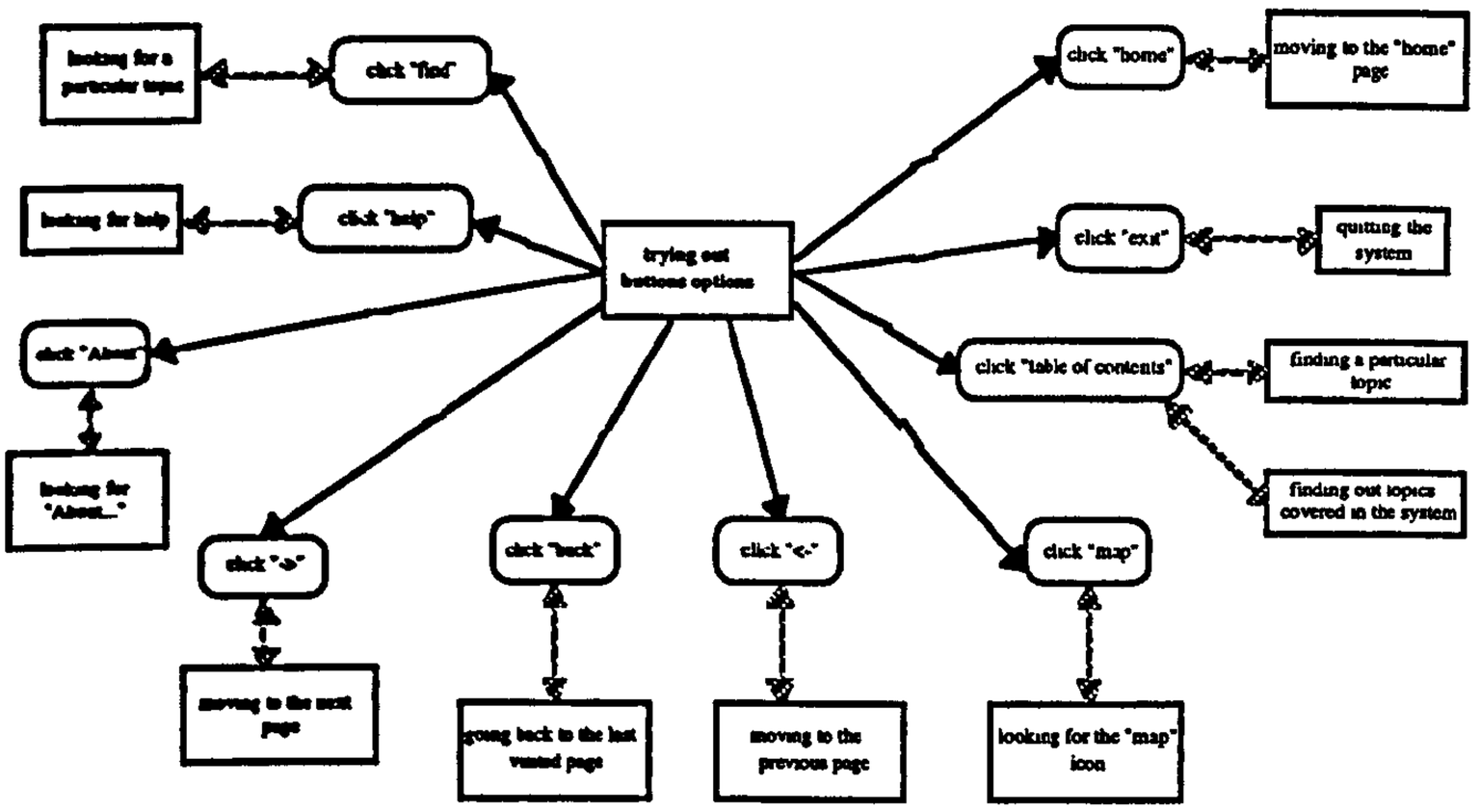


Figure 4. Task graph for trying out buttons options

5. Validating and refining cognitive task graphs

Real users need not be used only to evaluate designs, which is seen by many designers as the sole reason (Johnson, 1992), but also to generate design recommendations. From Rosson *et al.*'s study (1989), user-task analysis was among one of the several techniques used by designers for 'idea generation.' Other techniques involved using external sources such as available literature, meta-strategies such as concentration, design activities such as charts and diagrams, as well as trial and error.

To validate and refine the cognitive task graphs for *browsing*, a second investigation was carried out. We recorded two typical users' behaviour and actions using *ACM Hypertext-on-Hypertext* (ACM, 1989), a convenient hypertext system implemented in HyperCard, to complete some exercises associated with *browsing*. By conducting a video protocol of the users' interaction, we were able to interview users after the experiment and ask them to explain step-by-step why certain decisions and actions (e.g., moving to the next screen, going to "Home" screen, moving to the "Table of Contents") were taken while

using the hypertext. We were also interested to find out the frequency with which particular procedures were carried out, the circumstances under which one procedure was used rather than another and the inputs and outputs for each procedure. Users' feedback helped us understand their reasoning and learning processes when they chose to perform certain actions. The cognitive task graphs were then modified and refined accordingly.

It appears that this approach is very cost effective: an expert's initial task graph, then protocols of only two users, leads easily to an improved task graph. Clearly, one might scale up this procedure for use in a production environment, but our purpose was to obtain a structure suitable for the user models to be employed in hypertext design. Whether the additional cost of obtaining better task graphs would in fact be worthwhile is a separate issue; we suspect there would be diminishing returns, especially when the delay is considered.

Section 6 presents a cost-effective tool that enables designers to automate the task graphs, thus reducing the use of extensive and time-consuming real user validating and refining them.

6. From task graphs to executable models

The task graph can be considered a program. One of us (Rigny) has developed a cognitive architecture that can emulate the user executing such graphs. CUM-DesTool (Cognitive User Model Design Tool) is a general simplified architecture to build operational cognitive user models which enable designers to understand, predict and simulate users' behaviour and performance. The tool enables designers to model users' tasks and knowledge in terms of concepts and relations between these concepts.

CUM-DesTool's structure refers to ACT and the memory model proposed by Reason (1988) and is intended to enable a rapid and easy implementation of users' models. It can handle any type of user ranging from novice to expert. The tool enables designers to model users' tasks and knowledge in terms of concepts and relations between these concepts. For example, the users' task trees and knowledge are represented with concepts and various links (see example described in Figure 5). The links enable the modelling of users' cognitive processes and reasoning.

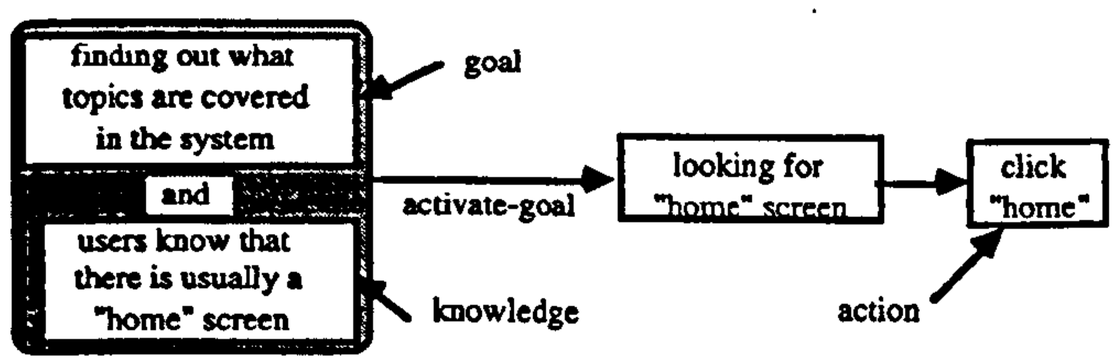


Figure 5. Description of the modelling process

A first version of the tool has been implemented in Common LISP. Crucially, using LISP allows the model to be embedded easily within other LISP programs, such as a general purpose hypertext development environment. There can be multiple instantiations of the cognitive models (randomised) and therefore we get the advantages of very fast, multiple user evaluation. The multi-module structure gives the tool some interesting properties:

1. It allows an *explicit representation* of the users' knowledge and cognitive processes which can support communication between designers and users, and therefore help designers to elicit users' needs.
2. It allows an *easy implementation* of a user model because there is a single representation structure for knowledge and tasks, and a single mechanism for reasoning.
3. *Various types of users* can be modelled by modifying few parameters. Designers are able to rapidly simulate types of potential users' behaviour from a generic user model. Such simulations are very useful to validate/test prototypes of the future system.
4. Measures of users' *cognitive workload, performance* and *satisfaction* can be identified and evaluated automatically.

The user's activity, e.g. when browsing hypertext systems, is for the most part goal-oriented. Figure 6 describes the cognitive processes which were incorporated into the user model. Users can identify a set of actions to achieve their goal and then select one action from this set of actions. If the initial goal is satisfied after performing for instance "action 1", a new goal can be activated, or the user can stop browsing the hypertext system. If the goal is not satisfied, users can perform another action from the initial set of actions" or can be disrupted from their initial goal and then, a new goal can be activated. The overall strategy which determines which action will be selected or whether there is disruption from the initial goal or not depends mainly on: users' general knowledge in hypertext; users' knowledge in the contents of the hypertext system they are browsing; and the context in which the user's activity is performed. The first two points determine users' profile. The third point refers to the situated actions described by Suchman (1987).

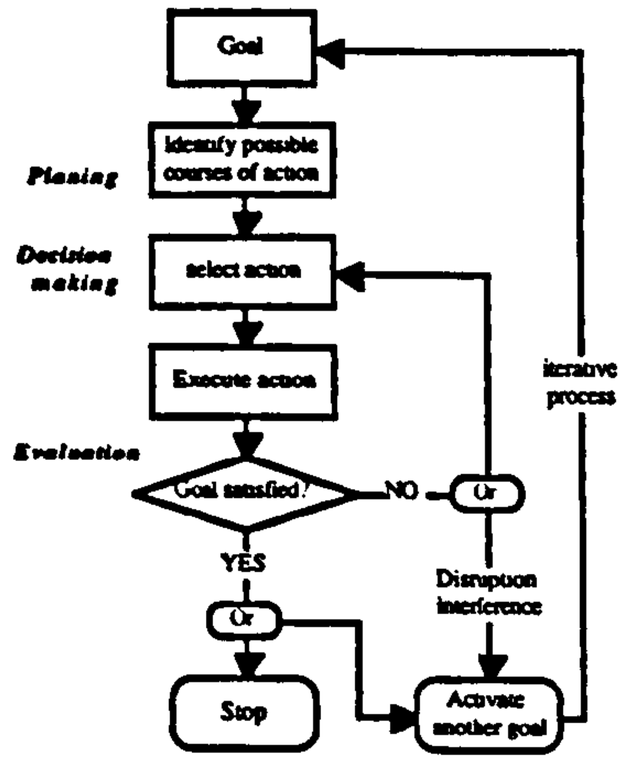


Figure 6. The cognitive model

Figure 7 shows the general interface of the executable cognitive user model for browsing provided by CUM-DesTool. The alphabetical list of concepts that have been implemented is displayed. When a goal is "set to true," for instance the "browsing" goal, the corresponding module of the simulation of the user's reasoning and activity is triggered. The trace of the reasoning mechanisms is stored and can be used to help designers understanding users' behaviour.

Several measures have been identified, including: *(a) the number of cognitive steps* performed by the user to achieve one particular goal; *(b) the time spent* by the user per goal; *(c) the number of documents browsed* by the user per goal, *(d) the number of visited hypertext nodes, (e) the number of failures*, goals which have not been reached, etc. **These measures can be automatically performed by the cognitive user model.** They enable designers to correlate subjective parameters, such as ease-of-use, with interaction metrics for various types of user. They yield useful insights and raise questions to be addressed by designers.

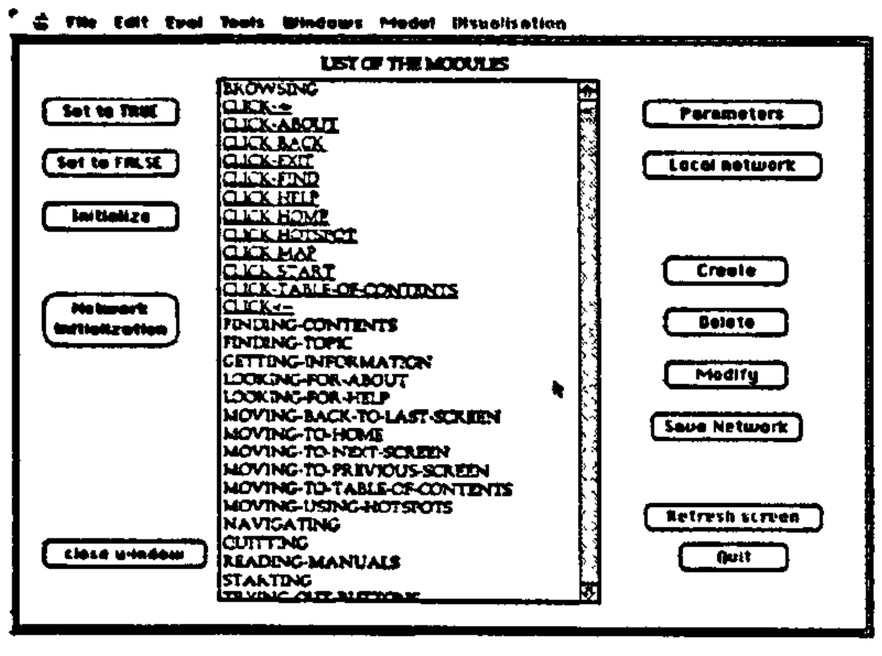


Figure 7. Example model description screenshot

7. Contributions to design

There are two aspects to TA (Johnson, 1992): gathering information and analysing users' tasks (described in Sections 4 & 5); and the generation of task models (described in Section 6) and task scenarios into running prototypes, in order that the outputs of the user-task analysis support a mapping and checking operation between user tasks and proposed design. In this section, we will describe how we used the information obtained from the cognitive task graphs for *browsing*, in conjunction with design principles and guidelines, to develop a design model, a prototype hypertext system. Design models can, of course, be represented in a variety of other forms: formal specifications, informal specifications, screen drawings, storyboards and simulations (Johnson, 1992).

Considerations for the design of the prototype hypertext system include: how best to support and improve the various aspects of the task *browsing*; how to support the way users could carry out actions while *browsing* hypertext; identify how much information should be provided on the screen at any one time, as well as give recommendations and suggestions on how to present information on the screen.
There are two aspects of design to which the cognitive task graphs directly contribute:

- *Design interface* is concerned with the visual appearance and screen layout of the user interface. This may include the design of pop-up menus, windows, icons, buttons, text fields, command names. It is used as a channel of communication through which the state of hypertext, accessible functions and feedback are presented to the user. Though cognitive task graphs may not give sufficient detail for complete screen layout or interaction style, they provide an initial input and guide for user-interface design.
- *Functionality* has to do with the performance and capability of the system. This may include navigational buttons, table of contents, home page, building links for learning.

Figure 8 shows our interpretation of the design of a simple prototype hypertext system produced from the cognitive task graphs. For example, if a user wants to perform *browsing*, then a pop-up menu is shown that allows the user to select any of the subtasks associated to *browsing* as described in Section 4.

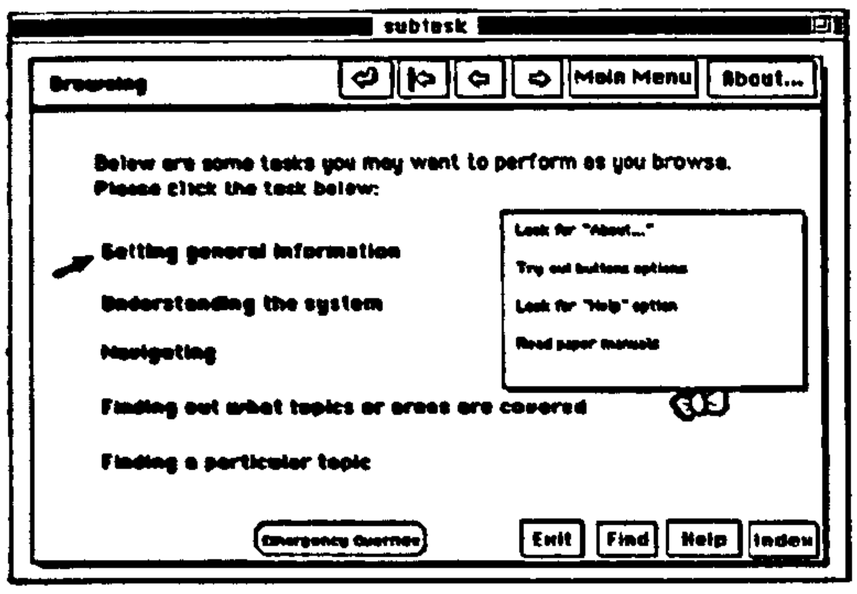


Figure 8. Interface design of a simple prototype hypertext system

8. Conclusions

We have shown developing cognitive task graphs for common tasks in hypertext that users' needs is straightforward. We showed that experts' initial task graphs are easily refined with cheap studies with actual users. We then showed that the resultant graphs can be interpreted by cognitive modelling tools. Based on the analysis of users' needs, we have implemented a prototype hypertext system.

The cognitive modelling tool, being an executable program, is itself executable from a hypertext system as if it was a user. Though we did not show this, it is standard LISP programming to achieve it. Not only does this allow automatic evaluation, it allows very large scale evaluation by introducing randomness into the model. Obviously such evaluation does not substitute for human evaluation, but it is cheaper, more comprehensive, and can identify critical incidents rapidly.

Subsequent work will involve building different cognitive user models for different user types (e.g., novice, intermediate, experienced).

Acknowledgements

We thank the users for taking part in the experiments. CUM-DesTool was built by Cécile Rigny as part of her PhD work sponsored by the French Ministry of Defence. The analysis of hypertext design issues was done by Yin Leng Theng as part of her PhD work sponsored by Middlesex University. The authors would like to thank Dr Ann Blandford and Dr Mark Addison for their helpful comments on earlier versions of this paper.

References

ACM (1989), "ACM Hypertext-on-Hypertext," ACM Press Database and Electronic Products Series.

Hansen, W (1971), "User engineering principles for interactive systems," *AFIPS Conference Proceedings*, **39**, AFIPS Press, pp. 523–532.

Johnson, P (1992), *Human Computer Interaction: Psychology, Task Analysis and Software Engineering*, McGraw-Hill.

Landauer, T K (1995), *The trouble with computers: Usefulness, usability and productivity*, MIT Press.

Lewis, C and Rieman, J (1995), "Getting to know users and their tasks," In Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S. (Eds) *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers (U.S.A.), pp. 122–127.

Newman, W M and Lamming, M G (1995), *Interactive System Design*, Addison-Wesley.

Payne, S and Green, T R G (1989), "Task-action Grammar: The model and its developments. In *Task Analysis for Human-Computer Interaction*, Ellis Horwood.

Preece, J, Rogers, Y, Sharp, H, Benyon, D, Holland, S and Carey, T (1994), *Human-Computer Interaction*, Addison-Wesley.

Reason J (1988). Framework models of human performance and error: a consumer guide. In L.P. Goodstein, H.B. Andersen & S.E. Olsen (Eds.), *Tasks, errors & mental models*. pp. 35–49, Taylor & Francis: London.

Rosson, M B, Mass, S, and Kellogg, W A (1989), "The designer as user: building requirements for design tools from design practice," *Communications of the ACM*, **31**(11), pp. 1289—1298.

Suchman, L A (1987), *Plans and situated actions: The problems of human-machine communication*, Cambridge University Press.

Thimbleby, H (1995), "Hypertext authoring without getting lost," *HCI'95 Adjunct Proceedings*, pp. 118—124.

Thimbleby, H (1990), *User Interface Design*, ACM Press (U.S.A.).

Wharton C., Rieman J. & Polson P. (1994). The Cognitive Walkthrough method: a practitioner's guide. In J. Nielson and R. Mack, Eds. *Usability Inspection Methods*. pp. 105–140. Wiley: New York.

"Lost in hyperspace": Psychological problem or bad design?

Yin Leng Theng, Matthew Jones & Harold Thimbleby
School of Computing Science, Middlesex University
Bounds Green Road,London N11 2NQ
Tel: +44 181 362 5000
Email: {y.theng, m.jones, h.thimbleby}@mdx.ac.uk

Abstract
A pervasive criticism of hypertext systems is that users tend to lose their way. Although much work has been done to address this "lost in hyperspace" problem, it still remains unresolved, even accepted as an inevtaible feature. But using the "lost in hyperspace" problem as an example of usability problems, we argue *usability*problems are not *user* problems. Once viewed like this, more appropriate solutions are applied to address it. We argue that some usability problems have been mis-classified as "psychological" and therefore are only palliated rather than cured. However, we show that there is much scope for new systems approaches that avoid such problems, and which offer considerably more creative ways of going about design of interactive systems.

Keywords
Hypertext, lost in hyperspace, conceptual model, executable user models, multidisciplinary design

1. Introduction

Ever since Vannevar Bush envisioned his hypertext Memex in 1945 (Bush, 1995), many diverse hypertext systems have been designed and built to meet a variety of functions: tutorial and educational needs, support for collaboration among a number of individuals, to production of on-line manuals. In hypertext, users not only benefit from the information they read but also from the richness of associations supported by the network of nodes and links. However hypertext systems are often criticised since in practice it is found that users readily lose their way in the network. After a few minutes a promising network of associations becomes a maze, with a lost user who has forgotten what he was trying to do. Ironically, the more useful a hypertext, the sooner a user gets so distracted he gets lost!

Fifty years later, Bush's Memex idea finds routine application in almost all modern CD-roms as well as in the World Wide Web. The usability of hypertext is a real issue. Even tiny usability gains for each of the millions of the Web's users would collectively aid humanity significantly!

In general, the "lost in hyperspace" phenomenon refers to any of the following conditions: users cannot identify where they are; users cannot return to previously visited information; users cannot go to information believed to exist; users cannot remember what they have covered; and users cannot remember the key points covered. The last point is worst: the others might have been bearable had users been able to get value from the experience even though it felt like being lost. Users really are lost; it is not just a superficial disorientation. The name of the problem, which we abbreviate *LIH*, is very appropriate! No wonder the most popular forms of system to design are computer games, where getting lost is made into fun - in the tasks fo which hypertext is promoted, though, it is *not* fun, a point well made by Carroll (1982).

2. The conventional design process

All user-centred methods can be recruited in design to reduce the impact of LIH:

- *Getting to know users and their needs.* It is a well-known fact that designers often design for themselves unless they are trained to realise that people are diverse, and that users are unlikely to be like them. Approaches to task analysis and cognitive modelling could be used to help designers build an accurate representation of users' behaviour and actions when they perform or try to perform common tasks such as browsing, information search, seeking references and recall. These findings could be used to guide hypertext design.

- *Early and continual user testing.* Because it is imperative that hypertext designers build into hypertext systems an accurate representation of user models, early and continual user testing is essential.

- *Iterative design.* Interactive systems require iterative design. The most promising approach is to iterate design and evaluation until a satisfactory result is achieved. We need to ensure that good hypertext design guidelines and principles are incorporated into the building of hypertext in the first place (Theng, Jones and Thimbleby, 1995a).

- *Prototyping.* We need better support tools for hypertext authoring to reduce the complexity of hypertext authoring, thus freeing hypertext authors to concentrate on the design and structure of hypertext systems (Theng, Jones and Thimbleby, 1995b). With good support tools, prototyping of hypertext systems can be achieved more efficiently in quicker time.

The conventional approaches above make routine assumptions:

- Users have a wrong or incomplete conceptual model of how information is structured and linked within the hypertext system (Elm and Wood, 1985).

- Users experience 'lack of closure' since they are not able to tell the extent of a network or what proportion of relevant items remains to be seen (Shneiderman, 1992).

- Users face the 'embedded digression problem' where they lose track of digression since they are distracted from the main tasks by lots of interesting information (Foss, 1989).

- Users generally lack the experience in using hypertext for learning, and this makes it difficult for them to remember, consolidate and understand the semantic content of nodes, resulting in a lack of detailed memory of any particular item and an inability to summarise what has been covered (Foss, 1989).

- Users do not have adequate systems, that they should have graphical browsers and query/search mechanisms (e.g., Edwards and Hardman, 1989, etc.).

- Users do not use their senses fully. They should have multi-modal experiences, fully using sensory and motor skills, audio, stereolocation and so on. This view leads to solutions being sought in virtual reality.

Note that all assumptions are referenced to the users' supposed failings: *users* have a wrong or incomplete conceptual model; *users* lack experience in using hypertext for performing tasks such as browsing; *users* are distracted because of the "embedded digression problem"; *users* don't understand the chosen display conventions; etc. And because we know that hypertext systems are difficult to build, many commentators are happy to seek solutions in better understanding users and helping them cope. There has, of course, been some success in this approach - maps, virtual reality visualisation, *etc.* - that no doubt seems to confirm it.

However, the repeated emphasis on *user* problems (ironically for such a promising technology!) seems suspicious. It seems to us, rather that *anything* would get lost in current hypertext systems. If so, then the problem is not based in user perception nor in user psychology. We might note that successful systems tend to be realistic (e.g., simulation environments) and that better realism is always attainable through incremental expenditure on the equipment. For example, by using a faster processor, bigger disc and a higher resolution display, then better pictures can be displayed. As systems get better *in this way* they get more expensive. It is possible, then, that cognitive dissonance influences the popularity of psychoperceptual (realism) solutions to LIH: the more expensive a solution, the less likely anyone is to question its appropriateness, especially when

spending a little more on it would make it better. (This is also hill climbing, a simple problem solving strategy that fails in all but the simplest cases.)

Note. It seems to us that anything would get lost. By "anything" we mean any device capable of algorithmic thought. To show that anything in this sense gets lost, then, we need to show that not being lost invokes non-computable functions, which we do by contradiction.

Consider (as just one possible way of arguing) the analogy of a Turing Machine (*head, tape)* with User (*head, hypertext)*, from which it follows that if what the user does with the hypertext is computable then, in particular, it is computable in finite memory (in fact, if the hypertext is writable, only a two state memory is required). Now consider a computable function, *f*: screen display X screen display -> user actions, with the intended meaning that given one display and an intended display, *f* gives the user actions that take the hypertext from one screen to the other. To evaluate *f,* the user must be able to distinguish screens. We can clearly construct a hypertext (indeed, a realistic hypertext!) having more screens than the bounded state space the user's head can distinguish. Therefore *f* is not computable under the assumptions (recall that it must be computed by the user).

One might argue that it is necessary for the user to distinguish screens since any well-designed hypertext would provide appropriate labels on screens, and labels on buttons (or other potential actions, such as menus) leading to screens. However, this only makes *f* trivial fo adjacent screens, or for screens a bounded number of steps apart. In the limit, if each screen refers to every other then *f* is computable if it is computable for certain hypertexts (rather dull ones at that; being complete graphs they have no organising structure); in general, *f* remains uncomputable.

The Church-Turing Thesis suggests that human users have no way around this non computability. As *f* is not computable, users will at best have a tough time deciding waht actions to take, they can only be given approximate advice by help systems, and they will readily become lost.

The Church-Turing Thesis is rather strong. More psychological arguments can be found in Johnson-Laird (1993), which suggest that users are even more limited than our arguments to show the inevitablity of LIH required. *End of note.*

3. The central issues

Hypertext design is hard, and hypertext systems are used less effectively than we would wish. Is the problem

- *A design problem*? Design (for whatever reasons) is done poorly. Poor design causes psychological problems.

- *A user problem*? The problem may be entirely due to users' inability to exploit computer screens, complex information structures, and that nothing in the design is to ameliorate this. Thus, as a psychological problem it can be alleviated but not solved by better design.

Most people think it is the latter, with the result that improvements are sought in presentation of the information.

We argue that there is more truth in the former view. We believe users now enter a world of distractions (of dramatic sounds and images) and tend to blame themselves when they get lost.

3.1 The book analogy

It is interesting to contrast this state of affairs with book design:

- book design is also difficult (there are good and bad authors)

- but readers can and do make accurate judgements of a book's quality. They make these judgements easily. They do not assume that the difficulty of understanding a badly designed book is their own fault.

- a book (or any other serial medium, like a film) can be evaluated in reasonable time, whereas a hypertext can take practically unlimited time to evaluate. Every time the designer adds a choice for the user, the

user's job of understanding the book can double. After only twenty choices, the user can no longer model the complexity of the hypertext.

- we take for granted very many organisational structures in books, like page numbers and alphabetic orders in reference books. Though they can be used in hypertext, none are useful *central* organising principles in hypertext - because if they were used the generality of hypertext is defeated, and we've just got an electronic, conventional book.

The book's organisational principles took centuries to develop (Thimbleby, 1992). How can we more quickly find better organisational principles for hypertext?

We now argue for a move away from treatment to prevention, from treating the user's symptoms - themselves a reaction to bad design - to avoiding the bad design.

3.2 Users are robots

For too long we have thought of users as being humans. Humans are remarkably adaptive, know a lot about the world, and adhere to deep conversational conventions. Yet in hypertext we wish to construct "virtual worlds" (virtual or metaphorical) with which the user has no prior experience. Indeed if users knew some hypertext world well, there would be less point in using it (we are making assumptions about what hypertext is for) - it is supposed to be an enhancement to users, not a substitute for something they already have.

When a function is non computable, it has to be evaluated by some extra-computational method. Since users are humans, it follows that hypertext systems (as presently conceived) *rely* on human skills - and humans' perseverance and their willingness to put up with bad systems. It would be better to design systems that in the first place did not rely on human skills to disguise their poor design. Instead, systems should be designed for robots to use. And only when it is *known* that a robot can use a system should additional presentation be added to make the system more attractive for human appreciation. At present we commonly make systems the other way around: we make them stunningly attractive, and then wonder how to make them usable. Often designers don't get that far, because the attractiveness is sufficient to disguise systems' fundamental usability. If a programmer designs a program, only half the job is done if they have only designed the data structures. They also have to design the procedures for operating on the structures. (Specifically, a programmer designs abstract data types.) Without the appropriate procedures for operating on data structures, a computer would literally get lost in the structures, even supposing it could start executing anything sensible. Notice that hypertext is always defined as a data structure! There is no emphasis on the user's procedures for navigating, finding information, or doing any general operations (Thimbleby, 1992). Therefore, users use whatever procedures "come to mind" - which we've suggested above are likely to be inappropriately carried over from the real world. The result is LIH.

3.3 Robots are not enough

We argued that treating users at least as well as we treat robots would be a good idea (see also Thimbleby, 1995c). Even so, standard user-centred practice in usability engineering is still required; because all the arguments for usability engineering are still valid (Nielsen, 1993). Unfortunately statistically useful involvement of users in hypertext design is infeasible (hypertext systems are combinatorially too large). Given our robot-oriented design stance, it is natural to suggest that what designers need is robots to model users. We call such "robots" executable user models.

Executable user models are software agents that simulate users. They can be faster, tireless, and they do more exhaustive checking of system designs. Moreover, they can embed multi-disciplinary knowledge that most designers and most users would not be expected to know or be able to verbalise in their accounts of interaction. Executable user models can work with hypertext prototypes, long before they have reached a stage where actual human user evaluation would be practicable. It is even possible for executable user models to be in iterative design cycles, managed by suitable artificial intelligence strategies.

4. Not merely difficult, but full of opportunities

We've argued that hypertext design is difficult, and viewing it as primarily a computational problem does not make it better. What it does do, though, is open up a very large range of potential solutions that have, so far, not been explored.

We mention three possibilities.

First, viewing the user as a robot suggests we search for appropriate algorithms to implement whatever tasks the user is engaged in. Certainly, algorithms books are much larger than the range of features that current interactive systems provide, so there are plenty new user interface styles to choose from! Such ideas have been explored at greater length elsewhere (Thimbleby, 1990 & 1992), where examples may also be found.

Second, computerised users in the form of executable user models could be used to simulate users' behaviour to help designers evaluate their systems without requiring (human) user attendance. Metrics on user performance, cognitive overload and satisfaction could provide approximations on the usability of the hypertext systems, which would pinpoint weaknesses of their systems and reduce the length of the evaluation process. We have started explorintg such approaches with our colleague Cecile Rigny who has implemented a LISP tool for running cognitive models, called CUM-DesTool. The LISP model simulates user cognition, but it is also a program that can run interactive systems, and therefore substitute for actual users in evaluation. Moreover, by randomising user models, it is possible to simulate large groups of users and to obtain useful statistical information.

Yin Leng Theng has built a hypertext prototyping system, called HyperAT, also written in LISP. We are now in the process of combining the two tools to explore the ideas proposed in this paper.

Third, it is clear that *ad hoc* methods of designing, constructing and validating hypertext systems are not enough. If users get lost in hypertext, designers do too. (The theoretical arguments are easier than for users: having an effective procedure for designing hypertext systems where users do not get lost is equivalent to the halting problem.) This suggests that tools for designing hypertext should provide much improved computational support for designers (Thimbleby, 1995a).

5. Conclusions

Although much research effort has been invested to address the LIH, it still remains unresolved. We argued that wrong or inappropriate solutions are being sought because incorrect or incomplete assumptions are made. Disorientation can arise in conceptual space (within the user's mind), which most research findings support - but we argue the "blame" should not rest on users alone! LIH is surely attributable to bad system design, and is not a psychological *problem* - it is just a psychological *symptom*. This paper has set down the reasons for a new approach designing hypertext.

We furnished an outline theoretical argument about the computability of the user performing certain tasks within hypertext. This line of enquiry could be extended. For example, it would be worthwhile exploring the complexity of the user performing certain actions and how to design hypertexts to optimise the user's performance (Thimbleby, 1995b). It would also be interesting to explore other computational models: even the simple Turing Machine model suggests that having user-writable hypertexts would make the user/hypertext system more expressive.

Since LIH is a complex problem, a multi-disciplinary approach is necessary to draw upon and integrate knowledge and findings in seemingly diverse disciplines such as Cognitive Psychology, Artificial Intelligence and Software Engineering, and to integrate this knowledge into human-computer interaction. This paper has shown some ways of doing this.

References

Bush, V (1945), "As We May Think," *Atlantic Monthly*, **7**, pp. 101-108.

Carroll, J M (1982), "The adventure of getting to know a computer," *IEEE Computer*, **15**(11), pp. 49-58.

Edwards, D M and Hardman, L (1989), "'Lost in hyperspace': cognitive mapping and navigation in a hypertext environment," *Hypertext: Theory into Practice*, Intellect Books, pp. 90-106.

Elm, W and Woods, D (1985), "Getting lost: A case study in interface design," *Proceedings of the Human Factors Society 29th Annual Meeting*, pp. 927-931.

Foss, C L (1989), "Tools for reading and browsing hypertext," *Information Processing and Management*, **25**(4), pp. 407-418.

Johnson-Laird, P N (1993), *Human and machine thinking*, Lawrence Erlbaum Associates.

Nielsen, J (1993), *Usability Engineering*, Academic Press.

Shneiderman, B and Kearsley, G (1989), *Hypertext Hands-On! An Introduction to a New Way of Organising and Accessing Information*, Addison-Wesley.

Theng, Y L , Jones, M and Thimbleby, H (1995a), "Reducing information overload: A comparative study of hypertext systems," *IEE Colloquium on "Information Overload,"* Digest No: **95/223**, pp. 6/1-6/5.

Theng, Y L , Jones, M and Thimbleby, H (1995b) , "Designer tools for hypertext authoring," *IEE Colloquium on "The Authoring and Application of Hypermedia-Based User Interfaces,"* Digest No: **95/202**, pp. 4/1-4/4.

Thimbleby, H (1990), *User Interface Design*, Addison-Wesley.

Thimbleby, H (1992), "Heuristics for Cognitive Tools," in NATO ASI Series **F**, *Proceedings NATO Advanced Research Workshop on Mindtools and Cognitive Modelling, Cognitive Tools for Learning*, Kommers P A M, Jonassen D H & Mayes J T, editors, pp. 161-168, Springer Verlag.

Thimbleby, H (1995a), "Hypertext authoring without getting lost," *HCI'95 Adjunct Proceedings*, pp. 118-124.

Thimbleby, H (1995b), "Users as computers": An approach to VR design and conceptual evaluation, *Proceedings Interface to Real and Virtual Worlds*, **IV**, pp. 305-313.

Thimbleby, H (1995c), "Treat People Like Computers?" *Extraordinary People and Human-Computer Interaction*, Edwards A, editor, Cambridge University Press, pp. 283-295.

Improved Conceptual Design For Better Hypertext

Yin Leng Theng, Cécile Rigny, Harold Thimbleby & Matthew Jones

School of Computing Science, Middlesex University
Bounds Green Road,
London N11 2NQ

Tel: *+44 181 362 5000*
Fax: *+44 181 362 6411*
EMail: *{yin2, cecile1, harold, matthew16}@mdx.ac.uk*

This paper discusses an iterative, engineering approach with an improved Conceptual Design Stage for the design of better hypertext. We emphasize the importance of getting design right "up front" so that many design blunders can be avoided. Task analysis and cognitive user modelling techniques are used to make design recommendations for rapid prototyping and testing of a prototype hypertext. We also explore the potential of executable user models as a cost-effective means to rapidly iterate and test design, without the attendance of real users.

Keywords: Hypertext, cognitive user modelling, iterative design, conceptual design

1 Introduction

Hypertext systems are hard to design and develop successfully. This is because hypertext systems consisting of a rich network of interconnected nodes and links present to hypertext authors design choices that are difficult to manage [8]: vast number of potential structures to create hypertext systems; and astronomical number of ways to create links. Because designing and producing hypertext is a relatively new discipline, there are few established tools or procedures that designers can readily use to assist them build well-structured hypertext systems [3]. As a result, most hypertext systems are poorly designed and built in terms of how information is structured and displayed.

One of the problems is the lack of a disciplined and systematic approach to designing well-structured hypertext systems that will meet users' needs [3]. We need a cost-effective way of designing that adheres to good usability practice yet avoids as many

design errors as soon as possible. The more errors that can be avoided "up front" by the right method, the less work both test users and designers will have to put in to make prototypes acceptable. If conventional iterative development process is found lacking, we need better ways to ensure that better hypertext systems are produced. Figure 1 shows the six stages described in the well-accepted iterative development process.

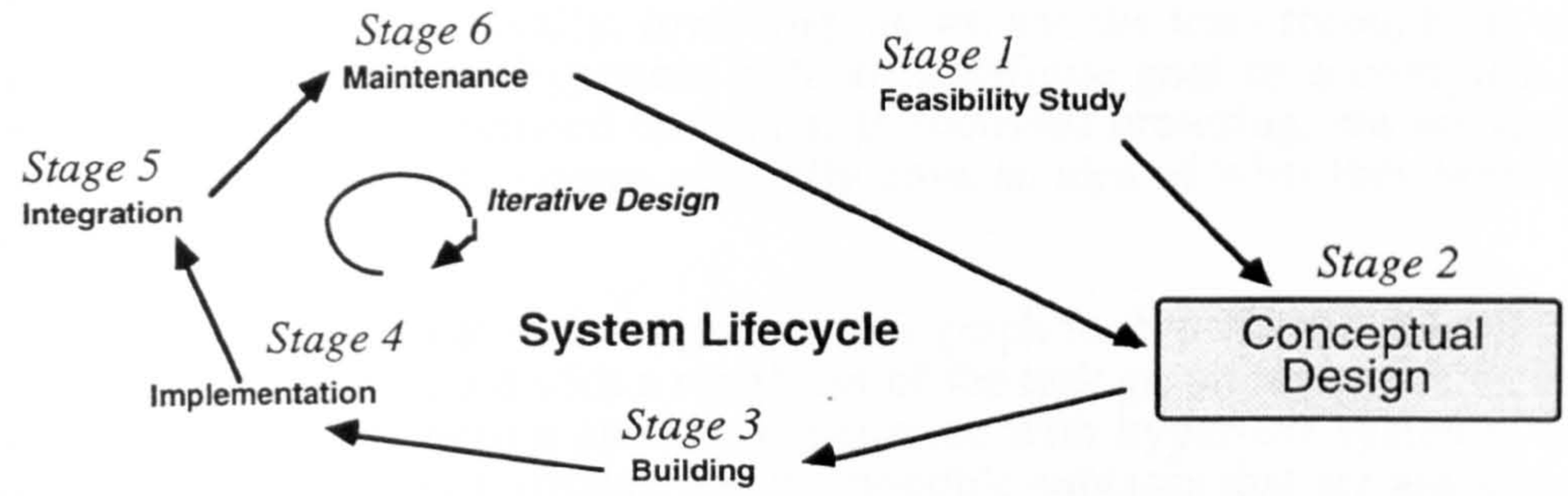


Figure 1. Conventional iterative development process

This paper argues *Conceptual Design* (Stage 2 of the conventional iterative process) can be usefully systematised (see Figure 2). We present our work including new tools, prototypes and studies, justifying this claim, and its encouraging preliminary results. We believe Conceptual Design is vital to the success or failure of the final system as it encompasses all activities relating to gathering and analysing users' needs, specifying users' requirements, building and testing prototype. This paper discusses how some of these activities can be systematically and effectively achieved. Our objectives include:

1. analysing tasks users want to perform (or try to perform) using task analysis techniques, leading to the building of cognitive task graphs, which are then used for design recommendation for building a prototype hypertext;

2. investigating whether a prototype built with the design recommendations from task graphs performs better than a prototype built without using the design recommendations; and

3. investigating the potential of using executable user models in place of real users for rapid prototyping and testing.

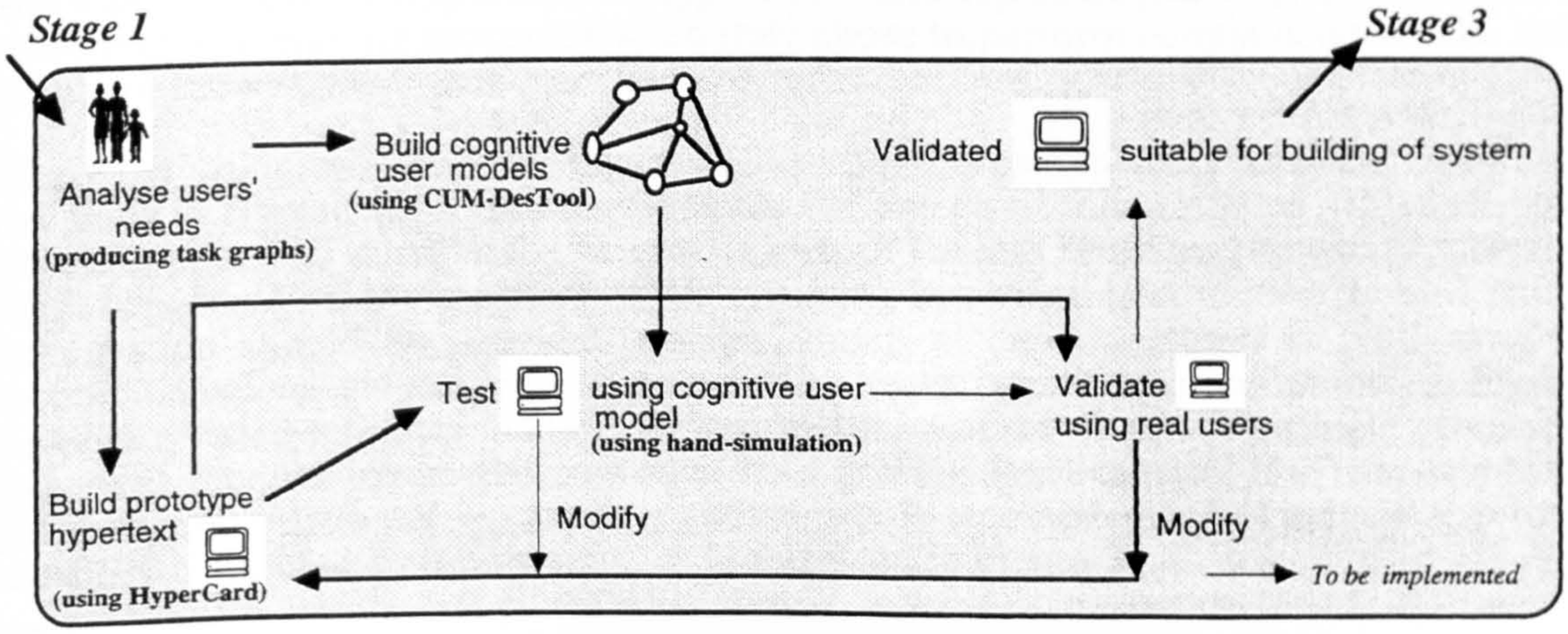


Figure 2. Improved Conceptual Design for better hypertext

2 Investigation I*

To achieve Objective 1, we need to know users and their needs by analysing the common tasks they want to perform or try to perform [e.g., 4, 5, etc.]. Based on the kinds of support provided by hypertext systems, we identify four representative tasks: browsing; information search; seeking references and recall. As an illustration, we have selected one out of these four: the task we call *browsing* to illustrate how cognitive task graphs were built. (Cognitive task graphs for other tasks can be developed similarly.) Generally, *browsing*, as we use the term throughout, refers to reading and navigating in hypertext without a definite goal to accomplish. In this paper, we also include focussed browsing. In focussed browsing, we are saying that users browsing hypertext systems generally have an idea of what they want to do to accomplish a goal.

To gather inputs for building the cognitive task graph for browsing, we performed the following activities: started with a rough cut of the task graph for browsing based on known facts about browsing and our experience with hypertext systems. Figure 3 shows the task graph for browsing and the possible subtasks that are associated to it: (1) starting; (2) getting general information; (3) understanding the system; (4) navigating; (5) finding out what topics or areas are covered in the system; (6) finding a particular topic; and (7) quitting.

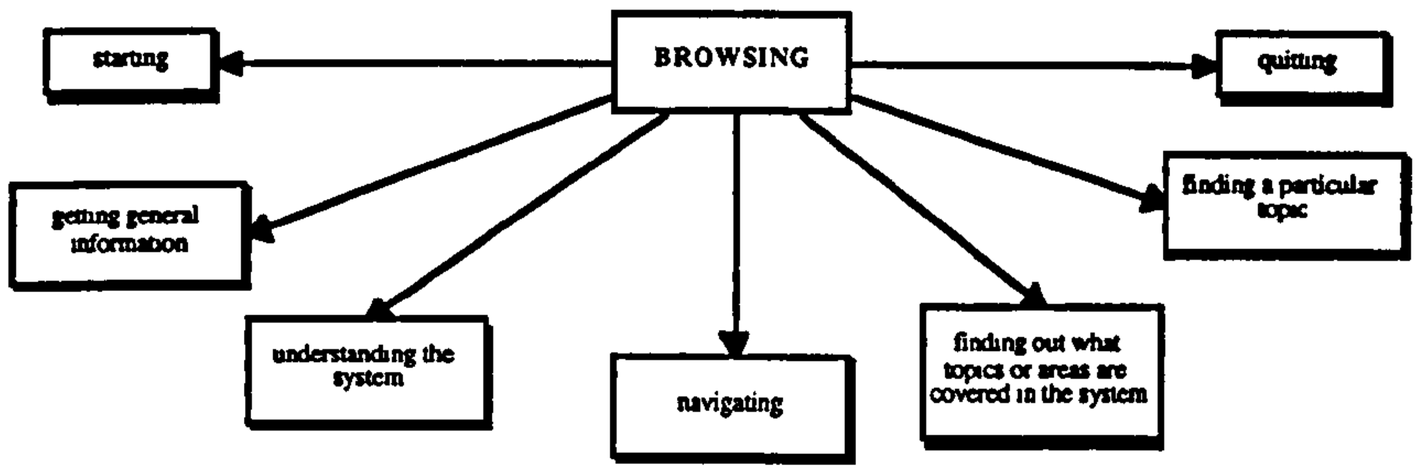


Figure 3. Task graph for browsing

To validate and refine the task graph for browsing, we recorded two "typical" users' behaviour and actions when using *ACM Hypertext-on-Hypertext* [1], a hypertext system, to complete some exercises associated to browsing. By "typical", we refer to a user who has general knowledge about navigation with hypertext systems. Through conducting this video protocol, we were able to interview users after the experiment and ask them to explain step-by-step why certain decisions and actions (e.g. moving to next screen, going to "Home" screen, moving to the "Table of Contents", etc.) were taken while browsing *ACM Hypertext-on-Hypertext*. Their inputs helped us understand users' reasoning and learning processes when they chose to perform certain actions. The task graph for browsing was then modified and refined accordingly.

Using the task graph for browsing, we built a prototype hypertext on *Basic Computer Anatomy* in HyperCard. Considerations for the design of the prototype included: how best to support and improve the various aspects of the task browsing; how to support the way users could carry out actions while browsing hypertext; how to identify how much information should be provided on the screen at any one time, as well as give recommendations and suggestions on how to present information on the screen. Figure 4 shows a visual representation of our interpretation of the design of a simple prototype hypertext produced from the cognitive task graphs. For example, if a user wants to perform browsing, a pop-up menu is shown which recommends the kinds of activities normally associated with browsing, as highlighted in Figure 3.

---

* Investigation I is described in greater detail in [7]. It is presented in this paper to provide the framework for Investigations II & III.

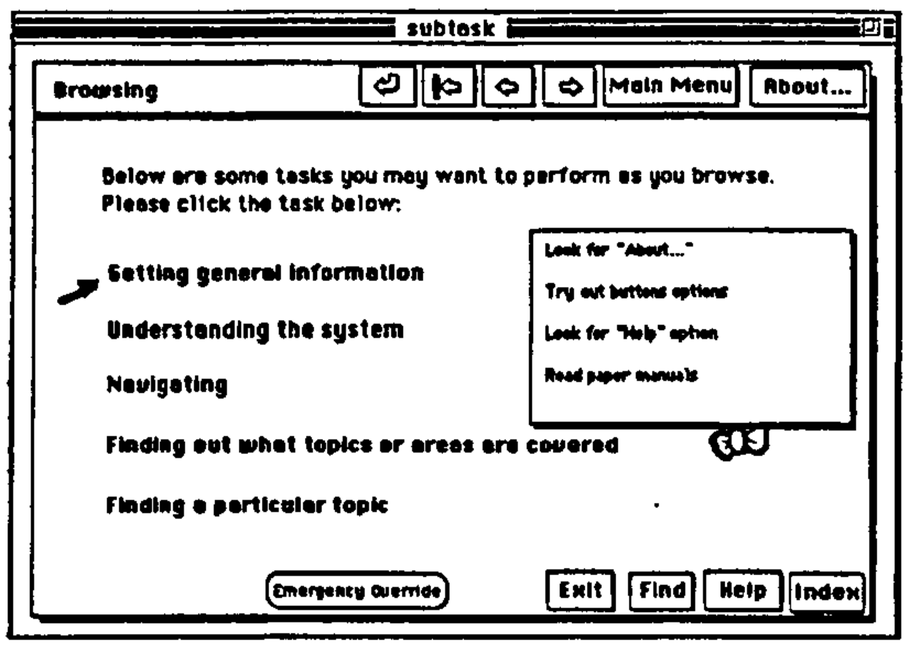


Figure 4. Interface design of a simple prototype hypertext

3 Investigation II and preliminary findings

To find out whether a prototype hypertext built using the design recommendations in Section 2 is better than a prototype hypertext built without the design recommendations (see Objective 2), we built another prototype hypertext on *Basic Computer Anatomy* in HyperCard using the conventional iterative "intuitive" development process recommended by Beekman [2]. We then selected a "typical" user to evaluate the two prototypes. She was asked to complete some exercises associated to browsing. She was then interviewed and asked to complete a questionnaire on the performance of the prototypes.

There was no significant difference in terms of her performance when using the prototypes as she was able to complete her tasks successfully in both cases. However, she indicated that she was more satisfied with the prototype that incorporated design recommendations such as quick tips for browsing, which included recommendations for possible activities normally associated to browsing (described in Section 2, Figure 4). She also rated it to be more effective, and as such she was more confident using it. These findings, though not entirely surprising, confirmed that users' needs can be straightforward if they are systematically analysed through using cognitive task graphs as shown in Section 2. Based on her feedback, we modified the prototype with the design guidelines. We then asked the same user to evaluate the modified prototype and she was more satisfied with its performance, compared to the other two prototypes.

4 Investigation III and preliminary findings

It appears that the approach described in Sections 2 & 3 is cost-effective: an expert's initial task graph, then protocols of only two human users, leads easily to an improved task graph. From the improved task graph, a prototype hypertext was built, which was evaluated to be effective in helping a human user complete her tasks successfully. Whether the additional cost of obtaining better task graphs and hence better prototype would in fact be worthwhile is a separate issue; we suspect there would be diminishing returns, especially when delay is considered.

To reduce the use of extensive and time-consuming real user validating and refining the task graphs, CUM-DesTool (Cognitive User Model Design Tool) has been developed to enable designers to automate the task graphs. CUM-DesTool's structure refers to ACT and the memory model proposed by Reason (1988) and is intended to enable a rapid and easy implementation of users' models. CUM-DesTool is a general simplified architecture to build operational cognitive user models which enable designers to understand, predict and simulate users' behaviour and performance [7].

Figure 5 shows the general interface of the executable cognitive user model for browsing generated by CUM-DesTool. The alphabetical list of concepts that have

been implemented is displayed. When a goal is "set to true," for instance the browsing goal, the corresponding module of the simulation of the user's reasoning and activity is triggered. The trace of the reasoning mechanisms is stored and can be used to help designers understand users' behaviour (see Table 1). By inputting into CUM-DesTool parameters such as users' profiles and tasks, different cognitive user models can be created. These models can then be used in *Stage 2* (see Figure 2) to build, test and modify prototype.

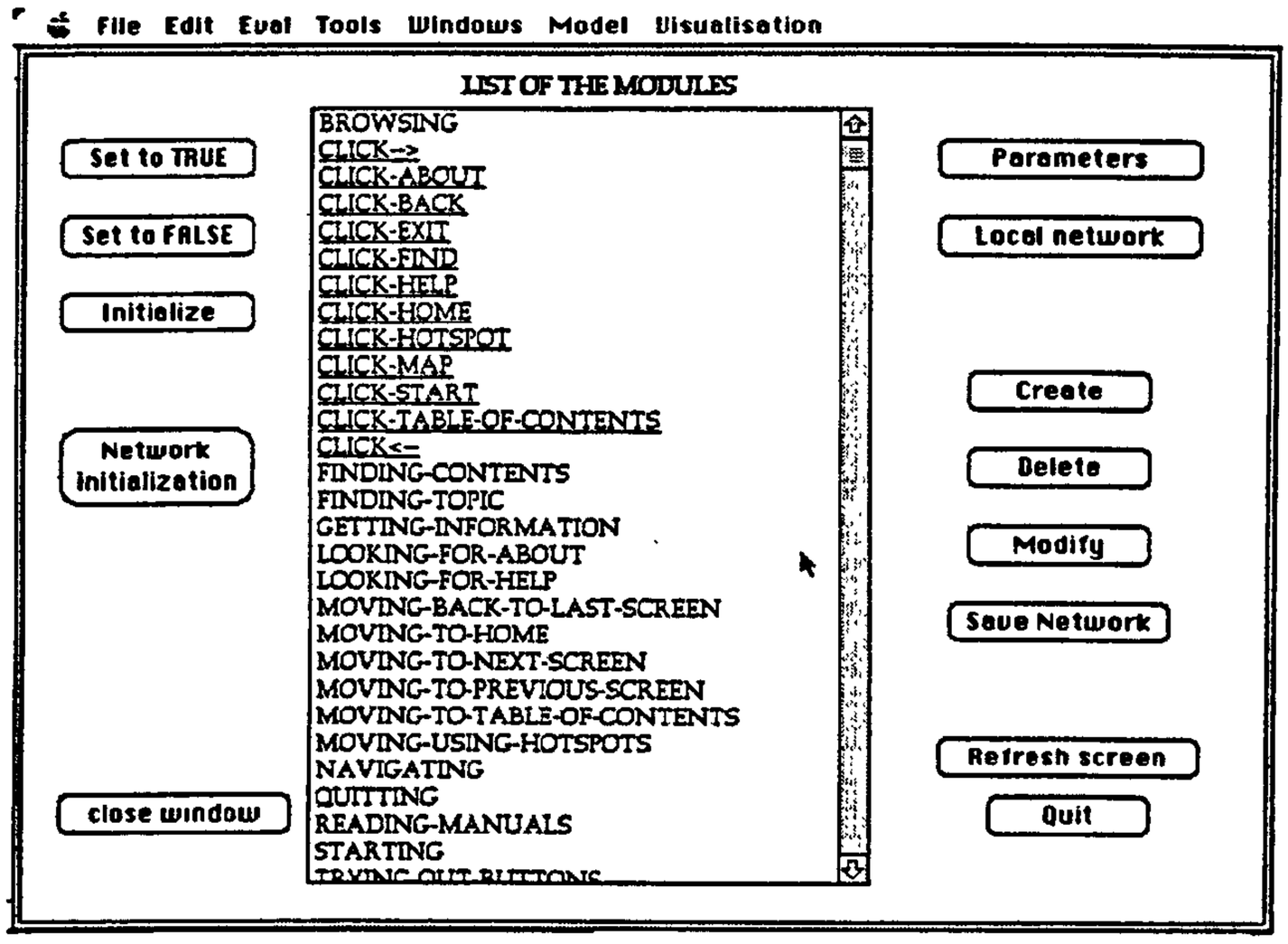


Figure 5. Example model description screenshot

To investigate whether CUM-DesTool is able to produce accurate enough executable user models (see Objective 3), we carried out the following experiment. We used the three prototypes built as described in Investigations I & II: one without the design guidelines; one with the design guidelines; and one with the design guidelines (improved based on user's feedback obtained in Investigation II). From Investigation II, we asked the user to write down the steps taken to perform the tasks using the three prototypes. We also recorded the user's feedback on what features were lacking in the prototypes. Using the executable user model for browsing, we carried out a hand-simulation of the user's behaviour to evaluate and analyse the usability of the three prototypes, by performing similar tasks the human user had to perform in Investigation II.

Table 1 shows the steps taken by the human user and executable user model in obtaining the answer to a particular task, for example, *finding a topic.*

Goal to perform: **Finding a particular topic**	Steps taken by **human user**	Steps suggested by **executable user model** (obtained from the *trace* and simplified for presenting below)	**Remarks**
(I) Prototype **without** the design guidelines	Look for "Find" button. Click "Find" button. Type in query. Select the relevant topic.	Browsing =>Finding a topic Finding a topic => Click-Find Click-Find => Type-Query Type-Query =>Select-topic	In the hand-simulation, the steps suggested by the executable user model matched the actions taken by the human user. This is because the "Find" button was found on the first screen of the prototype.
(II) Prototype **with the** design guidelines	Look for "Find" button. Click "Find" button. Type in query. Select the relevant topic.	Browsing =>Finding a topic Finding a topic => Click-Find Click-Find => Type-Query Type-Query =>Select-topic	In the hand-simulation, the steps suggested by the executable user model also matched the actions taken by the human user. This is because the "Find" button was found on the first screen of the prototype.

Table 1. Steps taken by the human user and executable user model in *finding a topic*

Goal to perform: **Finding a particular topic**	Steps taken by **human user**	Steps suggested by **executable user model** (obtained from the *trace* and simplified for presenting below)	**Remarks**
(III) Prototype **with the design guidelines (modified based on user's feedback)**	Look for "Find" button. Can't find button. Click on the next screen button. Look for "Find" button. Can't find button. Click on the next screen button. Look for "Find" button. Click "Find" button. Type in query. Select the relevant topic.	Browsing =>Finding a topic Finding a topic => Click-Find Click-Find not satisfied. Finding a topic => Looking-For-Table-Of-Contents Looking-For-Table-Of-Contents not satisfied. (Owing to space contraints, we will not list down all the possible buttons suggested by the user model such as "Back", "Previous", etc. When the suggestion was "Next", the simulation continued with going to the next screen of the prototype. The "Find" button was also not available in the second screen. The whole process continues for the second screen.) In the third screen the goal was satisfied because the "Find" button was found.	Based on user's feedback, the prototype was modified with the "Find" button only appearing in the third screen. The first two screens only provided general information about the system, which the user thought should only contain the "next" button. It is interesting to note that when the hand-simulation was performed, it took a number of cognitive steps to reach the third screen with the "Find" button. The goal was then satisfied like in (I) and (II).

Table 1(cont'd). Steps taken by the human user and executable user model in *finding a topic*

From Table 1, we note that the executable user model suggested steps that matched the way the human user would do to complete the task *finding the topic.* The goal was satisfied when the "Find" button was found. Prototype III, which was modified based on user's feedback, had the "Find" button removed from the first two screens. Therefore, it took us a number of cognitive steps to perform the task completely, when we carried out the hand-simulation. Whether the user was right in suggesting that the "Find" button was to be removed is debatable. What is interesting is that the executable user model randomly provided a list of suggestions that the human user would have in mind when performing the task *finding a topic.* This list of suggestions could provide valuable insights for designers to ensure the usability of

the prototype. Taking away the "Find" button may make one user happy but may also make other users unhappy, if so many cognitive steps had to be taken to perform the task successfully.

We based our design of the prototype hypertext on the recommendations of one "typical" user. However, it may be presumptuous to assume that all users behave like her. Although CUM-DesTool is in its early stages of development, initial results are encouraging. We see immediate benefits to the designers: validated executable user models generated by CUM-DesTool can be used to generate different user models, rapidly iterate and avoid many design blunders, thus reducing the use of extensive and time-consuming human users validating and refining them.

5 Conclusion and future work

This paper presents an improved *Conceptual Design Stage* built on top of the conventional iterative development process. Through the Conceptual Design Stage, we have presented ways of understanding users' behaviour and navigation issues when browsing in hypertext by giving them more structure using cognitive task graphs. We have shown that experts' initial task graphs can be easily refined with real users. Based on the task graphs, we have implemented a prototype hypertext system with the design guidelines recommended. Through user's studies, we have shown that it was rated better than the prototype without design guidelines. We have also shown that the resultant task graphs can be interpreted by CUM-DesTool, a cognitive modelling tool, to produce executable user models. We then compared the performance of the executable user model for browsing with real users. There is, thus great potential in using executable user models for rapid prototyping and testing, without requiring real users' attendance.

Subsequent work will involve building different cognitive user models for different user types (e.g., novice, intermediate, experienced) and different tasks (e.g., information search, seeking references and recall).

Acknowledgements

The authors would like to thank the users for taking part in the experiment, Ann Blandford and Mark Addison for their helpful comments in the earlier version of the paper.

References

[1] ACM (1989), "ACM Hypertext-on-Hypertext," ACM Press Database and Electronic Products Series.

[2] Beekman, G. (1995), *HyperCard 2.2 in a hurry: The fast track to multimedia*, Wadsworth Publishing, U.S.A.

[3] Glushko, R.J. (1995), "Seven ways to make a hypertext project fail," In Baecker, R.M., Grudin, J., Buxton, W.A. and Greenberg, S. (Eds), *Readings in Human-Computer Interaction: Toward the Year 2000 (Second Edition)* , Morgan Kaufmann, pp. 849-853, U.S.A.

[4] Hansen, W. (1971), "User engineering principles for interactive systems," *AFIPS Conference Proceedings*, **39**, AFIPS Press, pp. 523-532.

[5] Lewis, C. and Rieman, J. (1995), "Getting to know users and their tasks," In Baecker, R.M., Grudin, J., Buxton, W.A. and Greenberg, S. (Eds), *Readings in Human-Computer Interaction: Toward the Year 2000 (Second Edition)* , Morgan Kaufmann, pp. 122-127, U.S.A.

[6] Reason J (1988). Framework models of human performance and error: a consumer guide. In L.P. Goodstein, H.B. Andersen & S.E. Olsen (Eds.), *Tasks, errors & mental models*. pp. 35–49, Taylor & Francis: London.

[7] Theng, Y.L., Rigny, C., Thimbleby, H. and Jones, M.(1996), "Cognitive task graphs and executable user models for better hypertext," *APCHI96,* Singapore.

[8] Thimbleby, H. (1995), "Hypertext authoring without getting lost," *HCI'95 Adjunct Proceedings*, pp. 118-124.

Cognitive user models as design aids

Cécile Rigny, Yin Leng Theng & Harold Thimbleby

Middlesex University
Bounds Green Road
London N11 2NQ

Tel: *+44 181 362 5000*
Fax: *+44 181 362 6411*
EMail: *{cecile1, yin2, harold}@mdx.ac.uk*

In this paper we present an executable cognitive user model for browsing hypertext systems which enables designers to simulate the users' activity and perform usability measures for various types of users without requiring the attendance of real users. Those measures are particularly useful to test prototypes of the system. Furthermore, they can help designers to significantly shorten the time spent to evaluate the system. Preliminary results are presented. The advantages and disadvantages are discussed.

Keywords: usability, hypertext systems, cognitive user models, simulation, interactive system design.

1. Introduction

Designing effective and usable interactive systems is difficult. It involves a great variety of methods: defining the situation of concern; analysing users; eliciting requirements; building and testing prototypes; etc. It is crucial that designers perform usability analyses of their system (or prototype) to find out whether it matches users' requirements or not. Analytical methods allow only partial evaluation of the design. Designers need therefore to supplement the methods with empirical techniques, in which prototypes are built and tested. However, conducting user evaluations can be time-consuming [3]. They take time to organise, conduct, analyse and write up, and may not provide reliable data anyway. Before tests can be run and results analysed, usability measures must be identified, prototypes must be ready, users must be selected, and scenarios must be defined. Designers need help to shorten the

evaluation process to meet tight deadlines. One way to help designers consists of developing computer-based user models [2, 3].

In this paper, we describe a cognitive user model which allows simulations of interaction between users and hypertext systems. It is one way to help designers. Simulations enable designers to get rapid usability measures for various types of users [5].

2. The executable cognitive user model

The executable cognitive user model is built with a tool called CUM-DesTool (Cognitive User Model Designing Tool), which allows designers to build rapidly a set of user models tackling the variety of potential users based on a multi-module structure. Such simulations provide designers useful insights about usability aspects of the system such as ease of learning, users' performance, cognitive workload, etc. The objective is also to help designers determine whether the structure of the hypertext system matches users' mental models.

The cognitive user model itself can be partitioned into five components (as shown in fig. 1) and is represented in CUM-DesTool as a set of interconnected modules.

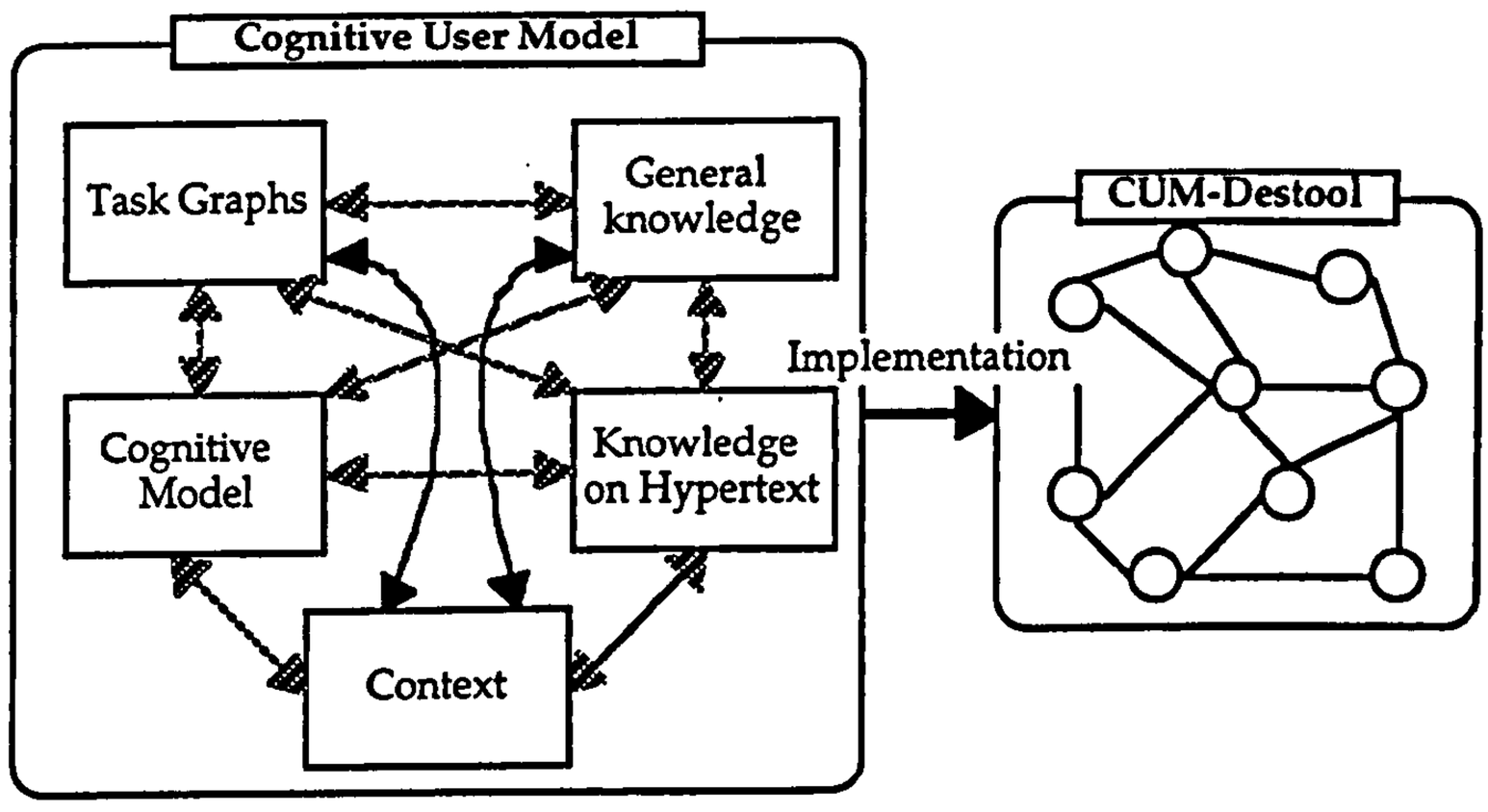


Fig. 1: The cognitive user model

The task graph is based on Polson and Lewis' Cognitive Walkthrough [7], and describes the user's activity in terms of tasks performed to achieve a particular goal.

The general cognitive model refers to the ACT model of human information processing, which assumes that memory can be partitioned into a short-term memory and a long-term memory [1]. It also assumes that human knowledge is distributed over a large semantic network.

The user's activity, e.g. when browsing hypertext systems, is, we shall assume, for the most part goal-oriented. Fig. 2 describes the cognitive processes which were incorporated into the user model. Users can identify a set of actions to achieve their goal and then select one action from this set of actions. If the initial goal is satisfied after performing for instance "action 1", a new goal can be activated, or the user can stop browsing the hypertext system. If the goal is not satisfied, users can perform another action from the initial set of actions or can be disrupted from their initial goal and then, a new goal can be activated. The overall strategy which determines which action will be selected or whether there is disruption from the initial goal or not depends mainly on:

- users' general knowledge in hypertext;
- users' knowledge in the contents of the hypertext system they are browsing; and
- the context in which the user's activity is performed.

The first two points determine the users' profile. The third point refers to the "situated actions" described by Suchman who draws attention to the way the surrounding circumstances can affect the course of action [4].

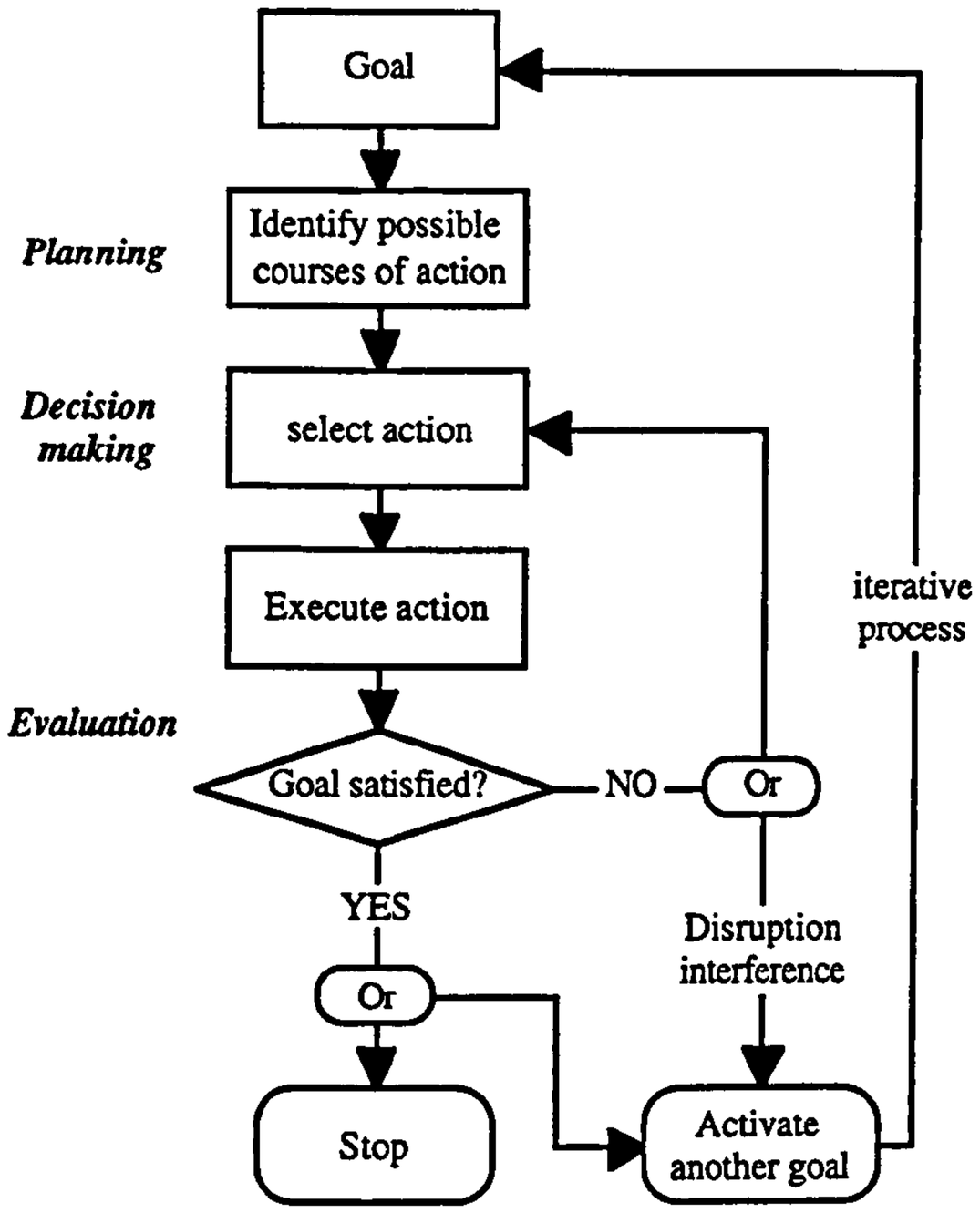


Fig. 2: The cognitive model

3. Simulation description, metrics and preliminary results

A "typical" hypertext user was modelled and implemented using CUM-DesTool. By this, we referred to a user who has already been browsing a few hypertext systems, but whose experience is limited. This user has thus constructed a simplified mental representation of common structures and interfaces of hypertext systems. The task graph and cognitive model for browsing hypertext systems have also been implemented into the user model. Fig 3 shows some of the main interfaces of the executable cognitive user model for browsing provided by CUM-DesTool. Concurrently, a hypertext prototype was built. Then we conducted the following experiment. The hypertext prototype was evaluated with one real user. The user's feedback was recorded and analysed. Then, we carried out a hand-simulation of the user's behaviour performing the same tasks using the cognitive user model. Similarly, the results provided were analysed and compared to the results obtained with the real user testing.

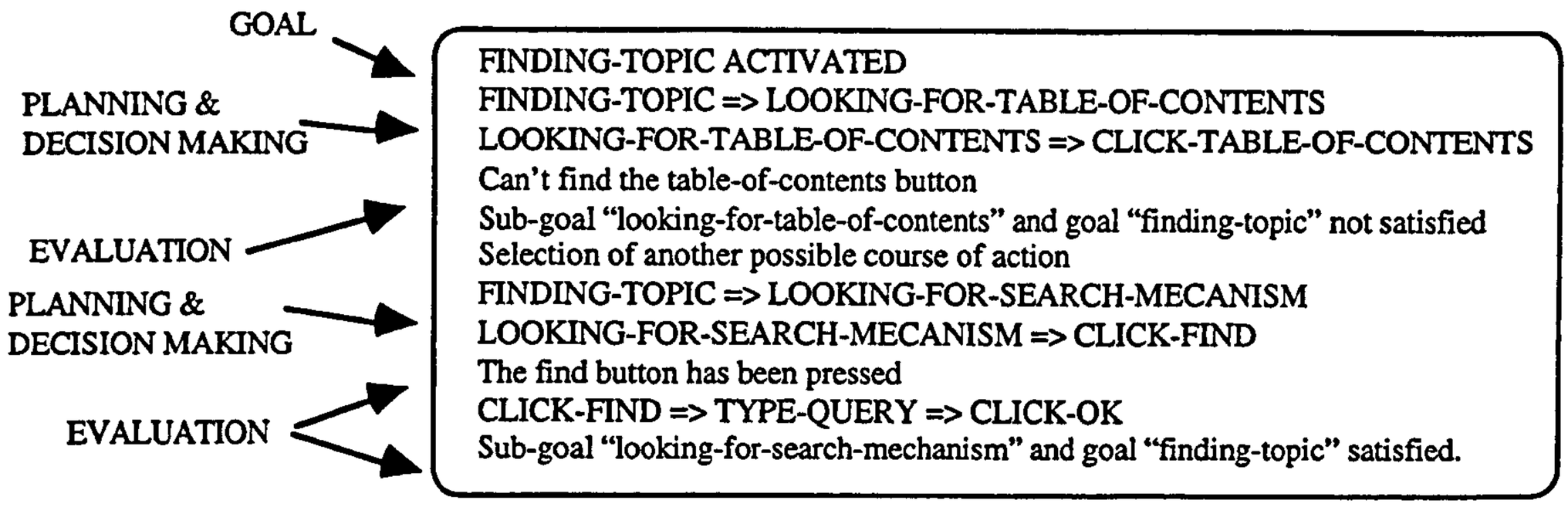


Fig. 3: Trace of the reasoning mechanisms

The user's behaviour when looking for a particular topic was simulated. Thus, the concept "finding-topic" (see alphabetical list of concepts displayed in fig. 4) was activated. The trace of the reasoning mechanisms and the sequence of actions taken are stored by CUM-DesTool. Fig. 3 shows a simplified sample of the data provided by CUM-DesTool trace function.

Such data are found useful to understand the user's behaviour and highlight possible design flaws. Several metrics have been identified to help designers analyse these data, for instance:

- number of actions taken to achieve one goal like "finding a particular topic";
- number of time a button (or menu) has been activated;
- number of goals which could not be achieved;
- number of goals which were satisfied;
- number of documents browsed; and
- number of visited hypertext nodes.

On-going work is being carried out to implement these measures which will be automatically performed by the cognitive user model. By using these measures, designers will be able to correlate user-dependant parameters, such as ease of use, with interaction metrics for various types of users. They facilitate and shorten the evaluation of the system and raise questions to be addressed by designers. It is also possible to generate a set of user models by modifying only a few parameters (e.g. weights of links).

Even though the simulation was carried out with a first-cut cognitive user model, some of the design flaws which were pointed out by real user testing were found during the simulation experiment, for instance having a table of contents button would help the search of some topics. However, the user model cannot tell designers whether the screen size is appropriate, for instance.

We have done exploratory work using CUM-DesTool to generate a particular user model. We suspect that a significant saving of time should be gained by using a suitable set of cognitive user models, rather than a unitary model, although this has not been fully investigated yet.

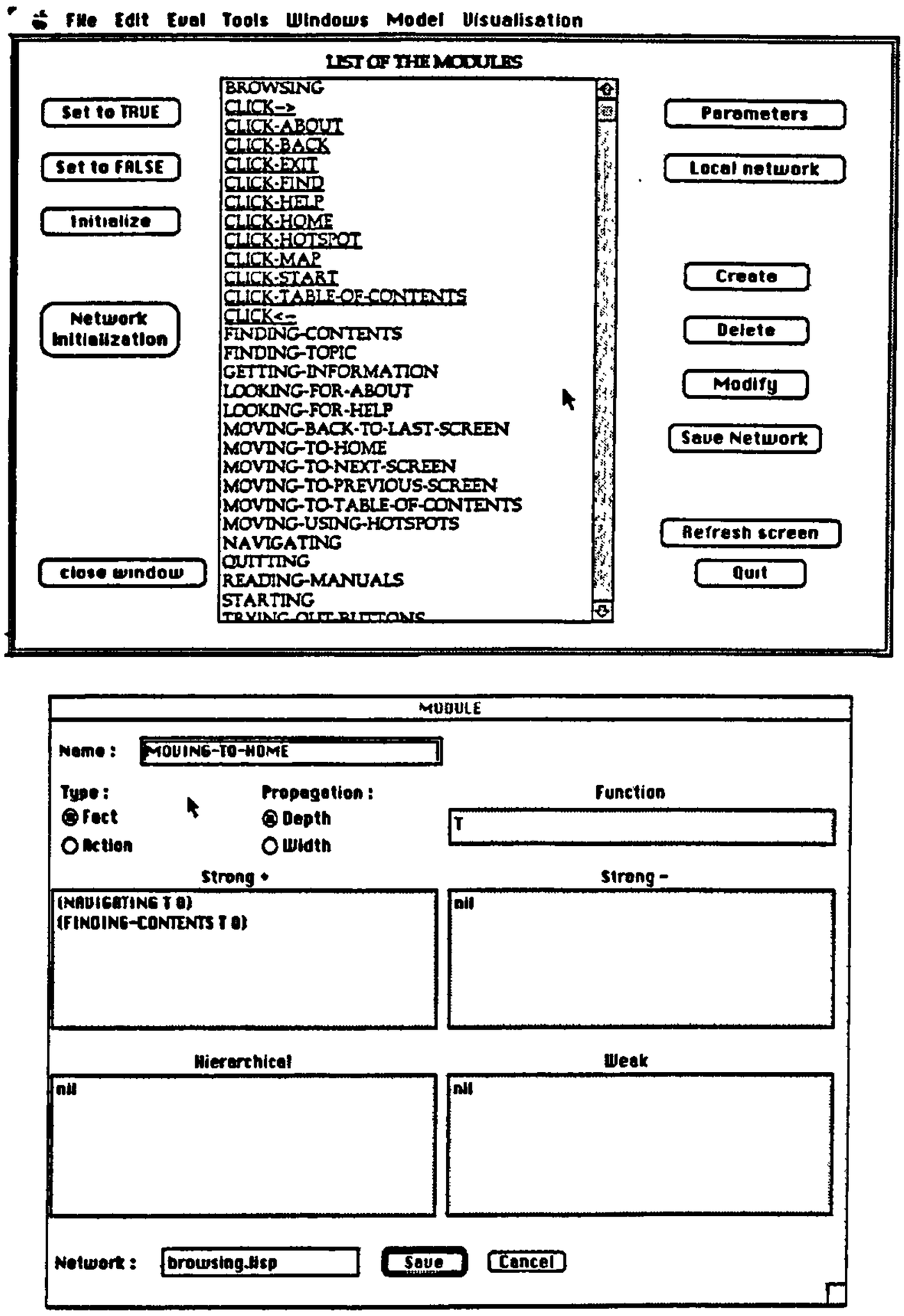


Fig. 4: Description screens

3. Conclusions

In this paper, we have shown through a hypertext example that simulating users' behaviour can help designers evaluate their systems without requiring user attendance. Moreover, we have shown that it is possible to identify and to get metrics on user performance, cognitive workload and satisfaction. Although those measures provide coarse approximations of usability factors, they can be easily and rapidly performed by an executable cognitive user model. They enable designers to point out some of the weaknesses of their systems and should reduce the length of the evaluation phase. On-going work is being carried out to extend the experiments to other types of users, depending on their knowledge in hypertext and to investigate the time saving compared to real user testing. The new results will be reported at the conference.

Acknowledgements

The authors would like to acknowledge the contribution of Ann Blandford, Matthew Jones and Thomas Green in the writing of the earlier versions of this paper.

References

[1] Anderson J. R. (1983). The architecture of cognition. Harvard University Press: Cambridge.

[2] Bass E. J., Baxter G. D. & Ritter F. E. (1996). Creating models to control simulations: a generic approach. To appear in AI and Simulation and Behaviour Quarterly.

[3] Kieras D. E., Wood S. D., Abotel K. & Hornof A. (1995). GLEAN: a computer-based tool for rapid GOMS model usability evaluation of user interface designs. *Proceedings of UIST'95.*

[4] Ritter F. E. & Major N. P. (1995). Useful mechanisms for developing simulations for cognitive models. *Proceedings of AISBQ spring 1995.*

[5] Suchman L. (1987). Plans and situated actions. Cambridge University Press: Cambridge.

[6] Theng Y. L., Jones M. & Thimbleby H. (1995). reducing information overload: a comparative study of hypertext systems. *IEE Colloquium on Information Overload*, Digest No. 95/523.

[7] Wharton C., Rieman J. & Polson P. (1994). the Cognitive Walkthrough method: a practitioner's guide. In J. Nielson and R. Mack, Eds. Usability Inspection Methods. pp. 105–140. Wiley: New York.

Is "lost in hyperspace" lost in controversy?

Harold Thimbleby, Matthew Jones and Yin Leng Theng
School of Computing Science, Middlesex University (U.K.)
Email: yin2mdx.ac.uk

1.0 Introduction

Hypertext technology still continues to excite many, ever since it became popular in the late 1980s. Many people then were sceptical that the technology might just be a passing fad. However, there are some who think that the promise behind hypertext is too fundamental to disappear quickly, and there are reasons to believe that hypertext technology promises something special. In hypertext, users not only benefit from the information they read but also from the richness of associations supported by the network of nodes and links. Hypertext has affected us directly, or indirectly in almost every facet of our lives, ranging from scientific work to business and education needs, to our general way of life. Take for example the World Wide Web (WWW) on the Internet. By the end of 1994, it has an estimate of 30 million users (Nielsen, 1995). However, hypertext is not a pancrea to life's problems (Nielsen, 1995). Associated with hypertext are two classes of problems (Conklin, 1987): problems with current implementations, which include delays in the display of referenced materials, deficiencies in browsers, etc.; and secondly, problems that seem endemic to hypertext such as cognitive overload and disorientation. Cognitive overload is the additional effort and concentration necessary to maintain several tasks or trails at one time. Disorientation is the tendency of users to lose their way in non-linear information. This is commonly referred to as the "lost in hyperspace" (LIH) problem.

Ironically, the LIH problem has given rise to much controversy itself. Some think that the LIH problem is one of the most difficult issues in hypertext research and there is yet more to be done to ameliorate this problem. However, there are others who believe it is not a significant problem and feel that efforts should be channelled to address other more pressing issues. In this paper, we want to re-examine the LIH problem and question whether LIH is a significant problem that still warrants the attention of the research community. For conciseness, we will call multimedia, hypermedia and the WWW "hypertext systems" since the issues surrounding the LIH problem with which this paper is concerned with apply to all of them. In this paper, the term "hypertext" is used to denote a hypertext document that is made up of interlinked pages or nodes. Whereas the term "hypertext system" refers to a set of software tools used to create a hypertext.

2.0 Controversy surrounding the significance of the "lost-in-hyperspace" problem

In a workshop on "The Missing Link: Hypermedia Usability Research and The Web" (1996) held at the Open University, some HCI researchers and practitioners felt that there are more important and pressing issues besides the LIH problem on the Web, and hypertexts in general, that need to be addressed. This came about because of the results reported in the 4th WWW User Survey by the Graphic, Visualisation and Usability Center at Georgia Tech Research Corporation conducted from 10 October through November 1995 (Pitkow and Kehoe, 1995): it was reported from a sample size of more than 23 000 responses that users were not "lost" and the classical LIH problem was not a problem (6.5%), as opposed to the most widely cited problem that it takes too long to view/download pages (69.1%).

If the LIH problem refers to "users not able to determine where they are" as reported in the survey, then perhaps it may not be a pressing issue. But 6.5% of the user population of approximately 30 million in 1994 on the WWW who reported being "lost," is certainly not a small number. The smallest of usability problems, when multiplied across thousands or millions of users, becomes a source of massive inefficiency and untold frustration (Nielsen, 1993). The LIH phenomenon in our view, however, can refer to any of the following conditions: users cannot identify where they are; users cannot return to previously visited information; users cannot go to information believed to exist; users cannot remember what they have covered; and users cannot remember the key points covered (Conklin, 1987; Mcknight, Dillon and Richardson, 1991; Nielsen , 1995; *etc.*). In the same survey, problems such as these: not able to find a page that they know is out there (34.5%); not being able to find a page once visited (23.7%); and not being able to visualise where they have been and where they can go (14.3%) were identified as "real" problems. This constitutes an enormous number of users on the WWW, who might not report that they were "lost" but

experienced different forms and degrees of "lostness". These findings give a snapshot of the current WWW user population, and the problems experienced by users on the WWW. Users really are lost. It is not just a superficial disorientation. If users are frequently lost, they will become frustrated and this may influence the way they interact with the hypertext. Worse still, they may cease to use the system because they may feel that they are wasting their time and overlooking crucial information. This is certainly not desirable as the primary objective of hypertext is to provide users with information!

3.0 Addressing the "lost in hyperspace" problem

Most people think the LIH is a *user's problem*, resulting in improvements being sought in the presentation of information. Not surprisingly, therefore, that much research solutions involve the use of graphical browsers and query/search mechanisms. They seem to make the following assumptions referenced to the *users'* supposed failings (Theng, Thimbleby and Jones, 1996): *users* have a wrong or incomplete conceptual model; *users* lack experience in using hypertext for performing tasks such as browsing; *users* are distracted because of the "embedded digression problem"; and *users* don't understand the chosen display conventions. And because hypertexts are difficult to build, many commentators are happy to seek solutions in better understanding *users* and helping *them* cope. There has, of course, been some success in this approach — maps, virtual reality visualisation — that no doubt seems to confirm it!

We argue that perhaps wrong or inappropriate solutions are being sought because incorrect or incomplete assumptions are made. Hypertext design is hard, and hypertexts are used less effectively than we would wish. The question we want to ask is: Is LIH primarily psychological or engineering? The answer to this question will have serious implications on the solutions being sought to address the LIH problem. If LIH is a *psychological* problem, then the problem may be entirely due to users' inability to exploit computer screens, complex information structures, and that nothing in the design is going to ameliorate this.

Though disorientation can arise in conceptual space (within the user's mind), which most research findings support — we argue that research should not rest on users alone! The LIH problem may not just be a *psychological* problem — it may also be an *engineering* problem. This implies that LIH is perhaps attributable to bad system design, and poor design causes psychological problems too. We agree with Mayes *et al* (1990) that addressing the LIH problem within hypertext goes beyond providing more and more navigational aids.

Could it be possible that because hypertext authors themselves are "lost" in the process of designing and authoring hypertexts, they inadvertently contribute to poorly designed hypertexts, which in turn leads users often being LIH? How can we more quickly find better organisational principles for hypertext? We argue for a move away from treatment to prevention, from treating the user's symptoms — themselves a reaction to bad design — to avoiding the bad design (Theng, Thimbleby and Jones, 1996).We need to re-examine the way hypertexts are designed and built. We need to examine fundamentally how information should be structured and displayed. We should not assume that if certain design features work well for some information contents and purposes, it will be appropriate for others.

4.0 Conclusion and current work

Although much research effort has been invested to address the LIH problem, it still remains unsolved and certainly merits further investigation. We argued that the LIH problem is not just a psychological problem, it is also an engineering problem. We are currently implementing a practical authoring tool to help designers manage the complexity of the design and validation processes without themselves getting "lost", which in turn produces better, usable hypertexts so that users will not experience "lostness" when navigating through them.

References

Conklin, J. (1987), "Hypertext : An Introduction and Survey," *IEEE Computer*, pp. 17-41.

Mayes, T., Kibby, M. and Anderson, T. (1990), "Learning about learning from hypertext," In Jonassen, D.H. and Mandi, H. (Eds), *Designing hypermedia for learning*, Springer-Vering, pp. 13.1-13.24.

McKnight, C., Dillon, A. and Richardson, J. (1991), *Hypertext in Context*, Cambridge University Press, U.K., pp. 64-104.

Nielsen, J. (1995), *Multimedia and Hypertext: The Internet and Beyond*, AP Professional.

Nielsen, J. (1993), *Usability Engineering*, Academic Press.

Pitkow, J. and Kehoe, C. (1995), *GVU's WWW 4th User Surveys,* http://www.cc.gatech.edu/gvu/user_surveys/survey-10-1995/

Theng, Y.L., Thimbleby, H. and Jones, M. (1996), "'Lost in hyperspace': Psychological problem or bad design?", *APCHI'96*, Singapore.

Thimbleby, H. (1995), "Authoring consistent hypermedia without getting lost," *HCI'95 Adjunct Proceedings*, pp. 118-124.

HyperAT: HCI and Web Authoring

Yin Leng Theng, Cécile Rigny, Harold Thimbleby and Matthew Jones
School of Computing Science, Middlesex University (U.K.)

Abstract
We review HCI problems with hypertext, and for authoring World Wide Web documents in particular. We suggest that a framework is required to understand the usability issues, and that these issues cannot be seen as pyschological or computing: they are multi-disciplinary. We discuss HyperAT, a prototype authoring tool, being implemented to test these ideas.

Keywords: "lost in hyperspace", authoring tool, World Wide Web, multi-disciplinary approach

1.0 The World Wide Web and its problems

When reading or writing a book, the user (reader or author) can use an algorithm for completing their task — for instance, start at page 1, process it, turn to next page, and so on, then stop on the final page. In contrast, there is no algorithm for reading or writing an arbitrary hypertext document that guarantees completion of, or even uniform progress during the user's task. In general, any non-trivial task involving hypertext is impossible to do well, unless computer support manages the task in such a way that a sense of direction can be provided. But this is rarely possible, either because the computer does not know enough about the task, or because the hypertext structure is unknown (as on the World Wide Web). Without a 'sense of progress' a user will never be certain when they are able to stop or when their task is completed, or if they pause, how to resume without repetition; there may always be other pages or other links in the document that need considering. There is a wide range of literature on this topic (e.g., Cockburn and Jones, 1995), though mostly concerned with users' behaviour and performance rather than the causes. For the purposes of this paper, we shall call the problem 'lost in hyperspace' (see Thimbleby, Jones and Theng, 1997 for more details). This paper will first review the problem, survey of solutions to the problem and then discuss an engineering approach to it.

When the World Wide Web (WWW) was developed in 1991, the intention was to link a select group of users such as physicists and engineers at different sites. In three years, it had an estimated 30 million users (Nielsen, 1995a). Today, the WWW is used by millions of users all across the world. It has affected us directly, or indirectly in almost every facet of our lives, ranging from scientific work to business and education needs. The WWW has changed the Internet to the extent that it has become almost synonymous with the modern use of the Internet.

This paper concentrates on the WWW because it is the largest hypertext ever, and any usability issues are 'scaled up,' affecting millions of users (Maurer, 1996). This view was supported by the results of the 4th WWW User Survey by the Graphic, Visualisation and Usability Center conducted over October/November 1995 (Pitkow and Kehoe, 1995). From a sample size of more than 23 000, the report showed that users suffered different forms and degrees of "lostness": not being able to find a page they know is out there (34.5%); not being able to find a page once visited (23.7%); not being able to visualise they have been and where they can go (14.3%); and not being able to determine where they are (6.5%).

2.0 Survey of solutions to address the LIH problem on the World Wide Web

Much work has been done to address the "lost in hyperspace" problem on the WWW. Some solutions are aimed at helping hypertext users, others are aimed at helping hypertext designers.

2.1 For the hypertext users

- *Better navigation support mechanisms*. Nielsen (1995b) lists eight navigation support mechanisms that had been implemented in Netscape Navigator, the most popular WWW browser, to help user navigation: (1) using a standard URL notation to go to an absolute address; (2) indicating hypertext links with underlined text or figure; (3) allowing users to return to previously visited nodes using a backtracking feature; (4) allowing users to build a set of direct jumps to favourite places in hyperspace using bookmark; (5) generating a history list to allow users to go back to a list of visited nodes; (6) changing colour of underlined text once the users have seen the destination node it points to; (7) showing prospective view in the footer *before* the user makes the jump; and (8) providing landmark like "Home page" or "What's new?". Maurer (1996) suggests having overview documents to represent the structure of the hyperweb, containing a list of links to other documents, an annotated diagram, map, etc. Cockburn and Jones (1995) propose building a graphical browser that dynamically adapts to, and reinforces, users' browsing actions and users' mental models. Dynamically generated structure maps in the form of graphical browsers are also suggested: global maps show the entire hyperspace; local maps show the "vicinity" of the current node in terms of hyperlinks to and from

other related nodes; and fisheye views focus attention on important nodes by deliberately distorting the view.

- *Search and linking facilities*. Sophisticated search facilities such as keyword search, content search and fuzzy (inexact) search are indispensable for finding specific information once the size of the hyperweb exceeds browsable proportions (Maurer, 1996). Work done includes automating indexes (such as web robots or spiders) to walk the entire server tree.

- *Better adaptive and adaptable facilities*. The lack of support for typed nodes and links limits the richness of information which can be represented. A project, called MacWeb undertaken by Nanard and Nanard (1993), draws upon knowledge-based approaches to address the LIH problem, by extending the hypertext metaphor with typed links and typed nodes to represent knowledge in the hypertext as a semantic network (Clibbon and Callaghan, 1996). This solution provides great potential for building adaptive and adaptable hypertexts, taking into consideration users' needs and browsing patterns. Users can navigate round hypertexts more efficiently with a reduced chance of getting LIH.

2.2 For the hypertext designers

- *More comprehensive style guides*. Style guides describe the design principles and guidelines used to create hyperdocuments on the WWW. Many style guides have been written to help designers produce better, usable hyperdocuments. According to Tilton (1996), users' perception and assumptions about the organisation of the websites can have a major impact on the usability of the page and site design. Therefore, designers need to give users a feeling of knowing where they are. Tim (1995) suggests structuring hyperdocuments using a tree structure, and designers can use this structure to organise files into directories. To help users identify the origin and relationships of WWW pages, consistent and predictable WWW pages should be produced. The essential elements that should appear on each WWW page are (Tim, 1995; Lynch, 1995; Thimbleby, 1995, Tilton, 1996, *etc.*): (i) a meaningful title to occur at the head of the document to identify the content of the document in a fairly wide context; (ii) text-labelled buttons to provide fixed links between a series of pages to bind them into a document, e.g., "Previous", "Next", "Home", Table of contents" buttons, *etc.*; (iii) links to other related pages in the local WWW site; and (iv) page footer to identify the origin, authorship, author contact information, copyright statement, date of creation and modification. In addition, other elements that are crucial for good WWW page writing include: writing device-independent HTML codes; combining all the WWW pages into a single document for easier printing; keeping language simple and clear; keeping typographical styles to a minimum; putting in links to explain themselves so that users know where they are going; and keeping WWW pages short ranging from half a A4 page to 5 pages, since scrolling pages can be particularly disorientating as users move through long HTML pages.

- *More powerful programming languages*. All browsers display pages written in HTML. However, HTML does not allow authors to have much control over page and presentation layouts. While HTML provides information about content, style sheets consist of style rules that tell a WWW browser how to present a hyperdocument (Pozadzides and Quinn, 1997). At the time of this writing, Microsoft Internet Explorer 3.0 and 3.01 are the only browsers supporting Cascading Style Sheets, which means that several different style sheets, each with a different order of importance, are combined in order of importance to create a presentation style (Tilton, 1996). Though style sheets are a new development on the WWW and currently are not widely used, Nielsen (1997) predicts that they are the only solution to getting nice presentation with ever-increasing numbers of browsers and display devices. JavaScript, a programming language from Netscape incorporated in their browsers, is also gaining great popularity because users are attracted to fanciful, animation features it can produce on the WWW. There are several features JavaScript can provide for existing websites (Harold, 1996): letting server draw pictures in a window on the client; using graphics primitives to create desired WWW page, putting less load on the server; and allowing more user interaction. Recognising the potential in JavaScript and Style Sheets, the WWW Consortium has defined a new standard called JavaScript Style Sheets.

- *More systematic testing methods*. Tim (1995) advocates carrying out testing on hyperdocuments to ensure that they are well-designed and well-structured. Designers should always proof-read hyperdocuments to avoid making "silly" spelling mistakes. Designers should test-run the hyperdocument using several different client programs to ensure that it has been coded in a device-independent way. Use the server log files to monitor the readership of the hyperdocument. Another way to test the hyperdocuments is to invite feedback from readers.

- *More efficient authoring tools*. HTML documents can be written in any text editor. Many authoring tools have been developed, which can broadly be categorised into commercial tools (e.g., HoTMetaL, Netscape Gold, Front Page, etc.), and research tools. All these efforts suggest that there is a need to

provide better editing facilities in authoring tools to help designers produce well-structured hyperdocuments.

So far the solutions discussed above are aimed at improving the performance of the WWW. Due to the exponential growth in WWW usage as well as an avalanche of servers, documents and hyperlinks, there is a grave concern as to how websites can be efficiently maintained. Hence, alternative solutions to the WWW are sought. A well-known example is the development of Hyper-G at the Graz University of Technology. Hyper-G, a second-generation hypermedia information system released in 1994, tries to combine the advantages of the WWW, WAIS (Wide Area Information Service), and Gopher while minimising their disadvantages. Hyper-G claims to have overcome some of the shortcomings of the WWW (Maurer, 1996). Another solution is the development of Microcosm, an open hypermedia system, by the University of Southampton. Microcosm does not suffer from some of the problems of the WWW and has been applied successfully to unstructured hypertexts on the WWW (Hall, Carr and Roure, 1994). The flexibility of Microcosm separating the link structure from the data in the system to enable separate link and data processing, makes authoring easier for designers.

3.0 Concrete proposals

Although much work has been done to tackle the LIH problem, it still exists (Thimbleby, Jones and Theng, 1997). *Ad hoc* methods of designing, constructing and validating hypertexts are not enough. If users get "lost" in hypertext, designers do too. This suggests that tools for designing hypertext should provide improved support for designers. Nielsen (1996) predicts that due to a change in the dominating styles for websites over recent years, a real HCI contribution essential for web design should consist of further research into these different knowledge areas: (i) knowledge of icon design; (ii) knowledge elicitation to discover appropriate information space structures; (iii) usability testing; and (iv) task analysis techniques. We agree that this is the way forward if the performance of the WWW is to be enhanced. But searching for solutions in isolated disciplines, and recommending them to designers in the hope that they would somehow remember to put them into practice, may not be as simple as it sounds. In practice, many factors could have prevented well-intentioned designers to put these good suggestions into practice. One of which could be that designers may be too overwhelmed, and/or may seemingly do not have the time and capacity to attend to all these authoring details. In order for Nielsen's suggestions to be truly effective and implementable, we should go beyond just providing designers with a list of do's and don'ts. Designers need authoring help. If some of these ideas could be automated so that designers need not worry about their implementation, chances are that better hyperdocuments could be produced since designers would be freed to concentrate on other critical issues that cannot be automated, but require sound human judgement and expertise.

This paper addresses the LIH problem in the WWW by taking a different stance, that is, integrating *proactive, multi-disciplinary* approaches to address the LIH problem with the emphasis of doing things right from the start. By integrating the approaches proposed below, a practical authoring tool called HyperAT for the design and building of usable hypertexts, is developed to test these ideas.

- *Need for good hypertext structure (Approach One).* Owing to the associative relationships that exist between nodes and links in hypertext, it is imperative that these relationships should be correctly captured and represented. Unlike books, there is no universally accepted organisational principles and structures in hypertext. Without which users will have to "guess" the structure of the hypertext, and try to understand what the interface wants to convey. The way the nodes and links within hypertext is structured is dependent upon the tasks users want to perform or try to perform, as well as the functional support provided by the hypertext. Many different approaches have been investigated to find out how best to structure information. Some researchers adopt a prescriptive approach by imposing a simple, regular structure on intractably complex information (e.g., Garzotto *et al* 1991, *etc*). These structures can be linear, hierarchical, and recursive. While hypertext is intrinsically non-linear, some hypertext authoring tools allow extensive use of linear structures, in the form of cards and stacks in HyperCard, or scrolling windows in Guide. Hierarchical structures have also been extensively used by many researchers to organise the contents of hypertexts (Smith and Newman, 1996), since users find them easy to understand (Garzotto *et al,* 1991). To some researchers, this may be forcing structure too early in the design process, which is not desirable (Halasz, 1987). However, a counter-argument is that divergence can be prevented in hypertext, normally the reason for users being LIH, and limitedness of the hypertext structure is a good way to do that (Am, 1994). Several researchers (e.g., Rada and Murphy 1992, *etc*) stressed the need for information in hypertext to be structured in such a way to support users' tasks. The types of information structures that have been investigated are hierarchical structures, network structures and a combination of both. Different information structures support different types of tasks. Mohageg (1992) found that trying to perform searching tasks in a network structure produced a negative effect on task performance. In fact, research has shown that "unfocused browsing or exploratory" tasks are best supported by a network or combination information structure, while "focused browsing or searching" tasks are best supported by a hierarchical information structure (Smith and Newman, 1996).

- *Need for good design guidelines and principles (Approach Two).* Good design takes into account characteristics of the intended users and the work that they do. Therefore, good computer systems are systems that are useful, usable and desirable, that is, people can easily learn, can do things they want to do because of the functions provided, and people like them. Interactive systems require iterative design. Since disorientation can occur in a spatial network of nodes and links, we want to re-look at design issues. We need to ensure that good hypertext design principles and guidelines are incorporated into the building of hypertext in the first place. Much of the work done so far in hypertext design concentrated only on the interface design issues. Design principles and guidelines should go beyond just providing an attractive interface. By "design interface issues," we refer to the information channel that allows the hypertext to explain the internal structure and representation of nodes and links to the user in the simplest and most effective way, and for the user to communicate his intentions and obtain the answer to his intentions. If users were to be helped in navigating hypertext, the user interface needs to be usable. On the other hand, users build models of what is happening in their minds, and they use these models called "mental models" in their interactions. To proceed with their interactions with the hypertext, users expect it to give them cues, otherwise they will need to rely on their prior experiences with hypertexts and computer systems. This in itself is not bad. However, there are occasions when users' prior experiences interfere with their understanding and interactions with the current hypertext they are navigating. To ensure that this does not take place, we should provide users with a clear and unambiguous user interface, reflecting accurately the underlying hypertext structure of nodes and links. Interface design principles applicable to hypertext authoring are: consistency of presentation; minimal mental overload for users; ease of learning and use; good conceptual model of users; well-structured network of nodes and links; and full and continuous feedback to users.

- *Need for an engineering, task-based approach to understand users' needs (Approach Three).* It is well-known that designers often design for themselves unless they are trained to realise that people are diverse, and that users are unlikely to be like them. Solutions to the LIH problem should then address the issues of helping users navigate through conceptual space. We need to have an accurate respresentation of users' behaviour and actions when they perform or try to perform common tasks such as browsing, information search, seeking references and recall. By trying to make sense of what users should do or what they actually do, hypertext authors will at least stand a better chance of producing user-centred hypertexts that will meet users' needs more effectively. Task analysis and cognitive user modelling techniques can be used to help us understand users' behaviour and actions, taking into consideration users' reasoning and learning processes (Theng, Rigny, Thimbleby and Jones, 1996).

4.0 HyperAT: Implementing the proposals

HyperAT stands for "Hypertext Authoring Tool". HyperAT is a prototype designer tool for authoring hypertext and WWW documents. It is implemented in Macintosh Common Lisp (version 3.9) for PowerPCs. We would like to emphasize that it is not the intention of HyperAT to provide a full range of editing facilities with attractive interface, and HyperAT does not claim in any way capable of competing with commercial tools in this aspect. However, being a research tool, HyperAT aims to investigate facilities not seen or fully exploited in popular commercial tools which we think are crucial in helping designers build more usable hypertexts. HyperAT aims to address the LIH problem by helping designers manage the complexity of the design process without themselves getting "lost", and users navigating the hyperdocuments produced by HyperAT without feeling "lost". We see HyperAT contributing in these areas:

- HyperAT is a practical authoring tool to help hypertext designers build usable hypertexts.
- HyperAT is an experiment in collaborative efforts involving many disciplines.
- HyperAT is an analytical research tool for contextualising and delivering the results of hypertext usability to hypertext designers.
- HyperAT is a preliminary and innovative investigation in cognitive user modelling minimalism in hypertext authoring.

4.1 General overview

Figure 1 gives a general overview of HyperAT, its inputs and outputs. Inputs refer to the multi-disciplinary approaches that underlie the design of the authoring and usability components that made up HyperAT. Because the WWW is a special hypertext, these approaches had to be adapted for use on the WWW. Approach One stresses good WWW page structure to help both designers and users. Approach Two examines design guidelines and principles, adapted to good WWW style guides. Approach Three emphasizes the importance of understanding users' browsing needs and the tasks they perform. Outputs are the deliverables produced by HyperAT. Besides providing the basic authoring facilities to produce WWW pages, HyperAT also delivers usability results to designers regarding any usability problems that might be detected during its analysis.

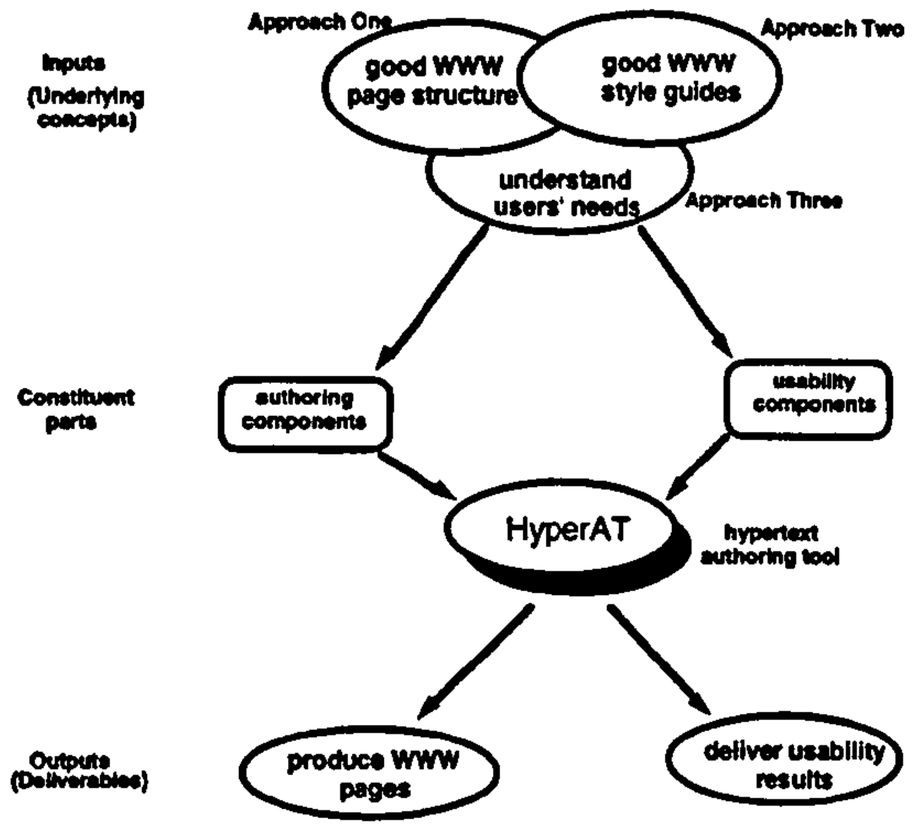


Figure 1. General overview of HyperAT, its inputs and outputs

An understanding of HCI elements essential for any interactive systems is crucial in designing successful interactive systems. In designing the authoring components, we incorporated two underlying design concepts, that is, the need to impose a structure, and the need to incorporate good WWW style guidelines and principles. For the usability components, we incorporated features that help designers to better understand users and their browsing behaviour.

4.2 Authoring components

The main objective of HyperAT is to help designers build usable, well-structured hyperdocuments. By that, we refer to a hyperdocument with the following characteristics: (i) no links that go nowhere; (ii) no nodes that are not linked; and (iii) minimum number of links traversed to reach required nodes. The authoring components in HyperAT provide the basic authoring environment for the creation, loading and modification of hyperdocuments. HyperAT's facilities are accessed using a graphical, user-based interface. Hyperdocuments are created via a form-like screen and converted into predetermined HTML format, which can be displayed on the WWW using a WWW browser. Designers can also display graphically both global and local views of the structure of hyperdocuments created. Hard copies of the hypertext structure and the associated HTML coding can be printed and kept for documentation as well as for maintenance purposes. A HTML-editor is also incorporated to provide designers with a menu to write HTML codes, without designers having to memorise the syntax.

Imposing a structure (Approach One)

Despite its broad appeal, one of the limitations of the WWW is that there is no information structuring facilities beyond hyperlinks (Maurer, 1996). Because hierarchies are easily understood and used by both hypertext designers and readers (Tim, 1995, Lynch, 1995), we had incorporated into HyperAT quasi-hierarchical structures as framework for capturing information. HyperAT captures node relationships using a simple parent-child analogy. This simple way of representing node relationships is not only intuitive to designers but powerful in constructing data structures. Node relationships are expressed in terms of the associations between nodes. There are two common kinds of associations between nodes and they are represented in terms of hierarchical and cross-referenced links.

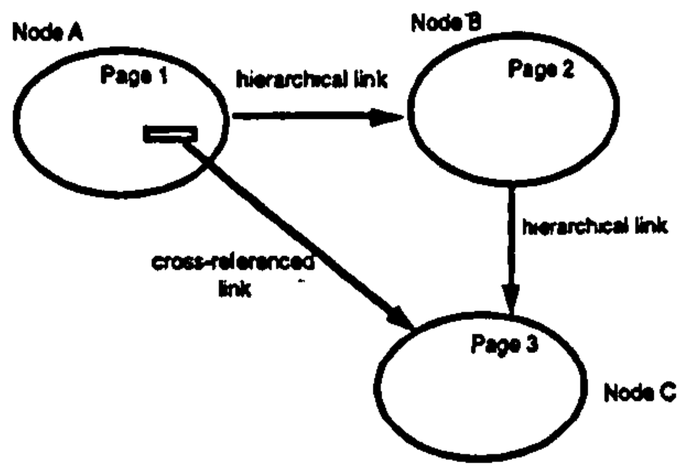


Figure 2. Relationship between nodes A, B and C

Representing these links in HyperAT is simple. Using Figure 2 as an example, the relationships among nodes A, B and C are described by these facts:

- *Fact 1*: "node A is hierarchically linked to node B"
- *Fact 2*: "node A is referentially linked to node C"
- *Fact 3*: "node B is hierarchically linked to node C"

To input information about the nodes, a form-like screen is used to capture information about the nodes. In HyperAT, a node refers to a unit of information representing a WWW page. Each page has the following attributes: file name refers to the name of the HTML document to be created; window name refers to the title of a WWW page; node name refers to the name of a WWW page; icon name refers to any icon file associated with a WWW page; node text refers to the body of text which may/may not contain hotspots or cross-referenced links in a WWW page; neighbour nodes refer to WWW pages referenced by cross-referenced links in the node text; and parent node name refers to the parent of a WWW page. To represent the relationship between node A and node C, we enter into the node text a HTML <a href="c.html"> tag to indicate a cross-referenced link from node A to node C. Figure 3 shows the various input screens recording how these three facts are captured in HyperAT.

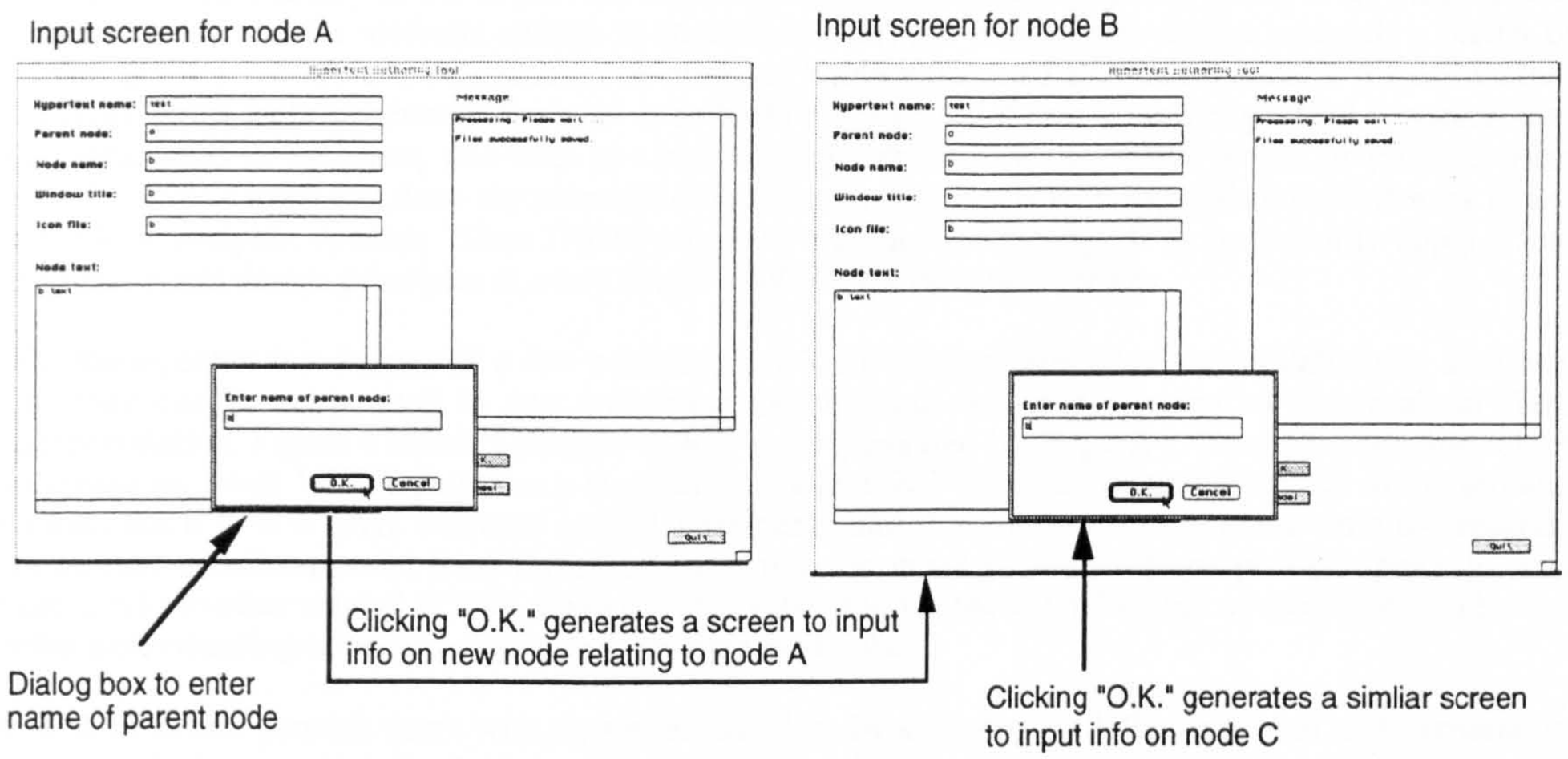


Figure 3. Screen shots to input information on nodes A, B and C

During the conversion of the hyperdocuments into HTML codes, HyperAT also generates a table of contents, a hierarchical representation of the structure of the hyperdocuments, accessible from every page of the hyperdocument, using the "contents" button (Figure 4). A fisheye view of related pages with respect to users' current page, is also provided to help users better understand the structure of the hyperdocument in relation to where they are, thus ameliorating the LIH phenomenon.

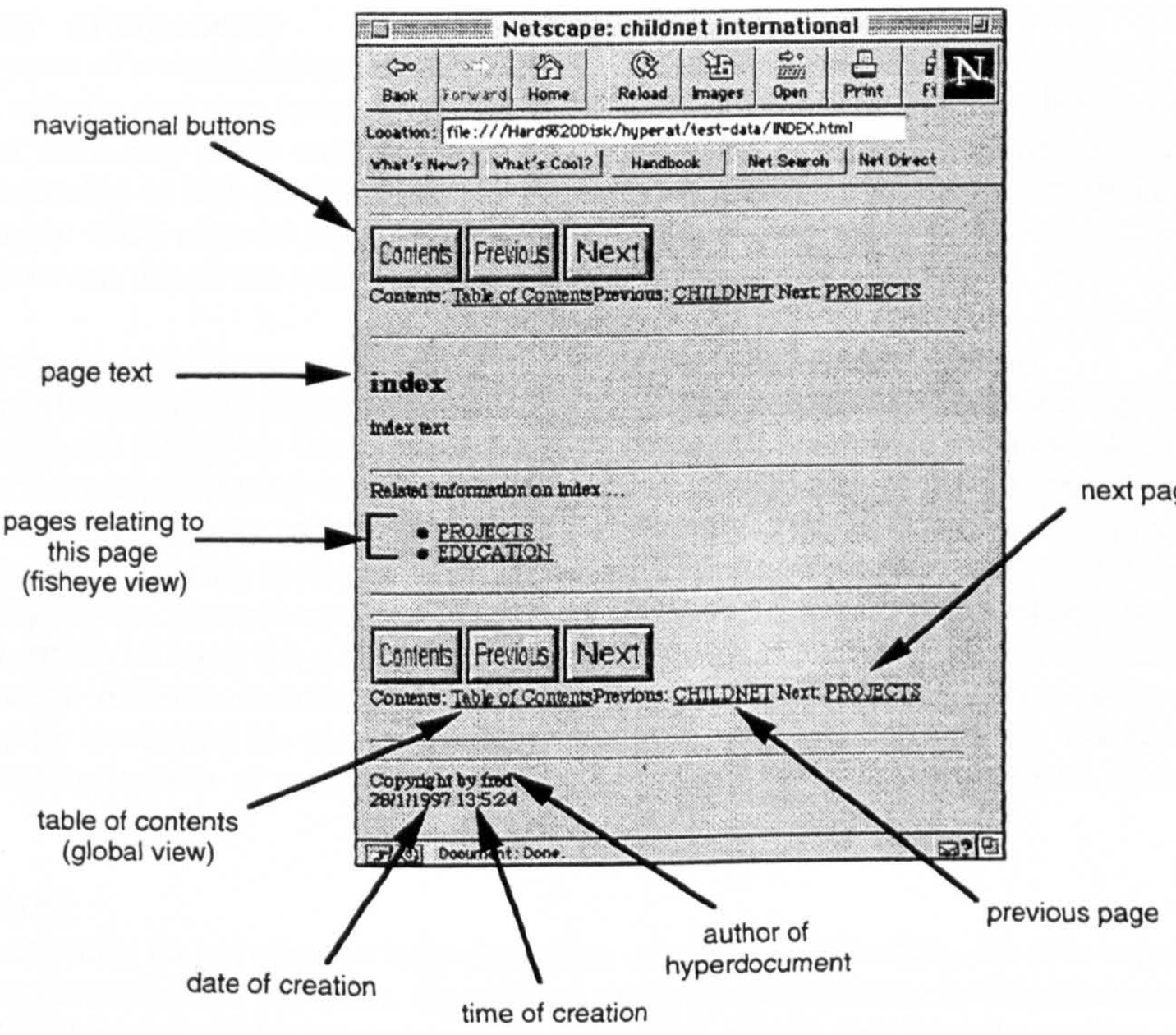


Figure 4. A sample WWW page generated by HyperAT

Besides providing within HyperAT's authoring environment an automated, hierarchical structuring feature to represent node relationships, we also incorporated other authoring aids. One is a generated trace of created nodes during a HyperAT session to provide useful memory jots for designers, who may be interrupted during the HyperAT session or are simply confused over the nodes created. Another is a generated global map showing the structure of the hyperdocument with its constituent nodes. Clicking onto a node will bring up another map, with that node as the root node, providing designers with a fisheye view cancelling off other details not related to it.

WWW guidelines and principles for better design (Approach Two)

Designing, structuring and maintaining websites is difficult. Not only should designers ensure that websites are structurally sound to prevent users from getting LIH while surfing the WWW, they have to create websites that are aesthetic enough to attract users. Proper WWW page design is largely a matter of balancing the structure and relationship of menu or home pages and individual content pages or other linked graphics and documents. The goal is to build a hierarchy of menus and pages that is natural and well-structured to the users, and does not interfere with the use of the WWW pages or mislead them (Lynch, 1995). Given that there are potential difficulties in creating WWW pages that are both easy to use and full of complex content, Tilton (1996) proposes that the best strategy is to consistently apply a few basic document design principles in every single WWW page designers create.

We implemented into HyperAT a few established website design principles and guidelines to illustrate that they can be automated in any authoring tool without designers having to worry about their implementation. Figure 4 shows a sample WWW page generated by HyperAT. To ensure consistency of presentation, every WWW page has a standard "look and feel" with navigational buttons at the top and bottom. Each WWW page contains essential elements like the title, author, date and time of creation, button bars which represent fixed links that allow users to move to content page, previous page, or next page. Links to other related WWW pages are also generated for each WWW page so that users can have a better understanding of how they can obtain related materials.

WWW browsers provide users with a prospective view by showing the URL with path and filename in the footer before users make the jump (Jones, 1996). Adopting this idea, HyperAT provides users with prospective information by generating the title of the pages users would move to if they were to click onto the navigational buttons. For example, "previous" and "next" buttons in Figure 4 indicate moving to pages named "Childnet" and "Projects" respectively. These names are more meaningful as they reflect titles of WWW pages, in contrast to the URL's way of naming of pathnames. Because these prospective views that accompany navigational buttons are not hard-coded but automatically generated by HyperAT, no extra effort is therefore required from designers to ensure the inclusion and maintenance of this feature.

4.3 Usability components

In the 'early' years of website design, efforts had been focused on hacking HTML as the main requirement for creating a website, and user interface design is often an afterthought. Nielsen (1996) predicts that web-surfing is dead, with only a few websites visited repeatedly by a substantial number of users. Owing to a change in relationship to web design, there is an increasing need to treat users as individuals rather than a nestful of hungry GET-request users. Usable WWW pages that subscribe to users' needs should be developed. However, this is not a simple task.

Understanding users' browsing pattern and needs (Approach Three)

Designers need authoring aids to help them understand users' needs and browsing behaviour. Tim (1995) suggests carrying out testing on the hyperdocuments produced even though testing takes time. However, the decision of how much testing designers do depends on the quality of the document designers wish to provide. Hence, apart from the basic editing facilities of create, edit and save, embodied within HyperAT is an experimental, authoring testbed which allows hypertext designers to carry out different modes of usability testing on the hyperdocuments created by HyperAT, all within the authoring environment of HyperAT: (1) structural analysis; (2) real user evaluation; and (3) executable user modelling. The ability to toggle between different modes makes testing less cumbersome, and hence more convenient for designers, thereby increasing the chance of creating more usable hyperdocuments. We have implemented the first and second modes of testing in HyperAT, and are exploring the potential of non-human user testing.

Structural analysis

In HyperAT, not only do we want to help designers structure information, we also want them to avoid structural inconsistencies and mistakes. By treating users like computers, we implemented a formal way of analysing the structure of the hyperdocument. HyperAT allows designers to analyse the structure by firstly, performing integrity checks on the nodes and links, and secondly, measuring the complexity of the structure of hyperdocuments. The analysis reflects usability measures for various tasks based on the structure of the hyperdocuments. If the structure of a hyperdocument is inconsistent or too complex,

chances are that users would become confused and "lost". This first-cut evaluation of the hyperdocuments alerts designers to take corrective measures as early as possible in the design process before it is too late.

The more complex the structure of the hyperdocument is, the more easily users may feel "lost". In HyperAT, designers can detect structural inconsistencies like missing or inconsistently-named files/nodes. This form of analysis brings to designers' attention "silly" mistakes that can be easily rectified. The simple metrics implemented to measure the complexity of the structure of the hyperdocuments are:

- *No. of nodes*. HyperAT calculates the total number of nodes in the hyperdocuments, together with a listing of all the nodes present. If the number goes beyond a certain value, 10 000 nodes for example, then perhaps the structure may be too complex. Designers may have to decide to reorganise the structure.
- *No. of links per node*. This metric indicates how "busy" the nodes are in terms of the number of links per node, both in-coming links and out-going links. If a node has too many links, then designers might infer that it contains too much information. Perhaps the design decision is to split it into simpler nodes, since good design guideline suggests that nodes should be kept simple.
- *All possible paths from a given node*. Designers can query this information by selecting from a list of nodes. This information provides designers with all possible paths from a given node to all the leaf nodes in the hyperdocument. Designers can find out from this information the number of nodes that has to be taken to reach a leaf node. If the number is too high, then it would imply that the structure is too complex. Designers can perhaps make use of this information to provide more directed navigational help to users.
- *Depth of a structure*. This metric shows the number of levels away from the root node that is present in the hyperdocument. Accompanying this information is a global map of the structure. Clicking onto a node will open up another map which is a fisheye map of that particular node.
- *No. of successors*. This metric isolates those nodes with less than three successors from those with three or more successors. Though the number three may be arbitrary and may vary for different domains, it forces designers to re-think of the structure if too many nodes have more than three successors, violating their design principle to keep structure simple.

Real user evaluation

Real user evaluation is important because hyperdocuments are designed for users and not just what designers think or feel are important. Real users can be employed to evaluate hyperdocuments on the WWW with their transactions logged by the server log files, a view shared by Tim (1995) and Shneiderman (1997). However, Tim (1995) cautions that analysing the server log files takes time, if designers have to do that manually. Therefore, to help designers analyse these log files, HyperAT has a facility that parses and analyses server log files and interprets them, providing designers with useful insights into understanding users'. browsing pattern. In HyperAT, we have implemented the following:

- *Frequencies of visits*. This information provides designers with information on the most visited pages to spur them to think of reasons why certain pages are more frequently visited than others. Could it be a case of design flaws? The whole idea about giving designers this kind of information is to bring to their awareness and set them thinking about their designs.
- *Clients' browsing information*. This report provides designers with the browsing path of all clients who visited the websites held in the WWW server for a particular period. This means that designers can get real users to browse a website, and then analyse their browsing pattern.
- *Pages visited*. Designers can perform a query to find out information about pages visited. The data captured include the time and date of visit, and frequencies of visits.
- *Clients visited*. Designers can also query browsing behaviour of a certain client.

HyperAT also generates a report to to compare users' performance in terms of the goals satisfied and the number of steps taken to achieve these goals, with the actual steps based on designers' opinions (Figure 5). If users were taking more steps than expected, then designers might want to investigate the reasons by re-examining the structure of the hyperdocument, and/or interviewing the users concerned. In HyperAT, we have implemented some ideas to demonstrate that analysing the server log files containing otherwise untappped users' data, can be useful design aids to pinpoint usability problems, and guide design decisions.

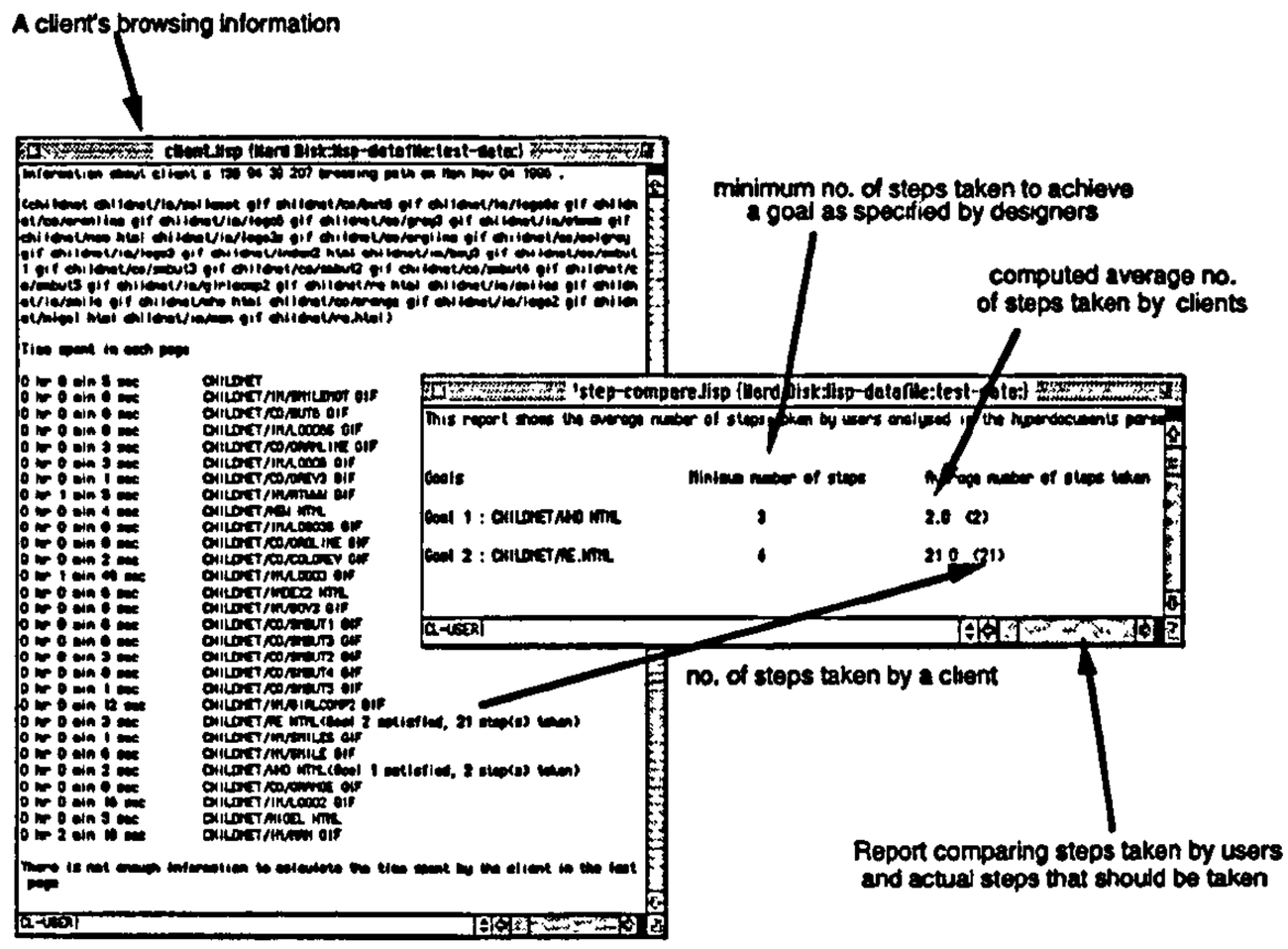


Figure 5. Report comparing steps taken by users and actual steps that should be taken

Executable user modelling

In user modelling evaluation, we propose incorporating CUM-DesTool into HyperAT to automate the usability evaluation of hyperdocuments produced by HyperAT. CUM-DesTool is a tool for running executable user models to pinpoint potential usability problems for interactive systems (Rigny and Thimbleby, 1996), so that hypertext designers can make informed decisions based on recommendations suggested for improved design. Executable user models are software agents that simulate real users' behaviour, as well as predict users' performance. They can embed multi-disciplinary knowledge that most designers and most users would not be expected to know or be able to verbalise in their accounts of interaction. They are able to do more exhaustive checking of the hypertext prototypes, long before they have reached a stage where actual human-user interaction would be practicable. Because user models are reusable, it is possible to rapidly simulate large groups of users and obtain useful statistical information. The idea in using executable models achieves a two-fold purpose in rapidly iterating the design process and avoiding many design blunders. However, the reliability and efficiency of the executable user models is very much dependent on the cognitive theories used to generate them (Barnard and May, 1993). It is, therefore, important that the results obtained from simulating executable user models be sufficiently tested for them to be reliable (Wilson and Clarke, 1993).

Preliminary work carried out to investigate incorporating CUM-DesTool into HyperAT demonstrates that it is feasible. Given that HyperAT generates a formal description of the hyperdocument (using HTML format), it is possible in principle to use the description as inputs into CUM-DesTool. All that needs to be done is to define and implement an interpreter to read and parse the description (Figure 6). Since CUM-DesTool and HyperAT are both written in LISP, combining the two tools is only a matter of programming.

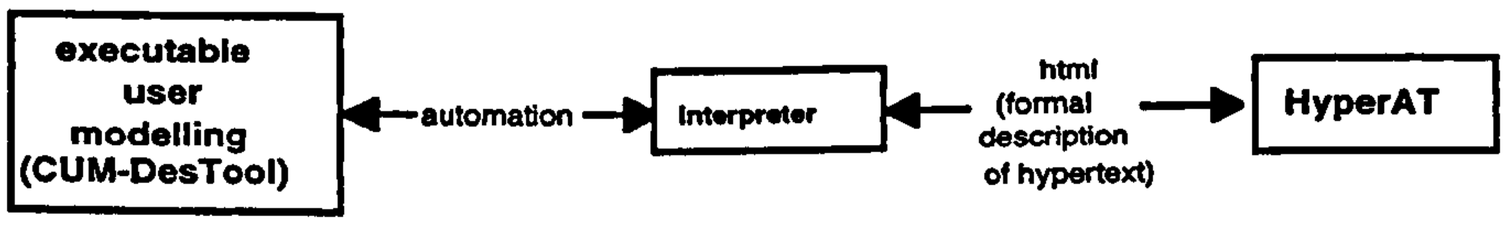


Figure 6. Implementing executable user models into HyperAT

5.0 Conclusions

Although much research effort has been invested to address the LIH problem, solutions have only been marginal. This paper described HyperAT, a research tool to help designers manage the complexity of the design and validation processes without themselves getting "lost". The approach taken in HyperAT is novel in that we integrated and implemented established HCI elements to ensure proper structuring and presentation of hyperdocuments, as well as to provide different modes of usability evaluation of the hyperdocuments.

Work with HyperAT now involves validating it and the approach it represents with different types of designers (e.g., novice, intermediate, experienced), as well as strengthening the usability environment it can offer to help designers build better, usable WWW pages.

References

Am, O. (1994), "Cyberspace and the structure of knowledge," <http://www.hsr.no/~onar/ess/cyberspace_and_the_structure_of_knowledge.html>

Barnard P. J. and May J. (1993), "Cognitive Modelling for User Requirements," In P. F. Byerley, P. J. Barnard & May J. (Eds), *Computers, Communication and Usability: Design issues, research and methods for integrated services*, Elsevier, pp 101-145.

Clibbon, K. and Callaghan, M. (1996), "Beyond Halasz's Hypertext Research Agenda – The WWW?" to appear in Special Issue of *International Journal of Human-Computer Studies on 'HCI & The Web'*, Buckingham Shum, S. and McKnight, C., eds. (forthcoming, 1997).

Cockburn, A. and Jones, S. (1995), "Trails, trials and tribulations: unravelling navigational problems in the world-wide web," *Proceedings of the 5th Workshop on Information Technologies and Systems (WITS '95)*, Netherlands.

Garzotto, F., and Paolini, P., Schwabe, D., and Bernstein, M. (1991), "Tools for designing hyperdocuments," In Berk, E. and Delvin, J. (Eds.), *Hypertext/Hypermedia Handbook*, McGraw-Hill.

Halasz, F.G. (1987), "Reflections on NoteCards: Seven issues for the next generation of hypermedia systems," *Proceedings of Hypertext '87*, pp. 345-365, ACM Press.

Hall, W., Carr, L. and Roure, D.D. (1995), "Linking the WWW and Microcosm," *BCS Workshop on New Directions in Software Development.*

Harold, E.R. (1996), "The comp.lang.java FAQ List,"<http://sunsite.unc.edu/javafaq/javafaq.html>

Jones, M. (1996), "Uniting authors and readers – active links on the World Wide Web," *APCHI96 Conference Companion*, pp. 77–81.

Lynch, P. J. (1995), " Yale's World Wide Web Style Manual," <http://info.med.yale.edu/caim/manual-1.html>

Maurer, H. (1996), *HyperWave: The Next Generation WEB Solution*, Addison-Wesley.

Mohageg, M. F. (1992), "The influence of hypertext linking structures on the efficiency of information retrieval," *Human Factors*, **34 (3)**, pp. 351-367.

Nielsen, J. (1997), "Trends for the web in 1997," Jakob Nielsen's Alertbox for January 1997, <http:www.useit.com/alertbox/9701.html>

Nielsen, J. (1996), "Relationships on the Web," Jakob Nielsen's Alertbox for January 1996, <http://www.useit.com/alertbox/9601.html>

Nielsen, J. (1995a), *Multimedia and Hypertext: The Internet and Beyond*, AP Professional.

Nielsen, J. (1995b), "Navigation features in Netscape 1.1," Jakob Nielsen's Alertbox for July 1996, <http://www.useit.com/alertbox/9507.html>

Pitkow, J. and Kehoe, C. (1995), *GVU's WWW 4th User Surveys,* <http://www.cc.gatech.edu/gvu/user_surveys/survey-10-1995/>

Pozadzides, J. and Quinn, L. (1997), "Cascading Style Sheet Quick Tutorial," <http:www.htmlhelp.com/reference/css/quick-tutorial.html>

Rada, R. and Murphy, C. (1992), "Searching versus browsing in hypertext," *Hypermedia*, **4(1)**, pp.1-30.

Rigny, C. and Thimbleby, H. (1996), "CUM-DesTool: applying executable user models for designing interacting systems," *HCI'96 Adjunct Proceedings*, pp. 145–149.

Shneiderman, B. (1997), "Designing information-abundant websites: Issues and Recommendations," to appear in *International Journal of Human-Computer Studies (1997).*

Smith, P. and Newman, I. (1996), "Applying usability research to the Web: Virtual hypermedia domains and virtual search hierarchies," to appear in Special Issue of *International Journal of Human-Computer Studies on 'HCI & The Web'*, Buckingham Shum, S. and McKnight, C., eds. (forthcoming, 1997).

Theng, Y.L., Rigny, C., Thimbleby, H., and Jones, M. (1996), "Cognitive task graphs and executable user models for better hypertext," *APCHI'96*, pp. 421–433.

Thimbleby, H. (1995), "Middlesex University Style Guide," <http://www.cs.mdx.ac.uk/esrc/style.html>

Thimbleby, H., Jones, M. and Theng, Y.L. (1997), "Is 'lost in hyperspace' lost in controversy?" *Hypertext'97*, Southampton (U.K.).

Tilton, J. (1996), "Composing good HTML," <http://www.cs.cmu.edu/~tilt/cgh>

Tim, B.L. (1995), "Style Guide for Online Hypertext," <http://www.w3.org/pub/www/provider/style/all.html>

Wilson, F. and Clarke, A. (1993), "Evaluating system design realisations," In P. F. Byerley, P. J. Barnard & May J. (Eds), *Computers, Communication and Usability: Design issues, research and methods for integrated services*, Elsevier, pp 379–411.