HyperAT: HCI and Web Authoring

Yin Leng Theng, Cecile 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.

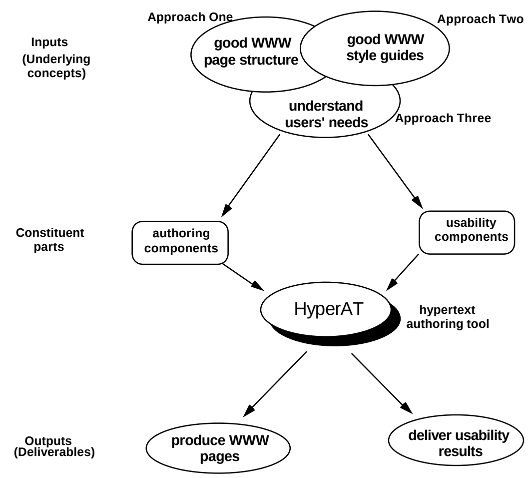

Figure 1. General overview of HyperAT, its inputs and outputs

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.

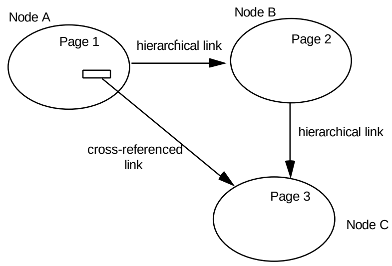

Figure 2. Relationship between nodes A, B and C

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.

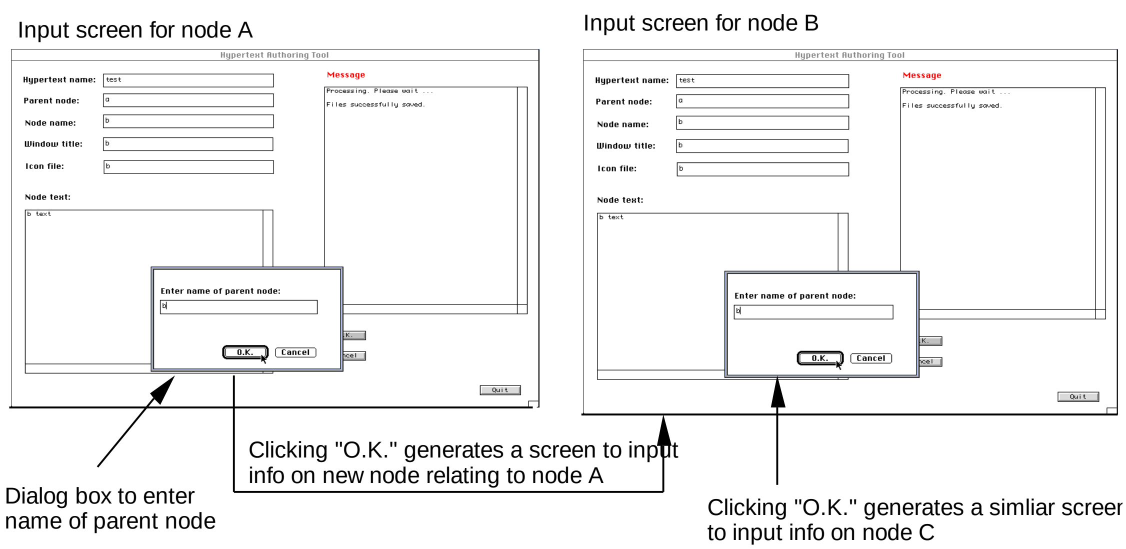

Figure 3. Screen shots to input information on nodes A, B and C

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.

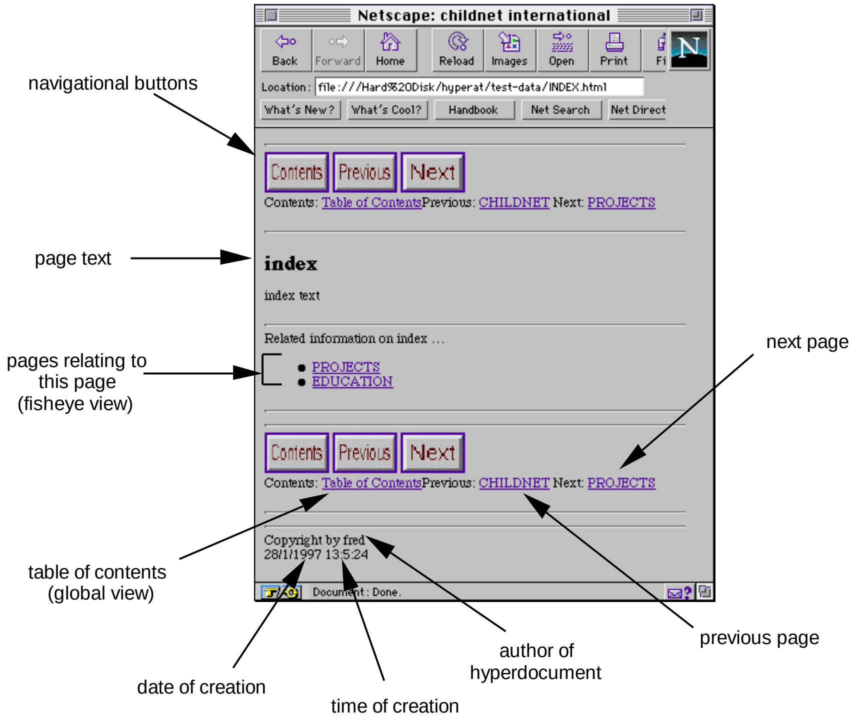

Figure 4. A sample WWW page generated by HyperAT

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.

<u>Structural analysis</u>

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.

<u>Real user evaluation</u>

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.

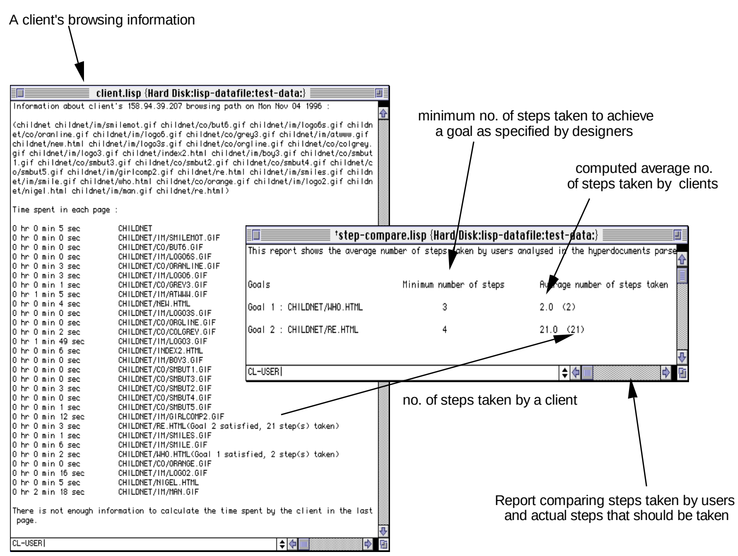

Figure 5. Report comparing steps taken by users and actual steps that should be taken

<u>Executable user modelling</u>

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.

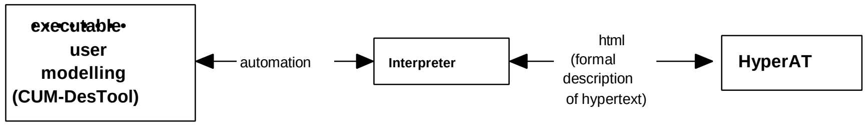

Figure 6. Implementing executable user models into HyperAT

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.