

Which Way Now? Analysing and Easing Inadequacies in WWW Navigation

Andy Cockburn

Department of Computer Science
University of Canterbury
Christchurch, New Zealand
andy@cosc.ac.canterbury.nz

Steve Jones

School of Informatics
University of Abertay Dundee
Dundee, Scotland. DD1 1HG
MCTSRJ@tay.ac.uk

February 21, 1996

Abstract

This paper examines the usability of the hypertext navigation facilities provided by World Wide Web client applications. A notation is defined to represent the user's navigational acts and the resultant system states. The notation is used to report potential, or 'theoretical,' problems in the models of navigation supported by three web client applications. A usability study confirms that these problems emerge in actual use, and demonstrates that incorrect user models of the clients' facilities are common. A usability analysis identifies inadequacies in the clients' interfaces.

Motivated by the analysis of usability problems, we propose extensions to the design of WWW client applications. These proposals are demonstrated by our system WEBNET which uses dynamic graphical overview diagrams to extend the navigational facilities of conventional World Wide Web client applications. Related work on graphical overview diagrams for web navigation is reviewed.

1 Introduction

The smallest of usability problems, when multiplied across thousands or millions of users, becomes a source of massive inefficiency and untold frustration (Nielsen, 1993). With the rapid increase of Internet usage, particularly of the World Wide Web (WWW), it is clear that popular WWW client applications (also termed browsers) such as Netscape and Mosaic are highly susceptible to this problem of escalating errors. Nielsen and Sano (1995) estimate that "it costs the world economy about half a million dollars in lost user productivity every time we add one more design elements (sic) to Sun's home page." If a single home page can induce losses of this scale, then the most trifling of usability flaws in the clients used to access *all* WWW pages must induce enormous productivity losses.

This paper examines the usability problems associated with navigation through WWW information spaces and describes a system that is designed to overcome some of these problems. Several other systems supporting novel web browsing paradigms are

under development, but there is a dearth of usability analysis revealing the nature, or existence, of difficulties in web navigation.

The browsing facilities of three WWW client applications (Netscape¹, Mosaic², and tkWWW³) are scrutinised. Our investigative approach focuses on a systematic analysis of client features (using a notation for describing web navigation and browser system states), on observation of usage, and on analysis with respect to usability principles.

This analysis has implications at three levels of WWW usage. First, page designers need to fully understand the navigation features of WWW browsers in order to best support their readers' needs. Second, browser designers need to consider the mapping from the navigation models implemented in their systems to the user models of those features. Third, problems and inconsistencies revealed in browser interfaces can be ameliorated through user-centred design of WWW navigational tools, which provide appropriate metaphors and utilities for navigation. WEBNET, described in this paper, is designed to overcome many of the navigational difficulties found in current browsers.

The structure of this paper is as follows. The navigation support facilities provided by the three browsers are introduced in section 2. The problems derived from the behaviour of the browsers' navigation support are described in the following four sections. First, section 3 introduces a notation that is used to precisely describe the user's navigational behaviour and the resultant system states. Theoretical problems with this behaviour are identified: that is, properties that seem inconsistent and curious are noted. Second, section 4 describes a usability study which demonstrates that users are misled by browsing behaviour and consequently form incorrect mental models which lead to disorientation. Third, section 5 uses heuristic usability analysis to pinpoint the flaws in the browsers' features. Fourth, section 6 notes further problems arising from the limited extent of the browsers' support for navigation. Section 7 describes and discusses WEBNET: a system that uses dynamic navigational overview maps to avoid these problems, and to extend the navigational facilities provided. Related work with other novel browsing systems are described in section 8. The paper concludes with an outline of future work and a summary of the main points that we present.

2 Support for Navigation in Three WWW Browsers

This section describes the navigation functions available in Mosaic, Netscape and tkWWW. It has recently been reported by Catledge and Pitkow (1995) that Mosaic accounts for 53% of all WWW related access to HTTP servers. We include Netscape as it is widely regarded as a popular successor to Mosaic. tkWWW is included because it is public domain software which we have extended to produce WEBNET which is described in section 7. The central navigation facilities of many other browsers, including Sun's HotJava⁴, are similar to those described here.

¹Netscape Communications Corporation, 501 East Middlefield Road, Mountain View, CA 94043.

²NCSA Mosaic is a product of the Software Development Group of the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign.

³By Joseph Wang. Global Network Academy, Macvicar Institute for Educational Software Development. Under terms of the GNU Public License Version 2.0.

⁴Sun Microsystems, Inc., 2550 Garcia Ave., Mtn. View, CA 94043-1100 USA. <http://java.sun.com/>

2.1 Classes of Page Display

The three browsers support similar models of navigation: that is, the *designers'* models are similar. Each browser provides functions for first-time selection and display of a page, for navigation through previously displayed pages, and for reloading a page. Each of the clients supports three classes of user technique for page display, which we term *loading*, *recalling*, and *revisiting*. Loading of pages occurs through *direct access* to a page which may or may not have been previously visited in the current session. Methods of directly accessing a page are URL⁵ selection, interest list⁶ selection, use of hard-wired client buttons, or selection of hypertext links. Recalling pages is achieved by navigation through a list of previously visited pages using commands such as forward and back. Revisiting pages is achieved by an explicit command to reload a page. Table 1 shows the eight possible methods for users to display a new page, and identifies whether each way loads, recalls or revisits pages in the three clients.

A given page may be displayed using any of the eight methods. However, the underlying class of technique significantly affects the state of the system after the page is displayed and therefore alters the effect of subsequent browsing commands. The class of page display technique used is transparent to users. Equally transparent to users is whether a page display request will require the page to be retrieved from a local cache or from its source location. Caching options (such as those accessible through the 'Network Preferences' in Netscape 2.0) allow users to tailor the underlying communication of the browser, but the fundamental interface behaviour described here remains unchanged.

| Num | Page selection method | Mosaic | Netscape | tkWWW |
|-----|--|---------|----------|---------------|
| 1 | Typed URL or command line option | Load | Load | Load |
| 2 | Hot-list or Bookmark selection | Load | Load | Load |
| 3 | Client-dependent hard-wired page buttons and menus | Load | Load | Load |
| 4 | HyperText link selection | Load | Load | Load |
| 5 | Forward | Recall | Recall | Not Available |
| 6 | Back | Recall | Recall | Recall |
| 7 | Go-List / History-List / Recall-List | Recall | Recall | Load |
| 8 | Reload | Revisit | Revisit | Revisit |

Table 1: Eight ways of displaying pages, and their system semantics.

2.2 Stack-Based Navigation

The clients' *Forward*⁷ and *Back* buttons do not control browsing of a temporal ordering of previously visited pages, but rather determine the currently displayed page in a *stack* of pages. This has also been noted by Brown and Shillner (1995). At the top of the stack is the page that has been most recently loaded. At the bottom of stack is the page that was least recently loaded.

⁵Uniform Resource Locator. The location of, and path to a specific WWW page and the communication protocol to be used in retrieving the page.

⁶Such as the Bookmarks list in Netscape or the Hotlist in Mosaic.

⁷tkWWW does not have a *Forward* button.

The stack of recallable pages is termed *History-list* in Mosaic, *View History* in Netscape, and *Recall list* in tkWWW. In the remainder of the paper, we will use the Mosaic terminology, *History-list*. Importantly, the *History-list* is not necessarily a stack of *all* previously visited pages. To *load* a page, while at some point other than the top of the stack, causes all pages above the current position in the stack to be lost: it is then impossible to use the *History-list* to retrieve those deleted pages. *Forward* and *Back*, then, allow the user to move up and down the stack of the *History-list*, recalling previously visited pages. Loading a page, while within the stack, risks losing the ability to recall pages.

In Mosaic and tkWWW the visual representation of the stack of pages grows from the top down with most recent pages at the bottom, but in Netscape the stack grows upwards. Mosaic and Netscape dynamically update their *History-list* stacks (including the current page), but tkWWW does not. Thus, in tkWWW the actual and visible histories could be entirely different. All three applications allow pages within the *History-lists* to be accessed directly by clicking on them, but tkWWW differs by loading the selected page rather than recalling it.

3 A Notational Study of Navigation Behaviour

In this section we present a notation to represent the user's navigational acts and the resultant system states. Using the notation, with reference to a sample WWW subspace, we analyse system behaviour in certain browsing scenarios. This analysis reveals system behaviour which, we contend, is likely to exacerbate users' feelings of being 'lost in hypertext,' and cause them difficulties in navigating WWW subspaces.

3.1 A Browsing Notation

The navigational features of the client applications can be described more formally by use of the following notation. The set of pages visited in a browsing session is represented by \mathcal{P} . Members of this set which begin a navigation path through a distinct WWW subspace are referred to as $p_1, p_2, p_3, \dots, p_n \in \mathcal{P}$. $p_{i1}, p_{i2}, p_{i3}, \dots, p_{in}$ denote the pages accessible via links present on page p_i . Traversal between pages is represented $p_i \rightarrow p_{ij} \rightarrow p_{ijk}$. For example, $p_1 \rightarrow p_{11} \rightarrow p_{111}$. A set of pages in the *History-list* is represented $\{p_i, p_j, p_k, \dots\}$.

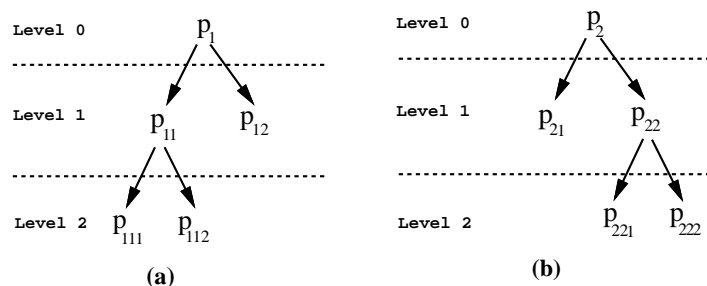


Figure 1: Sample WWW subspaces or navigation paths. Each subspace contains five pages and four links. Subspace (a) begins with page p_1 and is used in the following examples. Subspace (b) begins with page p_2 .

The discussion below refers to sample WWW subspace (a) shown in figure 1. This subspace contains five pages. The first loaded page is p_1 which contains links to two other pages p_{11} and p_{12} . Page p_{11} contains links to two other pages p_{111} and p_{112} .

Figure 1 does not imply that pages are arranged hierarchically. Indeed, there may be many other paths to the pages than those shown within the given subspaces. Similarly, the notation does not imply a hierarchical page structure, but rather a trail through a subset of WWW pages.

3.2 Navigations leading to an incomplete history-list

| State | Current history list | Current page | Loaded page | Recalled page | Resultant History list |
|-------|----------------------------|--------------|-------------|---------------|----------------------------|
| 1 | {} | | p_1 | | { p_1 } |
| 2 | { p_1 } | p_1 | p_{11} | | { p_1, p_{11} } |
| 3 | { p_1, p_{11} } | p_{11} | p_{111} | | { p_1, p_{11}, p_{111} } |
| 4 | { p_1, p_{11}, p_{111} } | p_{111} | | p_{11} | { p_1, p_{11}, p_{111} } |
| 5 | { p_1, p_{11}, p_{111} } | p_{11} | p_{112} | | { p_1, p_{11}, p_{112} } |
| 6 | { p_1, p_{11}, p_{112} } | p_{112} | | p_1 | { p_1, p_{11}, p_{112} } |
| 7 | { p_1, p_{11}, p_{112} } | p_1 | p_{12} | | { p_1, p_{12} } |
| 8 | { p_1, p_{12} } | p_{12} | | | { p_1, p_{12} } |

Table 2: Browsing behaviour leading to an incomplete *History-list*.

Table 2 represents a simple WWW browsing session, involving seven navigation actions in a subspace of five WWW pages. It demonstrates how subtle distinctions between loading and recalling pages can result in incomplete *History-lists*: the scenario below assumes the user is browsing with Netscape or Mosaic; tkWWW would maintain the full history (in this case) because direct selection from its recall list causes pages to be loaded.

At state 1 the user loads the first page p_1 . At state 2 the user clicks on a link to p_{11} , causing p_{11} to be loaded, and causing a traversal of $p_1 \rightarrow p_{11}$ resulting in a *History-list* of { p_1, p_{11} } with p_{11} the currently displayed page. At state 3 the user selects a link to p_{111} , causing a traversal of $p_{11} \rightarrow p_{111}$ resulting in a *History-list* of { p_1, p_{11}, p_{111} } with p_{111} the currently displayed page. At state 4 the user recalls p_{11} from the *History-list*, resulting in a change only to the current page.

At this point, p_{11} is taken to be the top of the *History-list* stack. When the user loads p_{112} at state 5 (this is a link within page p_{11}), the *History-list* is amended, with p_{111} popped off the top of the stack, and replaced by p_{112} . At state 6 the user recalls p_1 from the *History-list* and it becomes the current page, with no change to the *History-list*, and p_1 is taken to be the top of the stack. At state 7 the user loads p_{12} (this is a link within page p_1), and pages p_{11} and p_{112} are popped off the top of the stack, replaced by p_{12} .

Hence, after the user has visited only five pages using seven navigation actions, three of the five pages are no longer present in the *History-list*. The user will be unable to return to more than half of their visited pages through use of the *back* and *forward* buttons. To generalise, with reference to figure 1, if a page at level i is recalled, and a subsequent page is loaded, then all pages at level $i + 1 \dots n$ are removed from the *History-list*: they are popped off the top of the stack.

Any user whose model of the *History-list* is temporal rather than stack-based will find navigation behaviour unpredictable or ‘non-deterministic.’ Thankfully people are remarkably adept at working with incomplete and erroneous models (Norman, 1988; Thimbleby, 1990), but this is hardly a desirable or efficient situation. In our usability study (section 4), the majority of subjects incorrectly predicted the recoverability status of pages.

3.3 Browsing behaviour leading to the same display, but different states

Some browsing activities can result in repeated entries in the history list. This has the implication that the user may be viewing a page which occurs in the history list more than once. The effect of future navigation operations is dependent upon which instance of the page the user is viewing.

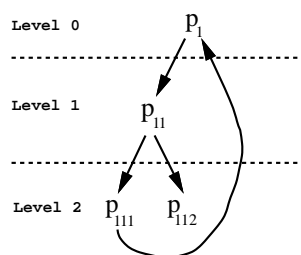


Figure 2: The sample WWW subspace used in Task 2. It contains an embedded link to a previously visited page.

| State | Current history list | Current page | Loaded page | Recalled page | Resultant History list |
|-------|---|--------------|-------------|---------------|---|
| 1 | {} | | p_1 | | { p_1 } |
| 2 | { p_1 } | p_1 | p_{11} | | { p_1, p_{11} } |
| 3 | { p_1, p_{11} } | p_{11} | p_{111} | | { p_1, p_{11}, p_{111} } |
| 4 | { p_1, p_{11}, p_{111} } | p_{111} | p_1 | | { $p_1, p_{11}, p_{111}, p_1'$ } |
| 5 | { $p_1, p_{11}, p_{111}, p_1'$ } | p_1' | p_{11} | | { $p_1, p_{11}, p_{111}, p_1', p_{11}'$ } |
| 6 | { $p_1, p_{11}, p_{111}, p_1', p_{11}'$ } | p_{11}' | | p_1 | { $p_1, p_{11}, p_{111}, p_1', p_{11}'$ } |
| 7 | { $p_1, p_{11}, p_{111}, p_1', p_{11}'$ } | p_1 | p_{11} | | { p_1, p_{11} } |
| 8 | { p_1, p_{11} } | p_{11} | | | |

Table 3: Browsing behaviour leading to the same displayed page but different states.

Consider figure 2. In this case page p_{111} contains a link to page p_1 . States 1 through 3 of table 3 are identical to those of table 2. They show previous navigation of $p_1 \rightarrow p_{11} \rightarrow p_{111}$. At state 4 the loaded page is p_1 and in this case this is accessed via a link present on p_{111} . Such a navigation results in p_1 appearing repeatedly in the history list (p_1' indicating the newly created repetition). At state 5 the user selects the link to page p_{11} . This results in a repetition of page p_{11} in history list (denoted p_{11}'). At state 6 p_{11}' is the currently displayed page. The user recalls p_1 via the history list. No

change occurs in the history because the page was recalled. At state 7 the user loads page p_{11} via the link on p_1 , resulting in the removal of pages p_{111} , p_1' and p_{11}' from the history list.

Consider the user action of loading p_{11} in states 5 and 7. In both states the same page is displayed to the user. However, these are two different system states. Loading p_{11} from p_1' in state 5 results in further repetition. Loading p_{11} from p_1 in state 7, however, results in the loss of pages from the history list. Note that in this case the resulting display will be the same; the user will be presented with page p_{11} .

The *forward* and *back* navigation operations are also affected by this behaviour. For example, *forward* is available from p_1 in state 5, but not from p_1' . *Back* is available from p_1' but not from p_1 .

Of course, as repetitions increase, the ‘non-deterministic’ behaviour of the system is compounded. This can be seen by considering the presence of p_{11} and p_{11}' in states 6 and 7.

To summarise then, *loading* a page appends it to the top of the stack regardless of whether the page is already held within the stack. Consequently, cyclic page links (such as the frequently used ‘Back to page *xxxx*’) allow numerous versions of the same page to be concurrently held on the stack, but the the system state will be significantly different dependent on the instance of the page (section 5.4).

4 A Usability Study of Browsing Problems

This section describes a usability study of the problems associated with the browsers’ model of navigation. Specifically, the study is concerned with investigating the potential mismatch between the systems’ model and the users’ model of the clients’ facilities for loading and recalling pages. Although users may not realise it, and may never use it directly, the behaviour of navigational actions such as loading and recalling pages is determined by the *History-list*. We concentrate on the aspects of *History-list* behaviour described in sections 3.2 and 3.3.

The aim of the usability study was to investigate the authors’ supposition that the clients’ navigation behaviour, described in Sections 2 and 3, mislead users. The two parts of the study focus on the usage of the *Back* function of the browsers. This function has been identified as the most common technique of page display using system provided navigation functions (discounting link selection) accounting for 41% of all page requests (Catledge & Pitkow, 1995).

Task 1 investigates the accuracy of users’ models of page availability when pages are removed from the *History-list* as a result of navigations similar to those described in section 3.2. Task 2 investigates the accuracy of those models when pages appear repeatedly in the *History-list* as a result of navigations similar to those described in section 3.3.

4.1 The Subjects

Eleven subjects were studied. All subjects are Computer Science academic staff. The subjects engaged in some WWW navigation activity at least two times per week, and none were novice users of WWW browsing tools. Although this sample is not representative of the WWW community, it is reasonable to assume that the performance of Computer Scientists should be no worse than other users. Indeed, they might be expected to maintain more accurate models of the system than other sections of the

community given the stacking behaviour of the internal model of WWW browsers. Subjects were allowed to use their preferred WWW browser, which in all cases was Netscape.

4.2 The WWW Subspace

The home pages of the Computer Science department in which the subjects are employed, and home pages of the staff themselves, were selected as the WWW subspace to be navigated around in both tasks. Again, it is reasonable to assume that the subjects' performance should be no worse than normal when browsing pages that they had authored. Figure 3 shows the web subspace navigated in the subjects' tasks.

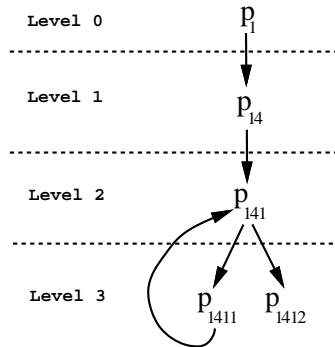


Figure 3: Sample web subspace for the usability study. The subscript values represent the ordering of the links on the page.

4.3 Identifying the Data-Structure

Initially, subjects were asked to name the data-structure that best reflected the behaviour of the *History-list*. This information reveals whether repeated, frequent usage of the browser results in an accurate user model of its behaviour. It also contextualizes subjects' responses to questions within the studies.

One subject described the *History-list* as a 'Stack,' one described it as a 'tree,' one described it a 'double ended queue,' and eight described it as a 'list' structure. Therefore, only one subject correctly identified the implemented structure. Noticeably, the majority of subjects identified the structure as being a form of list. This model is the one most strongly suggested by the forward and back operations, and representation of the *History-list* supported by the browser.

4.4 Task 1: Navigating using the browser's *Back* button

The subjects' first task was to navigate through the WWW subspace as follows:

1. start at the Computer Science home page p_1 ;
2. traverse to the staff index page p_{14} —by link selection ;
3. traverse to a specified person's page p_{141} —by link selection;

4. traverse to the *first* ‘current project’ page p_{1411} —by link selection;
5. traverse back to the same person’s page p_{141} —using the client’s *Back* button;
6. traverse to the *second* ‘current project’ page p_{1412} —by link selection;

Table 4 uses the notation to represent the browsing states reached in the task.

Having completed these actions (state 7), subjects were asked to state whether the *first* ‘current project’ page (p_{1411}) could be revisited using the client’s *Back* button. In fact it cannot be recalled, because the page was removed from the *History-list* by loading the second ‘current project’ page (p_{1412}).

| State | Current history list | Current page | Loaded page | Recalled page | Resultant History list |
|-------|--------------------------------------|--------------|-------------|---------------|--------------------------------------|
| 1 | {} | | p_1 | | { p_1 } |
| 2 | { p_1 } | p_1 | p_{14} | | { p_1, p_{14} } |
| 3 | { p_1, p_{14} } | p_{14} | p_{141} | | { p_1, p_{14}, p_{141} } |
| 4 | { p_1, p_{14}, p_{141} } | p_{141} | p_{1411} | | { $p_1, p_{14}, p_{141}, p_{1411}$ } |
| 5 | { $p_1, p_{14}, p_{141}, p_{1411}$ } | p_{1411} | | p_{141} | { $p_1, p_{14}, p_{141}, p_{1411}$ } |
| 6 | { $p_1, p_{14}, p_{141}, p_{1411}$ } | p_{141} | p_{1412} | | { $p_1, p_{14}, p_{141}, p_{1412}$ } |
| 7 | { $p_1, p_{14}, p_{141}, p_{1412}$ } | p_{1412} | | | |

Table 4: System states in the usability study: task 1.

| Subject | Initial Model | Task 1 | | Task 2 | |
|---------|---------------|--------------------------------------|------------|--------------------------------------|------------|
| | | Correct? | Confidence | Correct? | Confidence |
| 1 | List | ✗ | weak | ✓ | v weak |
| 2 | List | ✓ | weak | ✓ | strong |
| 3 | Stack | ✓ | strong | ✓ | strong |
| 4 | List | ✗ | weak | ✗ | weak |
| 5 | List | ✗ | strong | ✓ | weak |
| 6 | Tree | ✗ | weak | ✓ | weak |
| 7 | List | ✗ | weak | ✓ | weak |
| 8 | List | ✗ | weak | ✗ | strong |
| 9 | Dequeue | ✓ | strong | ✗ | strong |
| 10 | List | ✗ | weak | ✗ | strong |
| 11 | List | ✗ | weak | ✗ | strong |
| Totals | | 3 of 11 Correct 8 of 11 Incorrect | | 6 of 11 Correct 5 of 11 Incorrect | |

Table 5: The subjects’ initial model of the *History-list*, responses in both tasks, and the strength of responses in both tasks.

Two subjects confidently responded correctly, one subject unconfidently responded correctly, seven subjects unconfidently responded incorrectly and one subject confidently responded incorrectly. Responses were deemed to be “unconfident” when the subjects used statements such as “Hmm, I guess it can’t be,” or “I’m not sure, but it probably can be.” Table 5 shows the results of Task 1.

4.5 Task 2: Navigating using explicit ‘Back’ page links

Task 2 utilised the same WWW subspace as Task 1. The navigations required of users resulted in an identical display of pages to Task 1. However, the task differs in the method of carrying out a traversal $p_{1411} \rightarrow p_{141}$. In Task 2 this is achieved by selection of an explicit link within page p_{1411} to page p_{141} .

1. start at the Computer Science home page p_1 ;
2. traverse to the staff index page p_{14} —by link selection;
3. traverse to a specified person’s page p_{141} —by link selection;
4. traverse to the *first* ‘current project’ page p_{1411} —by link selection;
5. traverse back to the same person’s page p_{141} —using the explicit ‘Back to my home page’ link on the page.
6. traverse to the *second* ‘current project’ page p_{1412} —by link selection;

Table 6 uses the notation to represent the browsing states reached in the task. In this task no pages were recalled, all were loaded.

Having reached state 7, subjects were asked to state whether the *first* ‘current project’ page (p_{1411}) could be revisited using the client’s *Back* button. In fact it can be recalled, because traversal using an explicit link loads page p_{141} and appends it to the top of the stack (represented p_{141}').

| State | Current history list | Current page | Loaded page | Resultant History list |
|-------|--|--------------|-------------|--|
| 1 | {} | | p_1 | { p_1 } |
| 2 | { p_1 } | p_1 | p_{14} | { p_1, p_{14} } |
| 3 | { p_1, p_{14} } | p_{14} | p_{141} | { p_1, p_{14}, p_{141} } |
| 4 | { p_1, p_{14}, p_{141} } | p_{141} | p_{1411} | { $p_1, p_{14}, p_{141}, p_{1411}$ } |
| 5 | { $p_1, p_{14}, p_{141}, p_{1411}$ } | p_{1411} | p_{141} | { $p_1, p_{14}, p_{141}, p_{1411}, p_{141}'$ } |
| 6 | { $p_1, p_{14}, p_{141}, p_{1411}, p_{141}'$ } | p_{141} | p_{1412} | { $p_1, p_{14}, p_{141}, p_{1411}, p_{141}', p_{1412}$ } |
| 7 | { $p_1, p_{14}, p_{141}, p_{1411}, p_{141}', p_{1412}$ } | p_{1412} | | |

Table 6: System states in the usability study: task 2. No pages were recalled, all were loaded.

Two subjects confidently responded correctly, four subjects unconfidently responded correctly, four subjects confidently responded incorrectly, and one subject unconfidently responded incorrectly. Table 5 shows the result of Task 2.

4.6 Observations from the Study

The usability study was intended to test the authors’ suspicions of mismatches between user and system models of navigation support in WWW browsers. The study supports our suspicions, but was a small scale, informal investigation, and is therefore inadequate for formal evaluation.

Despite the limited ambitions of the study, its findings are interesting. One subject (subject 3) clearly had a correct mental model of system behaviour. We believe that one other subject (subject 2) also had a correct model (presenting correct answers in

both tasks) but was unable to articulate it. The others, who were all familiar with the application, were surprised when shown the stack-based navigational metaphor built into the system on completion of the tasks. This implies that the navigation metaphor presented in the user interface does not either accurately or sufficiently reinforce the systems' underlying navigation model.

One subject, whose home page had been the subject of the task, was fascinated to find that his 'Back to my home page' explicit page link had the side effect of allowing subsequent pages to be navigated without losing others from the *History-list*: this 'feature' had not been considered when he designed the page. Of course, this design has the disadvantage that repetitions can appear in the *History-list*. Further studies into the design considerations and decisions of WWW page designers may reveal whether they are aware of the implications of different page design models.

The majority of subjects did not have a correct model, and were not confident in their (knowingly incomplete) models. No subjects investigated the history list to support their answer to the navigation question posed in Task 1. This may imply a lack of awareness of the feedback provided by the *History-list* with regards to the accessibility of pages. This is supported by Catledge and Pitkow (1995), who found that *History-list* access to WWW pages accounted for a very small proportion of all page accesses.

A learning effect accounts for some interesting observations in Task 2. There are two sides to the learning effect. First, as a result of Task 1, most subjects found their models to be incorrect. Although this might have resulted in their being less confident about Task 2, the trend is towards greater confidence. Second, in Task 2 the subjects were presented with exactly the same series of displayed pages as in Task 1, although the underlying system state was different. This may explain the general increase in confidence in responses, although the majority of confident responses were incorrect. Therefore, it seems that the user models were strongly affected by the identical page display: several users assumed (incorrectly) that an identical set of pages would mean an identical system state. It might also be the case that users had modified models in Task 2, as a result of their experience in Task 1. The behaviour of the system in the first context may have engendered a more accurate model for the particular web-subspace.

5 Heuristic Usability Analysis

This section presents an analysis of the client navigational features with respect to established usability principles. Section 3 noted problems associated with the curious behaviour of the clients' implemented model of browsing, and Section 4 observed that these problems emerge as incorrect mental models of system behaviour. This section identifies the interface flaws that cause, and contribute to, these problems.

All three web clients bear many of the hallmarks of good user interface design. A complete analysis of these systems with respect to a set of usability principles would illuminate the good aspects of the interfaces as well as the bad ones. We only report the bad points (of navigation), but this does not imply that the overall design of the systems is poor. Our aim is to identify the problems in order to improve the next generation of browsers.

The principles used to analyse the systems are those of Nielsen (1993), who terms them 'Usability Heuristics'. This set of principles was chosen because it is concise enough to be manageable, and generic enough to allow a wide range of features to be

examined.

The following subsections note the major navigation problems captured by the principles.

5.1 Simple and natural dialogue

This principle applies directly to the *presentation* of information. It is therefore of direct relevance to *page* designers. Guidance such as “less is more,” and Tufte’s ink count (1984), could be beneficially applied by many web page producers.

Equally “less is more” may be applied to browser design. We have identified eight different page display methods supported by the three browsers (see Table 1). Many of these methods are brought about by the navigation model embedded in the system, and more critically by the representation of that model in the user interface. The inappropriate, pseudo-linear nature of the representation *requires* multiple methods for page display. In section 7 we describe a navigation model and its user interface representation which significantly reduces the required number of page display methods.

A navigation problem associated with information *presentation* stems from the mechanisms used to denote previously traversed hypertext links. In Mosaic, page links that have not been traversed are underlined, and those that have been traversed are underlined with a dashed line. Netscape uses colour coding to distinguish between links that have and have not been traversed, and tkWWW makes no distinction. Thus, in Netscape the useful information on previous link traversals is unavailable to users with monochrome displays, and in tkWWW it is unavailable to everyone.

Another component of the ‘natural dialogue’ principle is that users should not be susceptible to unnaturally high costs for a simple slip or mistake. The guideline of ‘commensurate effort’ (Thimbleby, 1990) guides designers with the advice that ‘things that are difficult to do should be difficult to undo.’ Consequently, if the user has invested substantial effort in achieving a particular system state, a small slip or misconception of system behaviour should not allow them to irreversibly lose the state. The stacking behaviour of the *History-list*, and the consequent risk of irrecoverably losing any number of previously visited pages by mistakenly loading rather than recalling a page, certainly contravenes this guideline.

5.2 Speak the user’s language

This principle begins to uncover the source of the failed mapping from the user’s model to the designer’s model.

We contend that the button commands *Forward* and *Back* offer an affordance of one of three types. Our limited usability study corroborates the strength of the first two affordances.

1. Linear and incremental navigation—for instance, flicking forward or back through the *pages* of a book;
2. Temporally based navigation—for instance, “back to the places that I’ve been to before, and forward to return from there to where I am now.” HyperCard’s *Go Recent* is a good example a temporally based navigational metaphor (Coulouris & Thimbleby, 1992);
3. The ‘tape recorder’ analogy. Under this affordance, the natural user model would be that traversal through pages is ‘recorded,’ and that the user can ‘rewind’ back

or ‘fast-forward’ through the recorded pages. Recording at any other point than the start of the ‘tape’ would, naturally, over-write the information under the tape’s recording head.

The stacking behaviour of the web clients is most closely associated with the third affordance, but the metaphor is a weak one. If a real tape device were to operate consistently with the clients’ metaphor, then to record at any point in the tape would have the side effect of erasing all material on the remainder of the tape. None of the subjects in our study identified a tape-recorder metaphor.

The system-images of the web clients present mixed metaphors of their navigational techniques. The notion of temporally based navigation is strongly reinforced by the naming of the *History-list* in Mosaic, the *View history* operation in Netscape, and is partially reinforced by the tkWWW naming ‘Recall’.

tkWWW does not support a *Forward* button, lending the notion that traversal *Back* cannot be undone. Consequently, its system image is (arguably) closer to the stack-based navigation that it supports. tkWWW’s ‘Recall-list,’ however, allows the user to select any page in the stack, thus over-riding the stacking behaviour. Curiously, ‘recalling’ a page in this manner has the effect of loading it! It is not surprising that many users fail to determine the distinction between loading and recalling pages that is fundamental to the formation of an accurate user model.

5.3 Minimise memory load

The memory overload problem is clearly applicable to the WWW, and may explain unintentional ‘web-surfing’ that users find themselves engaged in. For example, a user looking at page p_1 decides that she will visit the pages p_{11}, p_{12} , and p_{13} from the page. She clicks on the link to p_{13} , and having read the information on the page (presumably masking some of the information in her short-term memory) sees that p_{13} contains ‘interesting’ pages p_{132}, p_{133} , and p_{134} . She clicks on page p_{133} and reads the material... now where was she?

Overcoming the memory load problem is the crux of all Hypertext navigation research. Research into appropriate browsing techniques has a long history, arguably beginning with Bush’s Memex in 1945. Many metaphors have been proposed (Conklin, 1988; Nielsen, 1990), with graphical representations of the hypertext web featuring prominently (Vora *et al.*, 1994). Our related work (see Section 7) is working towards graphical browsing tools that supplement current browsing techniques. The graphical representation provides a continuously visible surrogate for the user’s short-term memory, and various filters of the Hypertext assist users in maintaining a focussed view of the information of interest.

5.4 Consistency

‘Consistency’ as a user interface design principle is sufficiently generic to capture many usability principles that would otherwise be considered independently. We use it here for two purposes, first to capture *mode* problems in web navigation, and second to capture errors of *consistency* between the client’s internal state and its visual representation.

Loading and *recalling* pages are implemented as two different modes of page display. The visual affect of *loading* or *recalling* a page is to replace the current displayed page with the display of the selected page. Visually, loading and recalling are identical, apart from the sensitisation of the *Forward* button—a truly subtle cue! tkWWW does not

have a *Forward* button to provide even this cue. The existence of the modal distinctions between these two activities is not effectively communicated by the system image (as demonstrated by our usability study).

The possibility of duplicate pages p_i and p_i' concurrently existing on the *History-list* stack further exacerbates problems of consistency. A traversal $p_i \rightarrow p_i'$ (or vice-versa) will not change the page display, and the only visual cue of a change in system state would be the state of the *Forward* and *Back* buttons. This cue will only be available if one of the pages is within the stack while the other is at the top or bottom of the stack.

6 Additional Shortcomings with Support for Navigation

In this section we highlight further shortcomings in the systems. These are not concerned with flaws in the implementation of navigation facilities, but rather they are concerned with the *extent* of navigation support. First, we discuss the limitations on the range of pages accessible at a given time. We then consider the paucity of support provided for contextualising pages of interest within the visited WWW subspace.

6.1 Restricted Page Access

At any given time, each of the browser interfaces gives immediate access to a restricted set of WWW pages. Of course, *any* page within the WWW is accessible through specification of a URL, but page access methods 2 to 8 in Table 1 are designed to ease the burden of knowing, remembering, and typing URLs. Often the user does not know a specific desired destination, as demonstrated by WWW browsing (or 'web-surfing').

The clients' support for directed browsing, where the user is following a general area of interest, is poor. Once a page p_1 has been replaced by another page, there is no way to see the potential destinations from page p_1 . Browsing by continually spawning windows is one option that the clients support, but screen real-estate is rapidly consumed. A user following one interesting path from p_1 must return to p_1 to follow any other paths from that page. The user's memory is therefore a critical, and highly fallible, resource when browsing.

Catledge and Pitkow (1995) found that selection of hyperlinks and use of the back command accounted for 93% of all navigational activity in their study of WWW browsing strategies (methods 4 and 6 in Table 1). One explanation for this may be that users backtrack to a page several times in order to investigate links within it. Indeed, Catledge and Pitkow comment on the widespread use of a 'hub and spoke' browsing strategy in their study. They direct page designers to consider the distance of information from home pages, and to provide indexes to support efficient use of such strategies.

In Netscape, Mosaic, and tkWWW, navigation to subsequent links from the same page imposes overheads on the user in two ways:

1. additional navigational activity (at least one traversal to the hub) is required to display the page containing the required link;
2. previously visited pages may not, as we have described, be present in the history list, which may result in the cognitive overhead of deciding how to return to the desired page.

Frames, a new feature in Netscape Navigator 2.0 (Netscape Communications Corporation, 1996), can ease the problem of redundant traversals in hub-and-spoke browsing

by allowing multiple independently scrollable frames on a single screen, each with its own distinct URL. Unfortunately, navigations within sub-frames are not recorded on the history-list, and therefore pages cannot be revisited using ‘Back.’

Well considered page design will assist effective and efficient navigation through the WWW (Nielsen & Sano, 1995), but no set of page design style-guides (Press, 1995), or formal techniques (Isakowitz *et al.*, 1995; Balasubramanian *et al.*, 1995) will be adopted unilaterally across the web. We believe that the effects of poor page design can be ameliorated by improving the support for navigation in WWW browsers. Improved client navigation support will also help users browsing within well designed WWW subspaces (Section 7).

6.2 Lack of Context of Node of Interest

None of the three browsers provide feedback about the context of the currently displayed page within the visited subspace, nor do they allow users to view alternative representations of the visited subspace. In comparison to many currently available commercial Hypertext systems, the browsers do little in providing an appropriate presentational metaphor for navigation through a hypertextual information space.

Previous work on hypertext navigation has suggested the use of graphical or spatial overviews to help users to situate themselves within the hypertext (Conklin & Begeman, 1988; Nielsen, 1990; Marshall & Shipman, 1995). Associated work on distortion-oriented visualisation and fisheye views allow users focus on specific information while maintaining an awareness of the surrounding information context. Reviews of distortion-oriented and fisheye techniques are provided in Leung and Apperley (1994) and in Schaffer *et al.* (1996). Related work on systems demonstrating these techniques is reviewed in Section 8.

7 WEBNET: Extending Navigation Support

The previous sections have used a variety of techniques to analyse the usability problems and shortcomings associated with three WWW browsers. To provide the motivation for our work on WEBNET, these problems can be summarised as follows:

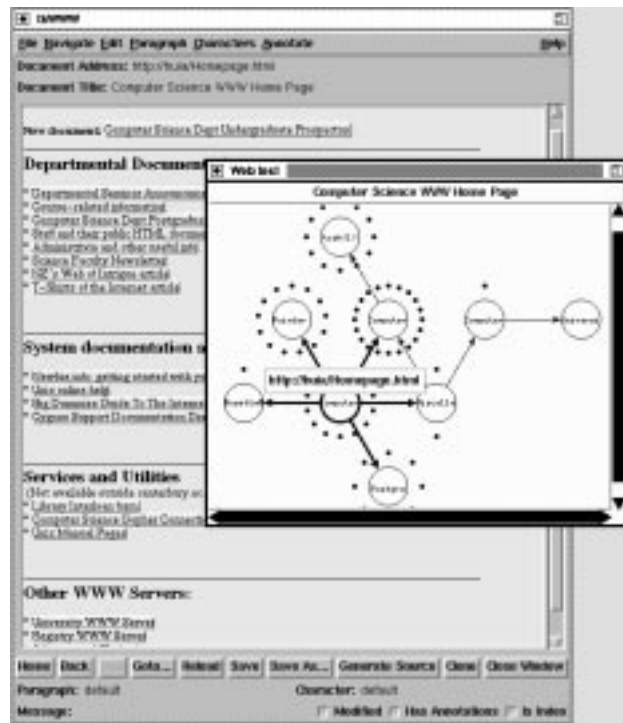
1. Failure of the users’ mental models of the navigation support provided by the browsers.
2. Lack of context. There is a paucity of support for the users’ awareness of where they are within a WWW subspace.
3. Memory overload problems.

WEBNET, an extended WWW browser, aims to overcome these problems through the use of dynamic navigational overview graphs: that is, graphical overviews of the WWW subspace through which the user is navigating. It is implemented using Tcl/Tk (Ousterhout, 1993) under the X Window System (Scheifler & Gettys, 1986).

The two following subsections describe the user interface to WEBNET, and its interface design rationale.



4a. A three page trail in WEBNET.



4b. tkWWW and an experimental version of WEBNET.

Figure 4: The WEBNET windows, showing a user's navigation path through a WWW subspace.

7.1 The WEBNET user interface

WEBNET is shown in Figure 4. Figure 4a is the full functionality version of WEBNET that we describe below. Figure 4b shows the tkWWW browser and an experimental subspace representation version of WEBNET. WEBNET provides an active representation of the WWW subspace through which the user is navigating. WEBNET exists in a separate window, but it offers 'equal opportunity' (Thimbleby, 1990) with the user's browser, allowing navigational acts to be initiated at the browser or at WEBNET with equivalent feedback in both companion tools.

WEBNET is designed to run along-side any standard web-browsing client. It does not embody a full redesign of existing support, but supplements and provides alternatives to the navigation facilities offered by the companion browser. There is no requirement for users make use of WEBNET's facilities: they can take it or leave it.

Currently WEBNET runs with the public domain browser tkWWW. It communicates with tkWWW (and vice versa) through remote procedure calls; both exist as separate processes. Consequently, WEBNET could be modified to communicate with other WWW browsers using the appropriate communication protocols (such as Mosaic's Common Client Interface (National Center for Supercomputing Applications, 1995)).

7.1.1 The WWW subspace overview

The main portion of the interface contains a scrollable graphical overview of the web subspace visited in a session. In figure 4a the user has visited three pages: *Computer Science WWW Home Page*, *Computer Science Course-Related Information* and

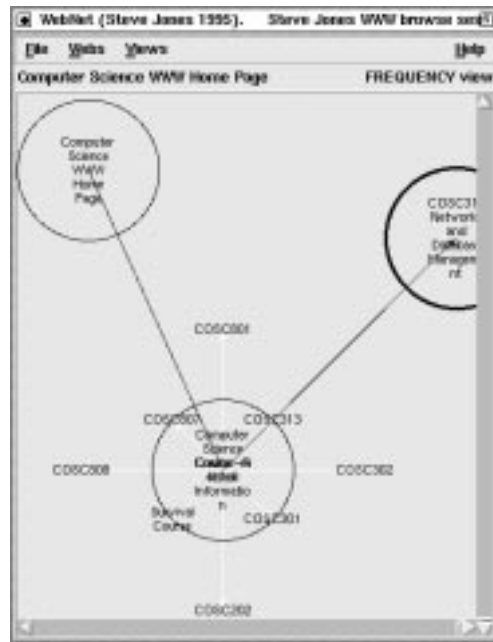


Figure 5: Revealing the hypertext links on page *Computer Science Course-Related Information*.

COSC313 Networks and Database Management. The user's path through the subspace is shown by blue lines and arrows which connect the nodes. The bold node in the overview indicates the current page displayed by the browser. The user's history of visited pages is shown by circular nodes.

Each node in the overview responds to three user actions. Clicking with the left mouse button displays the corresponding page in the tkWWW browser window. Clicking with the middle mouse button displays the titles of the links present on the corresponding page. Figure 5 shows the effect of clicking node *Computer Science Course-Related Information* with the middle mouse button. Displaying the links allows users to see where they *can* go, as well as where they have been. Displaying the links in this manner does not affect the browser's page display. Clicking on a node with the right mouse button hides the links emanating from it. It has no effect on the tkWWW page display.

Clicking on a link-title with the left mouse button displays the corresponding page in tkWWW, and WEBNET's overview is updated to show that the page has been visited.

A label in the top left of the overview indicates the initial page visited in this subspace: in this case, the departmental home page. The label in the top right indicates which, if any, graphical filter is currently applied (section 7.1.1).

WEBNET's other facilities include the following.

Saving and Retrieving Web Subspaces. Users can save the web subspaces generated during a browsing session, providing a rich variant of Netscape's "Bookmarks" or Mosaic's "Hotlist." Having browsed a subspace once, and generated a corresponding representation of the subspace, users can save the subspace with an associated file name. Subsequently loading the file displays the entire subspace allowing one-click immediate

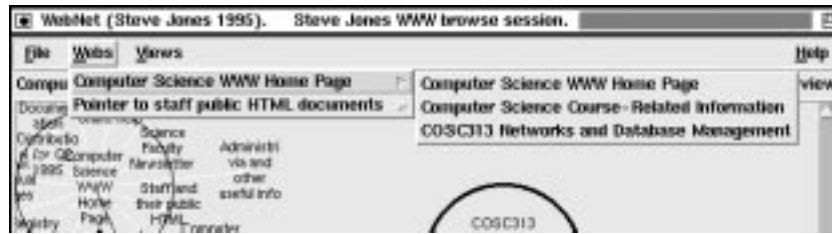


Figure 6: The WEBNET Webs menu after visiting two distinct subspaces.

access to a potentially large number of pages.

By abstracting away from the single-page bias of current browsers powerful and novel browsing paradigms may emerge. For instance, rather than recommending that a class investigate a particular URL, a class instructor could refer students to a pre-assembled WEBNETsubspace, with the recommended trails already provided.

Dynamic Page Menus. When a user enters a new subspace by directly accessing a page (for instance by typing a URL), the first page in that subspace is added to the *Webs* menu.

The *Webs* menu allows users to manage and access collections of distinct WWW subspaces. Users can quickly navigate between disjoint web-subspaces: a facility that may be useful when researching two or more distinct topics. As pages are accessed within a subspace, they are added to a cascading menu which is linked to the menu entry for the first page in the subspace. Consequently, the *Webs* menu contains entries for *all* visited pages, grouped into distinct subspaces.

Figure 6 shows two distinct subspaces navigated by the user in the current session: *Computer Science WWW Home Page* and *Pointer to staff public HTML documents*. When a menu entry is selected, the corresponding overview is displayed in the subspace overview area. Figure 7 shows a portion of the navigated staff subspace.

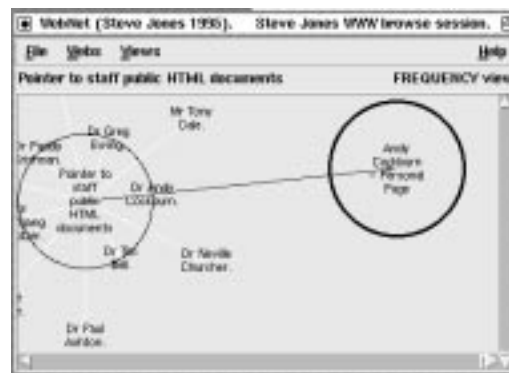


Figure 7: The WWW subspace displayed after selecting the staff subspace from the Webs menu.

Filtering the Graphical Overview. Information overload, or clutter, is a significant problem when representing complex information spaces in graphical format. Promis-

ing information suppressions and filtering methods such as distortion-oriented views (Schaffer *et al.*, 1996) are prominent research areas in Human-Computer Interaction. We have experimented with a variety of dynamic magnification techniques to emphasise the user's focus of interest, but the main filtering techniques supported by WEBNET are based on frequency of visits to pages, recency of visits to pages, and the distance of pages from the currently displayed page. These filtered views are selected through the *Views* menu.

The current view filter (if any) is shown by the label in the top right of the subspace overview. Each view alters the size of the nodes in the overview proportionately with respect to the selected criterion. The frequency view alters the size of the nodes with respect to the number of times that they have been visited during the current session. The fewer the number of visits, the smaller the node appears. Under the recency view, the most recently visited nodes are shown largest. The distance view diminishes the size of the nodes as their edge distance from the current node (the page shown in the browser) increases.

7.2 WEBNET design rationale

This section describes the design rationale that governed WEBNET's development. The three primary problems associated with current browsers, summarised above, are used as a framework for describing WEBNET's design rationale. Although much of WEBNET's design rationale is curative, focusing on overcoming problems in current browsers, it is also intended to be augmentative. WEBNET is designed to provide simple and powerful operations for page access and recall, using an appropriate presentation metaphor. Usability analysis has yet to determine the degree to which WEBNET satisfies these ambitions.

7.2.1 Incorrect mental models

As shown by the usability study, incorrect mental models of the browsers' behaviour are common. The simplest way to resolve this problem would be to improve each browser's system image to better reveal the underlying stack-based navigation model. An iconic representation of the *History-List* stack, for instance, could dynamically reveal the stacking behaviour: the growth of the stack as pages are loaded, the descent into the stack when pages are recalled, and the shrinking of the stack when loading causes pages to pop off the top of the stack. Such an iconic representation would provide an additional hook for the user's synthesis of a correct mental model, without a demand for large amounts of screen real-estate.

Even with correct mental models, however, users may still encounter problems with stack-based navigation. Section 5.2 noted that the natural affordances of web navigation were linear-incremental and temporal rather than stack-based. We contend that, even with correct mental models of system behaviour, user will still be periodically frustrated by their inability to return to pages (those popped off the history-list), and by the existence of duplicated pages on the stack.

WEBNET, in contrast, offers a full visual history of the user's navigation in a browsing session. Previous browsing sessions may also be recalled by loading the appropriate file. Thus, WEBNET provides external representations of the user's short-term and long-term memory of browsing sessions. By filtering the graphical overview, users can review various aspects of their navigation history, such as chronological ordering (recency) and

frequency of visits to nodes.

Graphical overviews such as those offered by WEBNET are only one potential solution to the problems of incorrect mental models. Low-fidelity interface solutions using standard interface widgets such as list-boxes or dynamic menus could readily provide access to full navigation histories, recency views, and so on. The simple techniques have two major advantages over graphical overviews: they require only small amounts of screen real-estate and they are less demanding of computational power. Their weaknesses lie in their poor support for the user's subspace context and for memory overload (discussed below).

7.2.2 Lack of Context

WEBNET uses graphical overview diagrams because they provide a natural navigation paradigm to resolve the problems of lack of context and of memory overload. The browsers examined in this study provide no situational cues to the user's position in the visited subspace. The subspace overview provided by WEBNET reveals a 'hub and spoke' model (Catledge & Pitkow, 1995) which allows users to return to 'hub' pages with a single mouse operation. Alternatively, the hub's 'spokes' can be browsed in sequence (one click per page) without the requirement that the user make redundant traversals back to the hub-page.

7.2.3 Memory Overload

Section 5.3 stressed that the user's memory is a fundamental, and highly fallible, resource when navigating with standard browsers such as Mosaic, Netscape, and tk-WWW. WEBNET's graphical overview diagrams replace much of the need for user-centred *recall* with user-centred *recognition*. Shneiderman (1987) emphasises the value of supporting "see and point versus learn and remember."

In WEBNET, navigation to a previously visited page requires the user to scan the visual display of pages, recognise the node, and click on it. In standard browsers the user must scan the *History-List*, normally by pressing the back button one or more times (direct selection from history-lists accounts for less than 1% of user actions (Catledge & Pitkow, 1995)) until the page is found, or until the user finds that the page is no longer on the stack. If the page is not on the stack, the user must try to remember and find the page that provided access to the desired one, then click the appropriate link, or remember the URL.

Beyond the problems of recalling pages that the user *has* visited, WEBNET's display of page links is designed to ease problems arising when the user *intends* to navigate to a page. For instance, while using Netscape, a user sees 'interesting' links p_{11} and p_{12} on page p_1 and intends to visit both, but while visiting a set of pages linked to page p_{11} , the intention of visiting p_{12} is masked out of short-term memory. WEBNET eases this problem by allowing the user to display the links that are available from all pages that have been visited. Thus, in the scenario above, pages p_{11} and p_{12} will both be displayed on the overview diagram, providing a graphical surrogate for the user's short-term memory.

8 Related Work

Related work on web-based usability analysis is sparse when compared to the work on development of new browsing paradigms. Nielsen and Sano (1994) describe a set of usability studies on web page design, and present guidelines for page designers. Related hypertext research is more prevalent, for instance, Bieber and Wan (1994) investigate backtracking in multi-window environments. Beiber and Isakowitz (1995) provide an overview of recent work on designing hypermedia applications.

In contrast to the lack of usability analysis, the number of commercial and non-commercial WWW browsers is increasing rapidly. Most support a model of navigation that is similar to that of Mosaic, Netscape, and tkWWW. This section briefly reviews some of the browsers that offer novel styles of navigation support.

Air Mosaic (or ‘Mosaic in a Box’) is a commercial modification to NCSA Mosaic which includes features such as hierarchical history-lists, folders and bookmarks, and saveable session histories. Internet Works provides access to visited pages through tabs within the browser window and via a Card Catalog system which allows catalogues of related pages to be created and saved for later use. OS/2 Warp: WebExplorer provides a WebMap (in indented ‘table of contents’ form) showing the path taken to visited pages which is, in effect, an enhanced history list. DeckScape (Brown & Shillner, 1995) offers a powerful variant of the card catalogue model. Pages are collected into decks, akin to file-management folders or directories, and users may leaf through the pages held within the decks. Decks are also used to encapsulate the results of operations such as “expand all the links on this page.”

These systems have not been analysed in this study, but emerging WWW navigation metaphors such as card catalogues and WebExplorer’s WebMap merit further investigation.

Systems supporting graphical paradigms for WWW navigation are under development at several research institutions. The Navigational View Builder (Mukherjea & Foley, 1995) creates overview diagrams by statically parsing the HTML pages in WWW subspaces. Users can filter the resultant representation of the subspace by categories such as file-size, author, and topic, but some of the filtering criteria require the incorporation of non-standard meta-data into the HTML. Although the Navigational View Builder supports a wide range of powerful visualisation techniques, it does little to assist the users’ *dynamic* navigation around the web: the overview diagrams are generated statically, before the user enters the subspace. Future work on the Navigational View Builder includes usability studies to determine the extent to which the additional hyperspace contextualisation assists the users.

The Hyperbolic browser (Lamping *et al.*, 1995) provides a “focus + context” (fish-eye) technique for visualising large information hierarchies. No mention is made of how a subspace representation is generated, but the user is provided with a magnified focus of interest (the current node), and a distance-dependent de-magnification of surrounding nodes. Animated graphical distortion is used to provide a smooth transition of the subspace representation as the user moves from one node to another. The Hyperbolic browser was used in a usability study involving a task of navigating to node representations of URLs with “the application intent being that a Web browser would jump to that node”. As yet, the Hyperbolic browser uses the web as a theoretical test-bed for navigation exercises. When integrated with an actual browser it will provide an powerful, though computationally expensive, platform for usability study.

Pad++ (Bederson & Hollan, 1994) provides another animated distortion-oriented

view of web navigation. In contrast to the Hyperbolic browser and the Navigational View Builder it is dynamic, providing a complete graphical history of the pages that the user has visited. All pages are displayed on a single display surface. Thus the current page, displayed at full magnification, is shown on the same surface as the icon-size representations of previously visited pages. Graphical links show the connections between pages in the history. The primary disadvantages of Pad++ are its restriction to showing only the pages that the user has visited (and not the links from them) and its high computational demands (due to the complex animated display).

MosaicG (Ayers & Stasko, 1995), like Pad++, provides a graphical history of the pages that the user has visited. This history is shown in a separate window using a tree-based structure and ‘thumb-nail’ iconic representations of each of the visited pages. The primary differences between MosaicG and WEBNET are that MosaicG does not represent the links available on a page (a feature we believe eases memory-overload), and MosaicG represents the N to N relationships between web pages as a tree while WEBNET uses a ‘hub-and-spoke’ representation. Future usability studies of MosaicG and WEBNET will reveal the relative merits and costs of these differences.

9 Summary

The introduction to this paper identified three implications of our analysis of WWW browser navigation facilities. First, that *page designers* need to fully understand the navigational features of client applications in order to best support their readers’ needs. In our usability study we observed one subject who had inadvertently provided useful navigational features on his page without knowing that he had done so. Presumably many page designers fail to provide these facilities for equally coincidental reasons.

The second implication of this study was that browser designers should consider the mapping from their model of system behaviour to users’ models of the system. We have noted the failed mapping from the implemented stack-based model, to a user model of navigation that is either temporal or linearly-incremental. This failed mapping suggests that work is required to clarify the system image of current browsers.

The final implication in this paper is that navigational problems can be ameliorated through user-centred aids for web browsing. We have designed and built a graphical browser (WEBNET) for the WWW that dynamically adapts to, and reinforces, the user’s browsing actions. The user’s short-term memory is reinforced by complete and accurate external representations of visited nodes, and traversal between pages is facilitated by direct manipulation of the graphical display. A network based interface metaphor is utilised to better support user navigation activities. Various filters of the graphical representation enhance the user’s ability to overcome visual information overload.

The focus of our on-going and future work is on evaluating the graphical navigation paradigm provided by WEBNET. We hope that the usability analysis and system development described in this paper will advance research on web navigation, and improve the facilities offered by future browsers.

Acknowledgements

This work was undertaken while S Jones was on sabbatical in the Department of Computer Science at The University of Canterbury, Christchurch, New Zealand, and was

partially funded by The Carnegie Trust for the Universities of Scotland. Many thanks to Carey Evans for helpful comments on the paper.

References

- Ayers, EZ, & Stasko, JT. 1995. Using Graphic History in Browsing the World Wide Web. *In: Proceedings of the Fourth International World Wide Web Conference. 11-14 December, Boston.* Also available from: <http://www.w3.org/pub/Conferences/WWW4/Papers2/270/>.
- Balasubramanian, V, Ma, BM, & Yoo, J. 1995. A Systematic Approach to Designing a WWW Application. *Communications of the ACM*, **38**(3), 47-48.
- Bederson, B, & Hollan, J. 1994. Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics. *In: Proceedings of ACM Conference on User Interface Software and Technology, 1994.* Also available from: <ftp://ftp.cs.unm.edu/pub/pad++/pad-uist94.ps.gz>.
- Bieber, M, & Isakowitz, T (eds). 1995. Designing Hypermedia Applications. *Communications of the ACM*, **38**(3), 27-29.
- Bieber, M, & Wan, J. 1994 (September). Backtracking in a Multiple Window Hypertext Environment. *Pages 158-166 of: Proceedings of the Fifth ACM Conference on Hypermedia Technologies (ECHT'94), Edinburgh, September 1994.*
- Brown, MH, & Shillner, RA. 1995. DeckScape: An Experimental Web Browser. *In: Computer Systems and ISDN Systems: Proceedings of the Third International World Wide Web Conference. 10-14 April, Darmstadt, Germany*, vol. 27. Also available from: <ftp://gatekeeper.dec.com/pub/DEC/SRC/research-reports/SRC-135a.ps.Z>.
- Bush, V. 1945. As We May Think. *The Atlantic Monthly*, **176**(1), 101-108.
- Catledge, LD, & Pitkow, JE. 1995. Characterizing Browsing Strategies in the World Wide Web. *Pages 1065-1073 of: Computer Systems and ISDN Systems: Proceedings of the Third International World Wide Web Conference. 10-14 April, Darmstadt, Germany*, vol. 27.
- Conklin, J. 1988. HyperText: An Introduction and Survey. *In: Greif, I (ed), Computer Supported Cooperative Work: A Book of Readings.* Morgan Kaufmann.
- Conklin, J, & Begeman, ML. 1988 (March). gIBIS: a hypertext tool for team design deliberation. *Pages 247-251 of: Hypertext '87* University of North Carolina, Chapel Hill, North Carolina, November 13-15 1987.
- Coulouris, G, & Thimbleby, H. 1992. *HyperProgramming.* Addison-Wesley.
- Isakowitz, T, Stohr, EA, & Balasubramanian, P. 1995. RMM: A Methodology for Structured Hypermedia Design. *Communications of the ACM*, **38**(3), 34-44.
- Lamping, J, Rao, R, & Pirolli, P. 1995. A Focus+Context Technique Based on Hyperbolic Geometry for Visualising Large Hierarchies. *Pages 401-408 of: Proceedings of CHI'95 Conference on Human Factors in Computing Systems* Denver, May 7-11 1995.

- Leung, YK, & Apperley, M. 1994. A Review and Taxonomy of Distortion-Oriented Presentation Techniques. *ACM Transactions on Computer Human Interaction*, **1**(2), 126–160.
- Marshall, C, & Shipman, FM. 1995. Spatial Hypertext: Designing for Change. *Communications of the ACM*, **38**(3), 88–97.
- Mukherjea, S, & Foley, JD. 1995. Visualizing the World Wide Web with the Navigational View Builder. *Pages 1075–1087 of: Computer Systems and ISDN Systems: Proceedings of the Third International World Wide Web Conference. 10–14 April, Darmstadt, Germany*, vol. 27.
- National Center for Supercomputing Applications, 1995. *NCSA Mosaic Common Client Interface. Version 1.1*. Available from WWW page: <http://www.ncsa.uiuc.edu/SDG/Software/XMosaic/CCI/cci-spec.html>.
- Netscape Communications Corporation, 1996. *Netscape Navigator Frames*. Available from WWW page: http://home.netscape.com/comprod/products/navigator/version_2.0/frames/index.html.
- Nielsen, J. 1990. The Art of Navigating through HyperText. *Communications of the ACM*, **33**(3), 296–310.
- Nielsen, J. 1993. *Usability Engineering*. Academic Press.
- Nielsen, J, & Sano, D. 1994. SunWeb: User Interface Design for Sun Microsystem's Internal Web. *In: Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*. Available from WWW page: <http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/HCI/nielsen/sunweb.html>.
- Nielsen, J, & Sano, D. 1995 (May). *Interface Design for Sun's WWW Site*. WWW page: <http://www.Sun.COM/sun-on-net/uidesign/>.
- Norman, DA. 1988. *The Psychology of Everyday Things*. Basic Books.
- Ousterhout, JK. 1993. *An Introduction to Tcl and Tk*. Addison-Wesley.
- Press, L. 1995. The Internet is Not TV: Web Publishing. *Communications of the ACM*, **38**(3), 17–23.
- Schaffer, D, Zuo, Z, Greenberg, S, Bartram, L, Dill, J, Dubs, S, & Roseman, M. 1996. Navigating Hierarchically Clustered Networks through Fisheye and Full-Zoom Methods. *ACM Transactions on Computer Human Interaction*, **In Press**.
- Scheifler, RW, & Gettys, J. 1986. The X Window System. *ACM Transactions on Graphics*, **5**(2), 79–109.
- Shneiderman, B. 1987. Direct Manipulation : A Step Beyond Programming Languages (excerpt). *Pages 461–467 of: Baecker, RM, & Buxton, WAS (eds), Readings in Human-Computer Interaction: A Multidisciplinary Approach*. Morgan Kaufmann.
- Thimbleby, H. 1990. *User Interface Design*. ACM Press, Addison-Wesley.
- Tufte, ER. 1984. *The Visual Display of Quantitative Information*. Graphics Press.

Vora, PR, Helander, MG, & Shalin, VL. 1994. Evaluating the Influence of Interface Styles and Multiple Access Paths in Hypertext. *Pages 323-329 of: Proceedings of CHI'94 Conference on Human Factors in Computing Systems* Boston, April 24-28 1994. Addison-Wesley.