WebView: A Graphical Aid for Revisiting Web Pages

Andy Cockburn[1], Saul Greenberg[2], Bruce McKenzie[1],
Michael Jasonsmith[1] and Shaun Kaasten[2]

[1]Department of Computer Science, University of Canterbury, Christchurch, New Zealand.
{andy, bruce, mpj17}@cosc.canterbury.ac.nz

[2]Department of Computer Science, University of Calgary, Calgary, Alberta, Canada.
{saul,kaasten}@cpsc.ucalgary.ca

Abstract

Current commercial web browsers such as Netscape Navigator and Microsoft Internet Explorer provide a wide and diverse range of utilities, such as history lists and bookmarks, that support revisiting previously seen pages on the web. Yet previous research indicates that these utilities are largely unused. In this paper, we present an alternative utility called WebView; a prototype designed to improve the efficiency and usability of page revisitation. It does this by paying particular attention to how previous pages are represented visually, and by integrating many revisitation capabilities into a single display space. Our preliminary evaluation of WebView indicates that users are enthusiastic about the functionality provided, and that it improves the efficiency of some navigational acts.

1 Introduction

Few, if any, technological advances have been as rapidly adopted as the World Wide Web. Fox (1999) notes that while radio and TV took thirty-eight and thirteen years respectively to attract 50 million users, the Internet took only four. User interfaces to web browsers have had little time for iterative refinement, and it should therefore be unsurprising that there are usability problems in even the most rudimentary facilities they provide.

In our previous related work we have shown that page revisitation—the act of returning to previously seen pages—is a fundamental part of web navigation. About 58% of all pages a person visits are to ones they have seen before, and use of the browser's **Back** button accounts for more that 30% of actions at the browser (Tauscher and Greenberg 1997). We have also shown that while many users have a naïve understanding of **Back**'s stack-based behaviour (Cockburn and Jones 1996), it is a reasonable (but not optimal) way to return to the most recently visited pages (Greenberg and Cockburn 1999). Returning to distant pages requires other revisitation schemes such as history lists and bookmarks. Many researchers believe these utilities could be improved and are now investigating alternate revisitation mechanisms.

In this paper we describe our own revisitation system called WebView and its preliminary evaluation. WebView is an add-on window to Netscape Navigator that presents an automatically generated graphical overview of the user's browsing paths. It provides a variety of facilities for navigational shortcuts, and it allows the user to tailor the display of a large set of pages. In designing WebView, we paid particular attention to two fundamental problems that all systems providing graphical navigational assistance must address. First, how can we assist the user in identifying previously visited pages, and second, what display organisation schemes can be used to enhance the visualisation of large sets of previously visited pages? The design issues associated with these two problems are the focus of a companion paper (Cockburn and Greenberg 1999).

The structure of this paper is as follows. Section 2 describes the general design objectives that have motivated and guided our development of WebView. Section 3 describes WebView's interface and its functionality. In Section 4 we evaluate WebView, and find both quantitative and qualitative indications of its usability. Section 5 concludes the paper and provides directions for further work.

2 Design Goals Motivating WebView

2.1 High ratio of utility to screen real-estate

Application windows that directly support a user's primary task, tend to be continually present on the display during use e.g. word processors, web browsers, and spreadsheets. In contrast, secondary windows that augment the task are raised and hidden as required. The problem is that iconified windows that are running in the background require small levels of overhead to bring them to the foreground, and this may be sufficient to preclude their use if the value of the information they provide is low. The 'History' windows of current browsers such as Netscape and Microsoft Internet Explorer demonstrate this issue—despite being readily accessible through menu or icon selection, they are rarely used. Tauscher and Greenberg (1997) indicate that less than 1% of web-browser actions are on the history list, and Hightower *et al.* (1998) indicated that the value is less than 0.1%.

To overcome this problem, we believe revisitation systems must have high utility as well as compact representation. In our design of WebView, we do not expect users to have it continually on display. We do, however, intend that WebView should provide a sufficiently usable, efficient and visually compact system so that users are willing to un-iconify it for anything more than the shortest navigational path.

2.2 Integrated navigation support for pages that are temporally near and far

Current commercial browsers support a wide range of navigation support features, but they are spread across a wide variety of user interface components. Navigation features of Internet Explorer 5 and Netscape Navigator include **`Back`** and **`Forward`** buttons, pull-down menus off these buttons, 'History' windows, pull-down URLs off the 'Address' or 'Location' text-entry box, and 'bookmarks' or 'favourites'. It is our intention that enhanced navigation systems such as WebView should integrate the functionality provided by these diverse features into a single unified system. A person should be able to use this integrated system to revisit pages regardless of whether they are temporally near (suitable for revisitation with the **`Back`** button) or far (requiring utilities similar to bookmarks or history).

We believe that two system capabilities are required to achieve this goal of integrated navigation support. First, the system should *automatically capture and record information* about all pages that the user visits. This is needed because users often do not know that the page they are currently viewing will be important to them in the future. Analogous observations have been made in studies of users' management of email (Mackay 1988), where the stereotypical behaviour of email 'archivers' is to keep copies of all email messages in case they become important at a later time. Furthermore, even when users *do* know that a page will be important to them later, they can forget to bookmark the page. By automatically capturing information about the page, the set of pages that the user must search through is enormously reduced (from the entire web to the set that the user has previously viewed).

Table 1: Data on bookmark collections.

	Total items	Max. depth	Num. folders	Items / folder mean	Items / folder max.
	113	3	18	6	21
	386	4	44	8	34
	541	4	85	6	27
	56	5	14	4	11
	104	3	13	8	18
	117	3	11	10	26
	85	3	4	21	80
	233	3	26	8	36
Mean	**204.4**	**3.5**	**26.9**	**8.9**	**31.6**
S.D.	*172.9*	*0.8*	*26.4*	*5.2*	*21.2*

Second, *lightweight and implicit* bookmarking techniques should be available to help the user return to important pages. The primary problems with current bookmarking schemes are that they require an explicit act from the user, and that they impose a management burden in reorganising their structure. Table 1 shows data that we collected on the bookmark structure of eight Computer Science academics (all of whom use Netscape Navigator). It is clear from the data that bookmarks require extensive management. The number of bookmarked items ranges from 56 to 541, with a mean of 204. All of the subjects had a hierarchical structure with at least three levels of depth. The right-hand column shows that the range for 'Maximum items in a folder' is from 11 to 80, with a mean of 31.6 (see also the bookmark study by Abrams *et al.* 1998). Bookmarks are normally accessed via pull-down menus, and despite research evidence that demonstrates that broad/shallow menu structures are preferable to narrow/deep structures (Landauer and Nachbar 1985), it is unlikely that menus with over thirty items in one level are efficient. Organising bookmarks is also problematic to the point that prior studies had described it as one of the top three Web usability problems, with users sometimes spending considerable effort re-organising them, or just deciding its not worth the effort (Abrams *et al* 1998). While bookmarks are valuable in principle, conventional browsers impose an excessive burden onto people wishing to use them: they require a user to decide upon a page's importance ahead of time; they must be organised to be effective (especially when there are many of them); and they require housecleaning as old, unneeded bookmarks clutter the space.

By automatically capturing information about every page the user visits, and by providing powerful searching and filtering schemes, we believe that systems can largely negate the need for explicit bookmarks. Furthermore, access to the user's most important pages can be enhanced by heuristics that infer information about the page through data such as visit counts. Finally, lightweight interface mechanisms that allow the user to explicitly mark a page as important should be provided.

2.3 Effective support for page identification, display organisation, and page retrieval

The previous two design goals raise conflicting requirements for systems that integrate techniques for web navigation. They must maximise the richness of the information display, and yet they must allow the user to view and access *all* previously visited pages, some of which will be unimportant to the user. To attain an acceptable compromise between information density and information overload it is necessary for systems to support the user in three ways.

Support for page identification. Any graphical scheme for page revisitation must provide a visual representation that aids page identification. The most common technique is to use the **`<Title>`** text extracted from the page's

representing pages with only the **<Title>** text, the consequence being that people can have problems finding a particular page in a list. The companion paper also described several alternative schemes for representing pages; the one used in WebView (Section 3) is to provide several redundant cues to page identification, including zoomable thumbnail images, a 'dogears' metaphor for the page visit count and for bookmarks, the page's **<Title>** text, and the page URL.

Effective display organisation. It is essential that the visualisation of a large set of pages is organised in a manner that aids browsing, searching and categorisation. Yet there is no easy answer on how best to do this: major issues are discussed in the companion paper (Cockburn and Greenberg 1999), and a survey of display organisation schemes is given in Cockburn and Jones (1997). WebView, described in Section 3, supports two of the more promising display organisation schemes: a temporal view, and a 'hub and spoke' view.

Powerful search facilities. Effective mechanisms for page identification and display organisation can increase the number of pages that the user can interact with and perceive in the display, but searching and information filtering schemes will be necessary to help the user restrict the number of pages displayed. As well as text-based searches for title and URL information, we also wish to investigate the usability of searches that are based on abstract page properties such as the page visit count, and the timing of the first and last visit to the page.

3 The WebView Prototype

WebView is an add-on window that interacts with unaltered versions of Netscape Navigator[1]. Whenever the user visits a page in Netscape, WebView's display is automatically updated to reflect the action. As with conventional systems, clicking on the text-title alongside any page makes Netscape navigate to the page. The following subsections describe WebView's techniques for page identification and page organisation.

3.1 Representing pages

WebView captures and displays a miniaturised zoomable thumbnail image of the rendered page (Figures 1-2), an approach also used in a few other research systems (Hightower *et al.* 1998; Ayers and Stasko 1995). It also detects the title and URL of the page, and these are (optionally) displayed alongside the thumbnail. Because some thumbnails may be difficult to distinguish from others (such as a site's pages that follow a standard look, we also provide larger views: mousing over any miniaturised thumbnail causes it to zoom to approximately four times the size (Figure 2a bottom).

3.2 Implicit and explicit bookmarking through dogears

WebView combines thumbnails with bookmarking cues through a 'dogears' metaphor (Figure 1). Dogears encode information about the number of visits to a page (an implicit bookmark based on the idea that frequently visited pages are somehow more important than others) and about page bookmarks (where a person marks a page explicitly). An implicit bookmark displayed as a 'dogear' at the top-left of each page grows progressively denser green with each visit to the page, allowing users to readily identify pages that they visit frequently. An explicit 'bookmarking' dogear is added to the bottom left hard corner of the thumbnail when the user clicks it with the middle mouse button.

Figure 1: Dog-eared thumbnail.

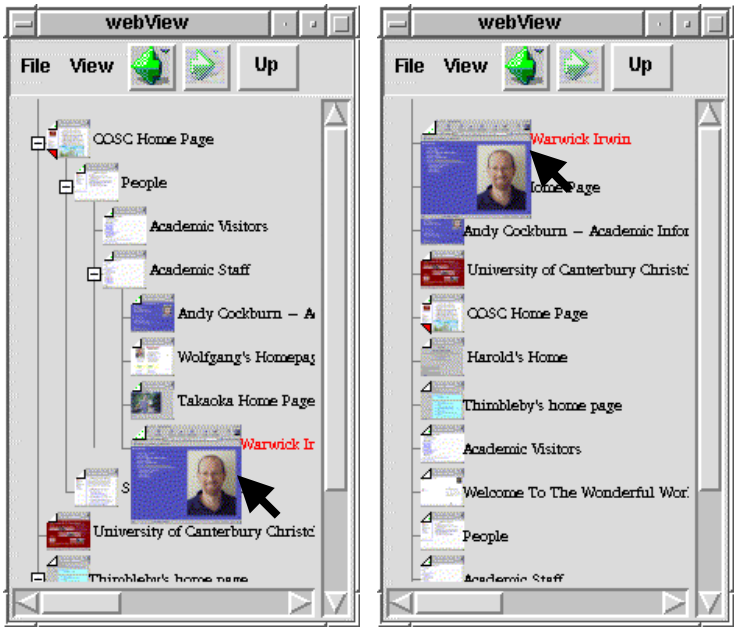
3.3 Organisation schemes: 'Hub-and-spoke', temporal display, and shortcut menus

WebView supports two primary display organisation schemes, controlled by options under its 'View' menu. Its 'hub-and-spoke' view (Figure 2a), displays the tree-like nesting relationship between the storage location of pages. In Figure 2a, for example, it is clear that the user has visited four pages under the "Academic Staff" page. Organising the display as a structural tree, such as this, is intended to help the user see the context of their navigational acts: pages on a similar topic are likely to be displayed in close proximity to each other. To avoid excessive nesting depth in the display, each new web site that the user visits is added to the display at the top-level of the nesting structure. The figure therefore shows that the "COSC Home Page", "University of Canterbury…" and "Thimbleby's home page" are each located at separate web-sites. Naturally, separating sites in the display will also separate pages on related topics that are stored at different sites. To ease this problem, WebView adds a feature that displays cross-site navigational links. The arrowhead links connecting thumbnails in Figure 3 shows that the user navigated from the "Academic Visitors" page to "Thimbleby's home page", and then to the page "Professor Ian. H. Witten…", each of which are stored at different sites.

[1] Interaction with unaltered versions of Netscape is achieved through Netscape's remote control facilities (**home.netscape.com/newsref/std/x_remote_proto.html**)

WebView's 'temporal' display organisation, shown in Figure 2b, provides a complete temporally ordered recency list of pages with duplicates removed. The back and forward buttons at the top of WebView's display operate on this temporally ordered data structure, and the 'Up' button has the same effect as the stack-based navigation scheme currently used by Netscape Navigator and Microsoft Internet Explorer (which provides an incomplete recency list). Greenberg and Cockburn (1999) provide details of the design issues and algorithm used in WebView's temporal display.

Finally, WebView parses the page's HTML contents, extracts the page links, and uses this information to construct a pop-up 'shortcut' menu associated with the thumbnail (Figure 4). Clicking on a thumbnail pops up the shortcut menu that can be used to navigate directly to any subordinate child page. The shortcut menu works within both temporal and hub-and-spoke views.



a) hub and spoke b) temporal ordering

Figure 2: WebView's two display organisation schemes.

4 Preliminary Evaluation

We performed a preliminary evaluation of WebView to gain insights into its usability and efficiency, and to direct subsequent development work. We used seven volunteer subjects, all graduate Computer Science students or tutors. All of the subjects used web-browsers as part of their daily work (6 using Netscape Navigator and one using Microsoft Internet Explorer), and all were very familiar with the structure and contents of the web site used for the evaluation tasks (shown in Figure 5). Given the preliminary nature of the evaluation, we wanted to see if it worked with knowledgeable 'resilient' users, and thus accepted obvious bias in our subject group.

Figure 3: Cross-site link indicators

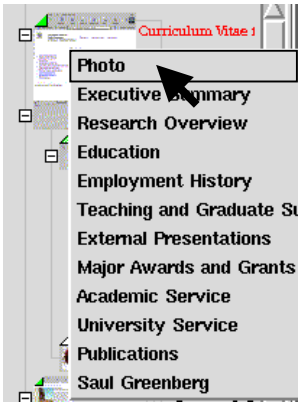


Figure 4: Shortcut pop-up menu

Each subject participated in a twenty-minute evaluation that compared WebView's navigation efficiency (in terms of task time) with that of Netscape. Only WebView's temporal view was used during the tasks (Figure 2b); we will evaluate the 'hub and spoke' view in subsequent studies. Subjects each received a five-minute introduction to the system. They then carried out two navigational tasks, each with two parts. A stopwatch was used to time all tasks. After the tasks, they answered three questions and provided general comments.

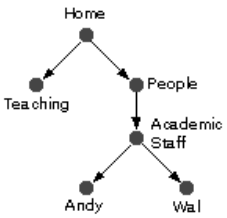


Figure 5: Web pages used in the evaluation tasks.

Task 1 compared Netscape's support for navigation with that provided by WebView. In Task 1a, without a WebView client running, the subjects were instructed to navigate to the "Teaching" page. At this point they were instructed to "Visit Andy's page, and then Wal's page, as quickly as possible". They were asked to confirm that they understood the task, and then they were told to start. The stopwatch was stopped when they reached "Wal's" page (assuming they had previously displayed "Andy's" page).

In Task 1b, a new Netscape session was started, and a new WebView client was launched i.e., only the root page was included in WebView's list. The subjects were asked to complete an identical navigation task to 1a (to visit "Andy's" then "Wal's" page from the "Teaching" page), but they were instructed to do so using *only* the navigational features provided by WebView. Although WebView is primarily designed to support page revisitation, its shortcut menus and other capabilities allow a wide range of navigational actions. We felt that this task would provide insights into the usability of many of WebView's capabilities, even though the task is outside

its primary usage scenario. The risk of a learning effect between tasks 1a and 1b was minimised by having the subjects navigate through the test web pages during the pre-test introduction.

In Task 2a, the subjects were asked to use WebView to return to the "Teaching" page from "Wal's" page, and in Task 2b they performed the same action using Netscape. This revisitation task is close to WebView's primary usage scenario.

Subjects then answered the questions in Table 3.

Table 2: Evaluation results: timings for Tasks 1 and 2 (seconds)

Subject	Task 1a (Netscape)	Task 1b (WebView)	Task 2a (WebView)	Task 2b (Netscape)
1	14.2	30.2	3.8	6.8
2	22.4	26.6	3.0	7.3
3	15.4	18.0	1.0	4.5
4	20.0	31.9	6.5	28.9
5	68.0	21.0	1.0	7.0
6	15.4	22.4	1.3	8.4
7	20.6	58.3	1.4	4.6
Mean	**25.1**	**29.8**	**2.6**	**9.6**
S.D.	*19.2*	*13.5*	*2.0*	***8.6***
T Test	**t(6)=0.48, p = 0.65**		**t(6)=2.71, p = 0.035**	

4.1 Observations and Results

Five of the seven subjects were enthusiastic about the system, three of them extremely so, making statements such as "It'd be great to have a system like this." The two subjects who were not enthusiastic about the system were primarily concerned about the redundancy introduced by having two different ways of navigating (the "Netscape way" and the "WebView way"). This is not a problem, as the ultimate intention with systems such as WebView is that they should replace, rather than duplicate, the current set of navigational facilities.

Table 3: Questionnaire and responses

Responses using a five point Lickert scale: from 1 "not at all" to 5 "very"							
Subjects:	**1**	**2**	**3**	**4**	**5**	**6**	**7**
How useful were thumbnail images in recognising pages?	4	2	2	2	3	2	4
How usable were the link shortcuts menus off the thumbnails?	2	4	3	5	5	5	3
If available, how likely would you be to use a system like WebView?	2	4	4	5	5	5	2

Users readily adapted to most of WebView's navigational facilities. For instance, in Task 1b, when navigating to "Wal's" page from "Andy's" page, all users immediately selected the shortcut menu off the "Academic Staff" thumbnail *without* returning to the "Academic Staff" page. Table 2 shows that there were no significant differences in the mean times for task completion in Task 1a (Netscape) and Task 1b (WebView) (two-tailed paired T-Test, $p > .1$). This lack of difference is good news, for although WebView was designed to favour page *revisitation*, this indicates that people can use it to navigate to *new* pages as well as old ones without excessive penalty.

In Task 2a all users were significantly faster in using WebView to return to the "Teaching" page from "Wal's" page. The mean times for Netscape and WebView were 9.6 seconds and 2.6 seconds respectively, giving a statistically significant improvement ($t(6) = 2.7$, $p < .05$, two-tailed paired T-Test).

Table 3 shows that subjects 2 to 6 were enthusiastic about the system, responding positively to questions two and three. Interestingly, the only subjects who thought the thumbnail images were useful in recognising pages were the ones that rated the system poorly (1 and 7). When asked for comments about the thumbnails, several subjects stated that the thumbnails would probably be more useful when navigating through sites that they were unfamiliar with. Subject 4 stated that he had not found the thumbnails useful because the tasks were introduced verbally, using the same names for the pages as was displayed in the text associated with each page.

We observed two primary usability problems with WebView. First, in Task 1b when using WebView's pop-up shortcut menus, the subjects often took time searching through the menu for the desired link. In contrast, when using Netscape in Task 1a, the users knew immediately where the desired link would be displayed. There are no obvious ways to ease this problem, but we are not overly concerned about it. The pop-up menus provide powerful shortcut facilities that promise to increase the efficiency of navigation, and we view the time taken to search for links in the menu as a necessary design compromise in providing these facilities. We also suspect that the subjects would have spent more time searching for links in Netscape if they had not been so familiar with the web space used in the study.

A second problem affected subject 7 severely (also noted by subject 6 during the pre-test introduction). In Task 1b, subject 7 repeatedly accessed the wrong shortcut menu: for example, he popped up the "Homepage" menu several times when he needed to access the "Academic Staff" menu. In comments after the task he reported that he found the movement of thumbnails extremely confusing, and that he wanted the thumbnails to "stay in the same place" (WebView automatically moves the current page to the top of the display to maintain its temporal list of pages). He

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.