The Explorer bar: Unifying and improving web navigation

Scott Berkun
Program Manager
One Microsoft Way
Redmond, WA 98052
scottber@microsoft.com

ABSTRACT

The Explorer bar is a component of the Internet Explorer web browser that provides a unified model for web navigation activities. The user tasks of searching for new sites, visiting favorite sites, and accessing previously viewed sites are simplified and enhanced by using a single user interface element.

Keywords

Navigation, bookmark, design, hypertext, searching, browsing

INTRODUCTION

The World Wide Web provides access to an enormous amount of information and resources. The primary usability issues with using the web involve insufficient support for helping users find and return to individual web pages. The Explorer bar was designed to improve these usability problems by providing a single well designed user interface model for the most common set of web navigation tasks.

The design problem

Three of the most problematic common end user tasks on the web are: searching for new web pages, returning to a bookmarked/favorite page, and returning to a particular non bookmarked page (Abrahms, D 1997, Pitkow, J 1996,). For users with existing web browsers these tasks were often cumbersome, and sometimes impossible.

Bookmarks or Favorites lists do help the user by providing a mechanism for remembering pages, but as that list grows the value of the mechanism decreases (Abrahms, D. 1997). It is also known that users frequently need to return to pages that are relatively close in chronological order but browsers may not be designed to support these needs (Tauscher, L. 1996).

The design solution

We separated the entire set of web documents into more digestible groups that users would understand. We settled on three groupings: web pages the user had not seen before, web pages the user was interested in, and web pages the user had visited before. We gave these groupings the names Search, Favorites and History for easier reference. As we sketched out different ideas for solutions to the most common navigation problems, we recognized that there were strong similarities for how users might interact with these sets of information.

For example, when searching the web using a search engine, users received a long list of search results. In formal and informal usability studies, we observed many users click on a result link which took them to a new site, click on several links on that site, and then recognize it was not the site they wanted. They would then use the back button repeatedly to return to the list of results and repeat the process with the next search result. We observed similar behavior when users were trying to find a particular item in their Favorites or history lists. The critical problem was a loss of context. There was no easy way for the user to return to an important page during a searching process, or skip between multiple items in their Favorites list to find the one they needed. This behavior is often referred to as spoke and hub navigation (Tauscher, L. 1996).

This indicated significant value in representing these different groups of web pages in a similar way. The user could then learn a set of concepts once, and apply it to all of these different types of web pages. We felt if we developed the right general model for navigating through lists of items, we could improve the usability of the most common navigation tasks performed with a web browser.

General model

We started by looking at existing mechanisms (Dennis E. Egan et al. 1989) for navigating through lists or hierarchies of information. In particular, we examined the Windows 95 file management utility called Windows explorer. It displayed a hierarchical view on the leftmost 20% section of the screen, and a viewing area comprising the remaining 80%. The left area acted as a map or overview for what was viewed on the right. Clicking on an item in the left section caused the main area to navigate to the selected item. This was useful for advanced users who were familiar with the file system hierarchy.

In the context of web browsing, we found several problems with Windows explorer design. The hierarchical view on the left never displayed folder contents. If the user clicked on a folder, it would navigate the main window to show a list of the items in that folder. This behavior was useful for file system maintenance, where the detailed information of each file is important. However, it forced the user to frequently navigate away from whatever you were viewing before.

This problem helped us recognize that we needed to help the user keep track of the relationship between the current page they were on, and the page they were trying to locate. We established a general principle: for web navigation: the explorer bar should have it's own context. The bar would only navigate the main window if the user clicked on a specific page. If the user clicked on a folder or group of pages, it would open the folder in place to show the available pages. This allowed the user to view potential navigation targets without losing their current place in the web browser.

Search

Helping users find new web pages was a particularly problematic area (Pitkow, J 1996). Search results are provided by the user's choice of web search provider and each provider had control over how results were presented to the user. We knew that we were restricted to providing a framework for searching to work in, and that to a degree we were dependent on the providers to do a good job with much of the searching experience. We worked with the providers to develop a set of guidelines for search bar content that they would follow to obtain some level of consistency.

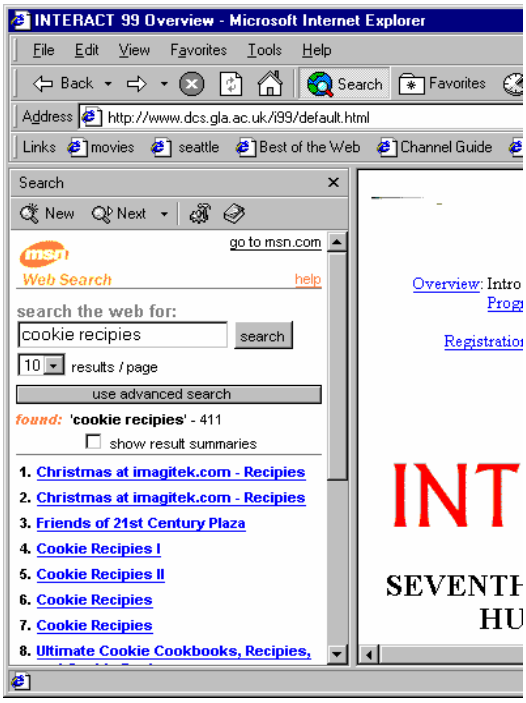

Figure 1. Searching using the Explorer bar

The first step was to figure out how to compress search results pages down to a size that would fit in the bar. We started with a design that could fit 10 search results inside the Explorer bar, which approximated the number found on regular search results pages. During our exploratory tests of a prototype in the usability labs, all five intermediate participants were able to complete basic search tasks on the first try using the explorer bar. Tasks included generating a new query, looking through multiple results pages, and clicking on different result links. The limited real estate forced us to leave out result information such as the URL or sample text. This caused problems in cases where there were poorly titled pages, and the user had no way to even guess which page was the best one without trying them all.

In response to this problem, we enabled an unused HyperText Markup Language (HTML) attribute called TITLE, to set the tooltip property for a search result link. When the user moved their mouse pointer over a search link, a small window would appear that provided additional information, such as file size, Internet address, or text abstract. This helped offset the small screen real estate available to the bar, and helped users decide what link to use before clicking on it. We saw evidence that when noticed, the tooltips improved user performance, however 2 of the 8 subjects did not even notice the tooltips. In all cases users were still able to complete their tasks, but without discovering the tooltips some tasks took longer. We accepted this as a reasonable tradeoff since we needed to balance the number of results we could fit, with how much information we could expose for each result. Tooltips were the only option we knew of that didn't consume more

real estate.

The toolbar area of the search bar provided a 'next provider' button that allows the user to recast the current query to a different search provider without retyping, increasing the speed of using multiple sources. We also provided a "new search" button that when clicked showed the user a list of available types of searches, grouped by task instead of by provider. This gave the user a way to recover from server problems, or broken content from a search provider. One special item in this list was a display of the ten previous queries they had entered. We did not have specific data on the recurrence of search queries, but Abrahms indicated that bookmark lists often contain references to searches.

History

There was strong evidence that improved access to viewed web pages would help users (Tauscher, L. 1996). The challenge was to develop a simple way to organize the large lists of history data that would allow users to quickly find the specific pages they wanted.

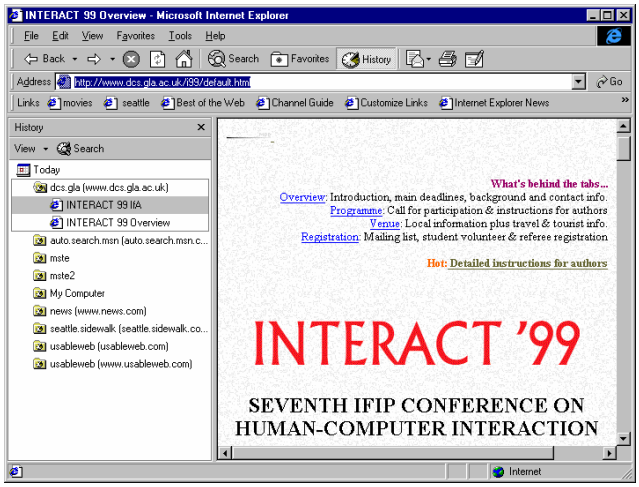

Figure 2. History using the Explorer bar

We started with a simple structure for the history list, organizing the data chronologically and then by site. This made it possible for a user to access a site simply by knowing when they visited, and then walking through the sites visited on that day. Figure 2 shows the History bar in use, with the time and site fields visible. In addition, the history list intelligently groups visited web pages based on their second level domain name, which would help speed locating a specific site from a list. We added this feature in response to watching users struggle to find sites with non-standard domain names (www1.microsoft.com) in a list that was alphabetized by the entire URL.

In the second usability study for history, seven participants with beginner to intermediate level web experience were given navigation tasks that required returning to pages previously visited at different time periods. All of the users were able to complete the navigation tasks using a combination of the history bar, the back button, and on occasion some of the navigation assistance provided by the site's themselves.

We also designed the history list to act as constant indicator for where the user is currently located on the web. By showing the list of web domains, and indicating in the list where the user was with a gray bar, we were able to give the user some context for where they were on the web. In the future we want to measure how effective this mechanism is for expediting use of the history bar, and in reducing the user symptom of feeling lost in cyberspace.

To cover cases where a strict chronology was useless, such as when a user remembers the site they were on, but not when they went there, we added the ability to change the history list. A dropdown menu in the history bar called 'view' allows the user to change the ordering to be sorted by date, by site, by frequency, or by the exact chronological order each individual pages was visited. To support scenarios where only text from the page is remembered, a searching feature is provided that searches the text of every page the user has visited that is still in the user's cache. For any page that has a match, its title is displayed conveniently in the history bar allowing the user to easily try out different result hits.

Favorites

The user behavior of navigating through existing favorites showed many of the same issues as navigating through a large history list (Abrahms, D. 1997). Even though the user created the Favorites hierarchy, the larger the list grew the harder it became to find an individual item. We used the same general model for Favorites as we had for History. This made it possible to easily navigate from one favorite to another, and open or close folders without losing the current web page in the main window. We also provided in place organization of favorites: the user could drag and drop items between folders, add items to favorites, and remove favorites items while viewing particular web pages. This was mostly of value to intermediate and advanced who were familiar with the drag and drop convention.

Unfortunately, most novice and intermediate users were not aware that this convention could be used in the Explorer bar. Nearly half of the 8 beginner and intermediate users tested on favorites tasks would open the favorites bar and fail to find a way to create folders or move items into existing folders. To improve access to these commands from the Explorer bar, we used a toolbar strip underneath the title of the explorer bar to display command buttons.

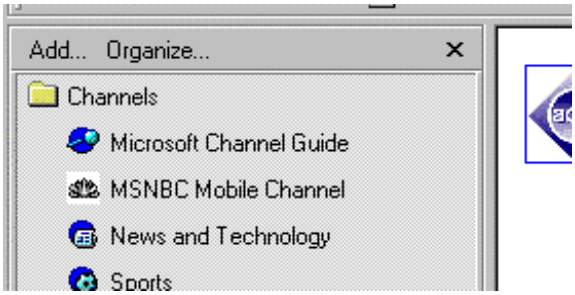

Figure 3. Favorites using the Explorer bar

We experimented with different visual elements, and text descriptions using an informal paper prototype with users of different experience levels. The simple approach of using the text label for each command was most effective, provided that the text labels for the commands would fit in the available space. In the final design, clicking on the Add button displayed the add to favorites dialog, and clicking on the organize button displayed the organize favorites dialog. We verified the success of these buttons in a smaller usability test after the toolbar had been added. All of the participants were able to get to the add and organize dialogs through the Favorites bar.

Conserving real estate

The explorer bar forced users to choose between maximizing the screen area for the web page they are viewing, and maximizing their ability to navigate using the explorer bar. We experimented with different sized bars, and found 200 pixels to be a balanced tradeoff between effective page viewing and web navigation. If we made the bar larger, many websites could not be viewed easily without horizontal scrolling. If we made the bar smaller, it was impossible to view lists of sites without horizontal scrolling. We knew based on survey data (Pitkow, J 1996) that the majority of the population that could report resolution was using 800 by 600 or better screen resolution. The explorer bar used 200 pixels allowing 600 for page viewing. Users with larger resolutions would have a full 800 or greater pixel width for web pages. We also made the explorer bar resizable to allow the user to customize the size to account for particularly troublesome pages.

To increase the user's control over available real estate, we provided a mode of the browser called fullscreen. This removed all secondary menus, status and cosmetic elements, thereby maximizing the screen real estate for the content of the page. When in fullscreen mode, the explorer bar assumes a special behavior called autohide. The bar slides away off the left edge of the screen. The user can bring the bar back by moving her mouse to the left most edge of the screen. The bar then slides back onto the screen and can be used to navigate the browser to another page. As soon as the user has clicked on an item, or moved his mouse away from the bar, the bar slides back off the left edge of the screen. A toolbar button and a menu command were provided to toggle in and out of fullscreen mode.

For some tasks, such as moving between multiple search results or history items, it is useful for the bar to remain visible even while in fullscreen mode. To enable this, we added a small pushpin button to the title area of the explorer bar. This button allowed the user to pin the bar in place for as long as necessary. When depressed, the bar would stay visible. If pressed a second time, it would return to autohide behavior. Figure 5 shows the pushpin button.

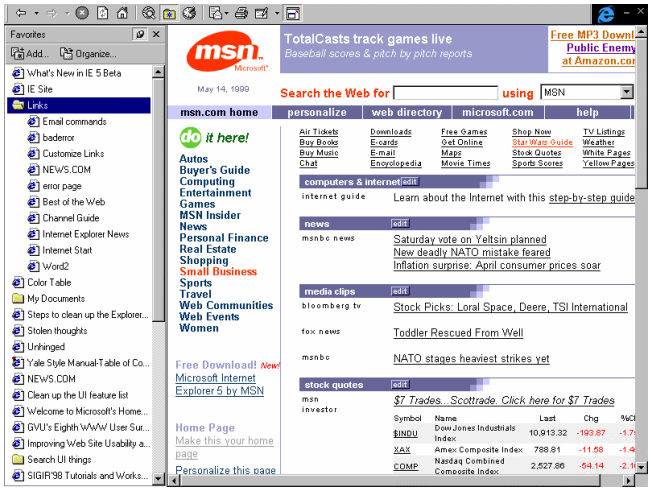

Figure 4. Fullscreen mode in Internet Explorer

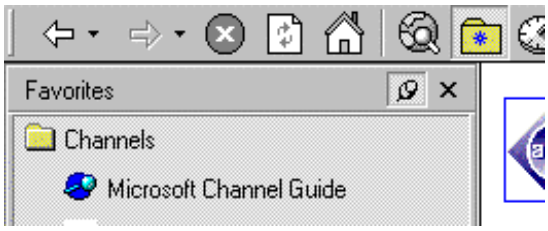

Figure 5. The pushpin button in fullscreen mode

Activating and closing the bar

There was a conflict between providing a single user interface element, and making sure that the critical features of the browser were easily discoverable. We experimented in sketches with having one toolbar button for activating the entire set of bars, one button with a drop down list for each bar, and relying purely on the menus for activation of each bar. We could not think of one button label that could describe all of these functions in a way that would be sensible to users. The best compromise we found was to allow each function that used the bar to have it's own toolbar button. The buttons would be mutually exclusive of each other, allowing the user to switch from

one bar to another. Each button also acted as a toggle switch, turning a particular bar on or off.

In usability tests of initial designs we found that users often had trouble discovering how to close the explorer bar. Nearly half of the 8 intermediate users failed on their first few attempts to make the bar go away. They would complete the task of finding a specific page, but would not recognize that the active toolbar button in the toolbar worked as a toggle switch. We reinforced the discoverability of the toggle behavior by adding a close box to the title area of the explorer bar. This provided a distinct visual affordance for closing the bar, and in follow up tests nearly all participants were able to close the explorer bar.

Problems with the Hierarchy View and Scrollbars

We examined the standard Windows 95 treeview control and saw two places where changes might improve the usability of navigating web pages. Removing the plus element for opening folders, and changing the limit on the number of folders allowed to be opened simultaneously.

Since the most common action users applied to a folder in the Explorer bar was opening or closing, it followed that the largest visual target on the folder should provide those actions. The standard treeview design has a small plus to the left of the folder name that was the only way to open or close the folder. We modified the treeview in the Explorer bar so that if a folder name was clicked, it opened. To select a folder for renaming or deleting, you needed to right click on it. We removed the plus element to simplify the treeview and make the most frequent task the easiest to perform.

Once we had a working prototype of the Favorites and History bars, we discovered in our own usage that it was easy to get bogged down in the number of open folders. The user would either have to manually go back to close folders they were no longer using, or use the scroll bar repeatedly to maneuver around them. We accounted for this by automatically closing unused folders. If the user opened folder A, and then opened folder B, we would automatically close folder A for them. This did create some negative side effects. For advanced users, this behavior made it more cumbersome to move items from one folder to another. Since advanced users were the minority, we decided to default to autoclosing folders. We added a switch to the program options for advanced users to turn this feature off.

We needed a user interface element inside the explorer bar to allow it to scroll. Large lists of bookmarks are common and cause obvious problems (Abrahms, D 1996). We found that the standard scroll box consumed a large amount of real estate and was visually unappealing inside the Explorer bar. We copied the scrolling affordances used elsewhere in Internet Explorer 4.0's toolbars and menus, which provided a simple arrow at the top and bottom of the list whenever it was necessary.

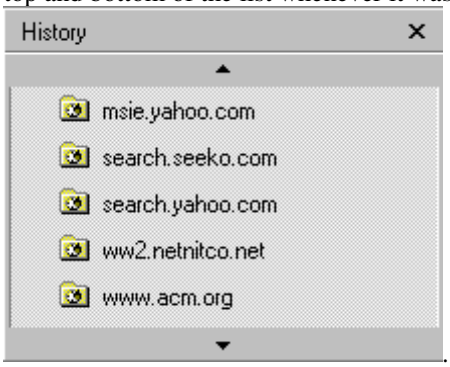

Figure 6. The new scroll bar style

After usability testing, we discovered that this new scrolling model had many problems. Unlike the standard scroll bar, this new affordance did now allow for easy paging of long lists, or for granular control over the pace at which items were scrolled. We reverted back to using the standard scroll bar control in version 5.0 of Internet Explorer.

Usability Testing Summary

Over the course of two versions of Internet Explorer (4.0 and 5.0), 6 different usability studies were conducted on different aspects of the Explorer bar. In some tests only specific features were examined, such as favorites or Search feature, or specific concerns were examined such as the ability for users to close the explorer bar. In many instances usability tests for other aspects of the product touched on Explorer bar issues and provided additional, though often more anecdotal, information.

Each usability study used from 5 to 10 participants, depending on the sophistication of the test design. The studies used a mixture of user backgrounds, ranging from windows 95 users with beginning web experience (little or no experience with Internet), to advanced (high experience with internet). Verbal protocol was used as one of the primary methods of data collection, except in cases where we performed benchmarking or performance comparison tests between two different prototypes. In those cases time on task and error frequency were the only primary measures.

Future Applications

After our initial success wth the explorer bar, we did some limited explorations into potential other uses. We added the ability to create horizontal explorer bars, than ran across the top or bottom of the browser. We expected that communication tasks such as chats or reading news information would work better in the horizontal form factor.

The major roadblock to other uses of the explorer

bar is the mutual exclusion rule for each kind of bar (vertical or horizontal). To keep the interface simple, we allowed only one vertical explorer bar to be active at any time. For example, if the Favorites bar is active, and the user clicks on the History button, the Favorites bar goes away and is replaced by the History bar. Horizontal bars follow the same rule when other horizontal bars are involved. However, the user is allowed to have one vertical bar and one horizontal bar at the same time. In general, mutual exclusion was the only design we could think of to keep turning individual bars on and off from becoming a complex task. Search, Favorites and History were critical parts of the user experience and we did not want to complicate those core functions in the name of enabling less frequent user tasks.

The vertical bar is most useful for navigation tasks. Any time the user needs to pick from a list of items, and move between them frequently, the vertical bar provides value. The ability to keep a separate context is likely to help the user stay on task and keep useful context. Good examples are table of contents lists, troubleshooting information, or help content. We considered moving Internet Explorer's help system to use the explorer bar, but hit the mutual exclusion problem: you couldn't view the help information for the favorites bar, and the actual favorites bar at the same time. Sitemaps for websites could work well in an explorer bar as well, provided there were guidelines or conventions for how sites design them. We experimented with sitemaps during IE4 but removed the feature for schedule and other reasons (Berkun, 1996).

We provided the ability for other software developers to add their own explorer bars to Internet Explorer. We expected that certain websites that are used as launching points, such as portals, could use an explorer bar to speed the user's access to specific pages or parts of the portal.

Conclusions

In working on the Explorer bar we recognized four themes:

- There is value in applying the same user interface to diverse data sets provided the usage patterns for each are similar.
- Using a large percentage of real estate is acceptable to users if you are solving an immediate problem and providing discoverable ways to customize or deactivate items.
- History can be a very effective tool for web navigation if you provide users with ways to mine useful data out of the large pile of history information.
- Standard user interface elements often have been thoroughly designed. Do not stray from them unless you have exceptional and well understood reasons. For example, reusing the standard scroll bar was the best choice for the Explorer bar. However, in the case of the treeview, we had strong evidence that something different was required for the user tasks we were designing for.

ACKNOWLEDGMENTS

There were many critical people that made this work possible: Steve Capps, Walter Smith, John Cordell, Chris Franklin, , Chris Nyman, Gayna Williams, Lisa Sanford, Shawna Swanson, Jennifer Shetterly, Shawn Murphy and many others on the Internet Explorer 4.0 and 5.0 development teams. Without the major impact these individuals had, the Explorer bar concept would never have been realized.

REFERENCES

Abrahms, D. (1997) *Human Factors of Personal Web Information spaces,* MS Thesis, Department of Computer Science, University of Toronto. http://www.dgp.toronto.edu/~abrahms

Berkun, S. (1996) Sitemaps in Internet Explorer 4.0, Microsoft Interactive Developer magazine, http://www.microsoft.com/Mind/1196/preview1196.htm

Catledge, Lara, Pitkow J, (1995) *Characterizing Browsing strategies in the world-wide web,* Third International World Wide Web Conference, , http://www.igd.fhg.de/www/www95/papers/80/userpatterns/UserPatterns.Paper4.formatted.html

Dennis E. Egan, Joel R. Remde, Louis M. Gomez, Thomas K. Landauer, Jennifer Eberhardt and Carol C. Lochbaum, Formative Design-Evaluation of SuperBook, *ACM Transactions on Information Systems*, Research Contributions, vol. 7, no. 1, 1989, pp. 30-57.

Gloor, Peter A. (1997), *Elements of Hypermedia Design: Techniques for Navigation and Visualization in Cyberspace,* Birkhauser publishing

Horn, Robert (1990) *Mapping hypertext: The analysis, organization and display of knowledge for the next generation of on-line text and graphics,* Lexington Publishing

Nielsen, J. (1996) *Multimedia and Hypertext* AP Professional, Cambridge, MA 02139

Pitkow, J (1996) GVU's 5th WWW User survey. http://www.cc.gatech.edu/gvu/user_surveys/survey-04-1996

Tauscher, L. (1996). *Evaluating history mechanisms: An empirical study of reuse patterns in World Wide Web navigation.* MSc Thesis, Department of Computer Science, University of Calgary, Alberta, Canada. http://www.cpsc.ucalgary.ca/grouplab/papers/

Tauschler L. and Greenberg, S. (1997) How people revisit web pages: Empirical Findings and Implications for the design of History systems. *Int Journal of Human Computer Studies,* 47(1),95-138

Wurman, Richard. (1989) Information Anxiety