Using Automated Tools to Improve Web Site Usage by Users with Diverse Abilities

Melody Y. Ivory  
*University of Washington*

Jennifer Mankoff  
*University of California - Berkeley*

Audrey Le  
*University of California - Berkeley*

Follow this and additional works at: [http://repository.cmu.edu/hcii](http://repository.cmu.edu/hcii)
Using Automated Tools to Improve Web Site Usage by Users with Diverse Abilities

Melody Y. Ivory
Jennifer Mankoff
Audrey Le

Abstract

The World Wide Web plays an important role in our society—enabling broader access to information and services than was previously available. However, Website usability and accessibility are still a problem. Numerous, automated, evaluation and transformation tools are available to help Web developers build better sites for users with diverse needs. A survey of these automated tools is presented in the context of the user abilities they support. Furthermore, the efficacy of a subset of these tools based on empirical studies is discussed, along with ways to improve existing tools and future research areas.

Melody Y. Ivory is an Assistant Professor in The Information School at the University of Washington.

Jennifer Mankoff is an Assistant Professor in the Electrical Engineering and Computer Science Department at UC Berkeley.

Audrey Le is an undergraduate Cognitive Science Major at UC Berkeley.

In addition to thanking the survey respondents and study participants, special thanks are also due to the following: Maya Draisin, Jerry Evas of SpiderMetrix.com, Beth Kolko, Mary Deaton, Sheryl Burgstahler, the DO-IT program staff, distribution list moderators (CHI-Web, SIGIA, SIGHCI, and many others), Judy Ramey, Marti Hearst, Aline Chevalier, Chantrelle Nielsen, and the anonymous reviewers.
The World Wide Web has become the predominant means for communicating and presenting information on a broad scale. Unfortunately, Website usability and accessibility continue to be pressing problems. An estimated 90% of sites provide inadequate usability (Forrester Research 1999), and an estimated 66% of sites are inaccessible to users with disabilities (Jackson-Sanborn et al. 2002). Although numerous assistive devices, such as screen readers and special keyboards, facilitate Website use, these devices may not improve a user’s ability to find information, purchase products and complete other tasks on sites. For example, sites may not have links to help blind users skip over navigation bars, or sites may not enable users to increase the font size of text, so that they can read it.

An abundance of design recommendations and guidelines exist to potentially help designers build sites that are more usable and accessible than sites built without such guidelines (See Comber 1995; Computer Science and Telecommunications Board 1997; Detweiler and Omanson 1996; Flanders and Willis 1998; Levine 2000; Spool et al. 1999 and Web Accessibility Initiative 1999). The current trend seems to be to develop new guidelines to address aspects that are unaddressed by existing guidelines. However, there are some fundamental problems with the existing guidelines, including: (1) guidelines are often difficult to understand and to apply (Borges et al. 1998; Chevalier and Ivory 2003; de Souza and Bevan 1990; Scapin et al. 2000), (2) the needs of different types of users may conflict, and there is little guidance on how to address these conflicts and (3) designers may not be able to address all guidelines at authoring time.

A number of automated tools have been developed to help designers to interpret and to apply guidelines, as well as to help users dynamically modify sites to accommodate their needs. There are over 30 automated tools, such as WatchFire Bobby (2002), UsableNet LIFT (2002) and the W3C HTML Validator (World Wide Web Consortium 2001) for evaluating Web site usability, accessibility, coding, etc. (Ivory and Hearst 2001; World Wide Web Consortium’s Web Accessibility Initiative 2002). These tools analyze the HTML code of Web pages to determine if it conforms to a set of guidelines—referred to as guideline review. A small subset of tools, the LIFT tools in particular, also assist designers with modifying the HTML code to conform to guidelines. In addition to these evaluation tools, there are numerous automated tools, described in Huang and Sundaresan (2000), Kaasinen et al. (2000) and Mankoff et al. (2002), for filtering and transforming Web sites. For example, these tools may facilitate Web site use by people with motor impairments (Mankoff et al. 2002) or visual impairments (Huang and Sundaresan 2000).

Despite the potential benefits of automated evaluation and transformation tools (Schneiderman and Hochheiser 2001), little evidence about their efficacy exists, specifically whether they: (1) from a designer’s perspective result in better sites than those produced without tools; and (2) from a user’s
perspective result in sites that are more usable and accessible than those produced or accessed without tools. The authors’ objectives in this article are to:

1. Characterize the types of user abilities that are currently addressed by automated tools.
2. Discuss findings from empirical studies that examine the effectiveness of these tools.
3. Highlight future research needed to better support diverse user abilities via automated tools.

To begin, Section 1 presents a survey of automated tools, characterizing the types of users they currently support. (Note that this article does not include a discussion of automated transformations to support users of small-screen devices; support for small-screen devices is a related area, but it is orthogonal to the problems caused by varying user abilities.) Next, Section 2 presents an empirical study of automated evaluation tools is described that shows that the tools themselves are difficult to use and, furthermore, suggests that the tools did not improve user performance on information-seeking tasks. This is perhaps the first such study of automated evaluation tools. Next, Section 3 presents ways to expand and improve the automated transformation tools such that they make the Web more usable by users with diverse abilities are discussed. Specifically, automated transformations of Web pages to accommodate the needs of low- and medium-bandwidth users are discussed (i.e., users who have no or limited use of a mouse and keyboard). These requirements are currently not supported in the automated transformation space.

SECTION 1: SUPPORTING DIVERSE USER ABILITIES

While the need to support blind users on-line—for example, specifying alternative text for images—is often discussed (Shneiderman and Hochheiser 2001), there is a broader range of user abilities that need to be supported. Table 1 provides a synopsis of this range of user abilities and the current support provided by automated tools.1 See the World Wide Web Consortium’s Web Accessibility Initiative (2001) for a detailed discussion of these user abilities. The categories shown in Table 1, are “Mouse Use,” “Keyboard Use,” “Vision,” “Hearing,” and “Cognition.” Each category refers to user capabilities. Motion is separated into “Mouse Use” and “Keyboard Use” for the following reasons: these two devices are almost exclusively used to control a computer, they are required by different aspects of Web browsing interfaces, and they depend on motor skills in different ways.
### Table 1: User Abilities Addressed by Automated Evaluation and Transformation Tools

<table>
<thead>
<tr>
<th>User Ability</th>
<th>Automated Evaluation Tool</th>
<th>Automated Transformation Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mouse Use</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Partial      | • Pennsylvania’s Initiative on Assistive Technology 2001  
               • UsableNet 2002  
               • WatchFire 2002 | • Hermsdorf 1998            |
| None         | • Pennsylvania’s Initiative on Assistive Technology 2001  
               • UsableNet 2002  
               • WatchFire 2002 | • Lynx-me 2002  
               • Novak 2002  
               • Internet Explorer™ Web Accessories 2002  
               • Kennel et al. 1996  
               • Brown and Robinson 2001  
               • {textualize;} 2002  
               • Betsie 2002  
               • James 1997  
               • Asakaw and Itoh 1998  
               • Christian et al. 2000.  
               • Zellwager et al. 1998. |
| **Keyboard Use** |                           |                              |
| Partial      |                           | • Mankoff et al. 2002         |
| None         |                           | • Pen for Windows™, Christian et al. 2000 |
| **Vision**   |                           |                              |
| Partial*     | • Ivory and Hearst 2002  
               • Websat 2001  
               • LiFT 2002  
               • WatchFire 2002  
               • {textualize} 2002  
               • Internet Explorer™ Web Accessories 2002  
               • Hanson et al. 2001 |
## Table 1: Tools for Specific Disabilities

<table>
<thead>
<tr>
<th>Disability</th>
<th>Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hearing</td>
<td></td>
</tr>
<tr>
<td>Partial</td>
<td>RetroAccess 2002</td>
</tr>
<tr>
<td>None</td>
<td>RetroAccess 2002</td>
</tr>
<tr>
<td>Cognition</td>
<td></td>
</tr>
</tbody>
</table>

*Partial vision includes low vision and color blindness.

**Partial comprehension includes learning and memory impairments.

There are obvious gaps in the work summarized in Table 1, such as the lack of evaluation tools that address reduced keyboard ability or the lack of transformation tools that support the hard-of-hearing, despite the increasing use of multimedia on the Web. Less obvious is the fact that many of these tools only partially address the domain of problems to which they apply. Web accessibility is a largely unsolved area with much work still remaining.
1.1 Automated Evaluation Tools

Fairly recent Section 508 standards require Federal agencies to make their electronic and information technology, including Web sites, accessible to people with disabilities (Center for Information Technology Accommodation 2002). Consequently, automated evaluation tools appear to be gaining acceptance as a means for improving the accessibility and usability of Web sites. For example, the World Wide Web Consortium’s guidelines on evaluating Web sites for accessibility (World Wide Web Consortium’s Web Accessibility Initiative 2001) recommend that designers use two of three automated evaluation tools (Bobby WatchFire 2002; WAVE Pennsylvania’s Initiative on Assistive Technology 2001; A-Prompt Adaptive Technology Resource Center 2002) to identify and correct potential accessibility problems during the design process. Many of these tools analyze the HTML coding to ensure that it conforms to accessibility or usability guidelines.

Most of the automated evaluation tools check whether sites conform to the Section 508 guidelines, the W3C Content Accessibility guidelines (Web Accessibility Initiative 1999), or both. There is a great deal of similarity between these two sets of guidelines, because the Section 508 guidelines are based on the W3C guidelines. (See Thatcher 2002 for a side-by-side comparison.) Overall, they address all the user abilities depicted in Table 1, but they focus on enabling access, rather than on improving user experiences. To address these limitations, Coyne and Nielsen (2001) proposed an additional 75 guidelines (e.g., minimize the use of graphics or small font sizes for text, avoid pop-up windows, separate browser windows or cascading menus, and minimize the number of links on a page), derived from studies of users with vision and mobility impairments. These guidelines are embedded in the Nielsen Norman Group Edition of the LIFT tool LIFT-NNG (2002).

Fundamentally, most tools check for conformance to the Section 508 or W3C guidelines and then generate a report that provides some guidance for repairing pages. Figure 1 depicts example reports generated by the Bobby (WatchFire 2002) and Doctor HTML (Imageware, Inc. 2001) tools. Bobby checked for Section 508 conformance, and Doctor HTML used a custom set of usability and HTML coding guidelines.

Only a few tools provide assistance with making changes. For example, the LIFT tools (UsableNet 2002), 508 Accessibility Suite (Macromedia 2001), AccVerify/AccRepair (HiSoftware 2002), and A-Prompt (Adaptive Technology Research Center 2002) provide wizards to walk developers through code modifications. Figure 2 depicts a task flow for evaluating and repairing a Web page using the LIFT-NNG tool within Macromedia Dreamweaver.
Figure 1: Bobby (Top) and Doctor HTML (Bottom) Reports Generated for the Same Web Page
**Figure 2: Screenshot of the LIFT-NNG Tool within Macromedia**

*About this report*

**508 Bobby.** If the Section 508 issues listed below do not apply to your page, then it qualifies as Bobby Section 508 Approved and you are entitled to use the Bobby Section 508 Approved Sticker. To obtain the logo and learn how to place it on your page, visit the [Bobby Guidelines page on the WebMedia website](http://www.bobby.org).

Follow the links in guideline title for detailed information about the error.

### Section 508 Accessibility

#### Section 508 User Checks

User checks are triggered by something specific on the page, however, you need to determine manually whether they apply and if applicable, whether your page meets the requirements. Bobby Section 508 Approval requires that all user checks pass. Even if your page does not conform to these guidelines they appear in the report. Please remember that all of them:

1. If you can't make a page accessible, construct an alternate accessible version.
2. If only sheets are supported, use style sheets only if they are still available and usable.
3. If a table is used for layout only, identify headers for the table rows and columns. (7 instances)
   Lines 61, 12
4. If you use color to convey information, ensure the information is also represented another way. (4 instances)
   Lines 14, 15, 22, 23, 24, 25, 26, 37, 39, 63, 65, 88, 90, 121
5. Make sure any changes in the position of elements on the page are accompanied by the correct JavaScript. (6 instances)
   Lines 14, 23, 24, 25, 26, 27, 63, 65, 88, 90, 121
6. If a form includes required information but the user does not fill it out in the alternative text, provide an external description. (6 instances)
   Lines 31, 61, 12
7. If you have more than one list of items or sub lists, provide a structural index to identify their hierarchy and relationships. (9 instances)
   Lines 31, 61, 12
8. If a sound, process is about to expire, give the user notification and a chance to extend the timeout.

The following 7 items are not triggered by any specific feature on your page, but are still important for accessibility and are required for Bobby Section 508 Approved status:

*Copyright © 2002 WebMedia Corporation. All rights reserved. Use of this software is subject to the Bobby Software License Agreement.*

---

**Doctor HTML's Report**

If this report does not produce a navigation bar at the bottom of the page, then it is incomplete and you should check the following areas:

- **Summary**

  | Table Analysis | 1 | There was 1 table structure problem found. |
  | Spelling | 12 | 12 potential spelling errors were found. |
  | Browser Support | 6 | There are 7 browser support conflicts in your document. |
  | Document Structure | 10 | There was 1 document structure problem found. |
  | Meta Tags | 1 | There is 1 potential metadata error in your document. |
  | Squash HTML | 0 | A Squash HTML page that is not downloadable by 12.0 to 4.9 is available here. |
  | HTML Parse | 0 | There were no HTML parsing errors detected. |
The boxed numbers and arrows in Figure 2 denote the task flow for a developer who wants to evaluate a page, where:

1. Shows how the developer initiates an evaluation of the page.
2. and 3. Show how to explore the evaluation results and detailed descriptions. Specify additional guideline information.
4. and 5. Show how to explore additional guideline information.
6. and 7. Show how to use the repair wizard to make appropriate modifications.

AccessEnable (RetroAccess 2002) is a Web-based tool that can make some changes to pages automatically; the developer can then download the modified pages. PageScreamer (Crunchy Technologies 2002) and InFocus (SSB Technologies 2002) are desktop applications that can also make changes. Most evaluation tools are either Web-based (e.g., AccessEnable and WebSAT (National Institute of Standards and Technology 2001)) or are desktop applications (e.g., AccVerify/AccRepair and Bobby). LIFT, the 508 Accessibility...
Suite, and AccVerify/AccRepair plug into other applications, such as Dreamweaver, FrontPage, and WinRunner. LIFT, the 508 Accessibility Suite, and AccVerify/AccRepair plug into other applications, such as Dreamweaver, FrontPage and WinRunner. Bobby and WAVE overlay guideline violations on the actual Web page, as shown in Figure 3. Bobby flagged areas for manual reviews with a “?” whereas WAVE used several icons to flag potential issues and also depicted the page reading order.

Next, the types of evaluations that the tools conduct for each of the user abilities depicted in Table 1 are described.

Mouse use: A few tools check to see if Web pages enable other forms of input (e.g., keyboard or Braille device use) to address the needs of users with some or no ability to use a mouse. For example, Bobby and WAVE look for keyboard shortcuts for forms and links, or for equivalent ways to handle events like mouse rollovers. However, the tools do not analyze the shortcuts to see if they are meaningful, consistent with common interface shortcuts, or conflicting with each other (e.g., the same shortcut may be specified for more than one action). Bobby and WAVE also check for an explicit tabbing order on pages, but there is no analysis of the tabbing order, for instance to see if it is logical. LIFT-NNG checks for the use of cascading menus, which the developers determined to be problematic for users with disabilities (Coyne and Nielsen 2001).

Keyboard use: To accommodate limited keyboard use, automated tools could check to see if Web pages enable speech input, rather than physical input (keyboard, Braille device or mouse input). Although the tools check for the ability to output speech (e.g., checks for the presence of alternative text for images, labels for form elements, and the like), they currently do not assess the ability of users to employ speech-based navigation. Additionally, these tools do not consider the needs of users for whom keyboard input is limited or slow. For example, a person using single switch input might prefer to see all menus on-screen as radio buttons. This might not be the most space efficient or aesthetic interface for a user with normal keyboard and mouse use. These conflicting user needs show that authoring-time solutions may not always be appropriate, and thus that automated transformation tools are a necessary component of true universal access.

Vision: Nearly all guidelines embedded in the existing automated evaluation tools address the needs of users with little or no vision. Most checks entail looking for the existence of page and frame titles, alternative text for images and other objects and headings for tables and labels for form elements, so that screen readers can process this information (Adaptive Technology Research Center 2002; Bowers 1996; Crunchy Technologies 2002; National Institutes of Standards and Technology 2001; Pennsylvania’s Initiative on
Figure 3: Web-Page Overlays Generated by Bobby (Top) and WAVE (Bottom) Tools
Assistive Technology 2001; RetroAccess 2002; UsableNet 2002; Vorburger 1999; and WatchFire 2002). Several tools provide wizards to help developers correct
such violations (e.g., Adaptive Technology Research Center 2002, HiSoftware 2002, Macromedia 2001 and UsableNet 2002). Other tools, such as the ALTifier (Vorburger 1999) and A-Prompt (Adaptive Technology Research Center 2002), suggest alternative text based on the page context and other heuristics. LIFT-NNG also checks for the presence of table summaries. None of the tools, however, can assess whether alternative text, titles, labels, headings or table summaries are meaningful to users.

Bobby, AccessEnable, and LIFT-NNG check whether pages enable users to skip repetitive navigation links. However, the tools do not check whether the skipped links actually take the user to the appropriate area. LIFT-NNG also checks to see if the page launches separate browser windows, which can impede navigation.

AccessEnable, Doctor HTML, A-Prompt and WAVE conduct rudimentary analysis of table and form structure. For example, A-Prompt depicts the linear reading order of table contents, and WAVE depicts the linear reading order of the entire page as shown in the bottom half of Figure 3 above. Developers have to manually evaluate the reading order to determine if it is logical or efficient, because automated tools do not support this type of analysis.

These tools perform several checks to address the needs of users with partial vision, such as analyzing color usage (Ivory and Hearst 2002 and National Institutes of Standards and Technology 2001), analyzing font and image sizes (Ivory and Hearst 2002; UsableNet 2002) and checking for the separation of content from presentation via stylesheets (RetroAccess 2002; WatchFire 2002; World Wide Consortium 2002). For example, the W3C CSS Validation Service (World Wide Consortium 2002) enables developers to see if their stylesheets conform to W3C CSS coding standards. As another example, Bobby checks to see if pages use relative positioning rather than absolute positioning, which facilitates screen magnification.

The WebTango prototype (Ivory and Hearst 2002) analyzes the quality of text and panel color combinations based on Murch's (1985) study of color combination preferences: it reports the number of good, neutral, and poor color combinations along with the specific combinations used. It also reports on the sizes used for images and text: these sizes are compared to statistical models derived from an analysis of over 5300 Web pages (Ivory and Hearst 2002). Similarly, the WebTango prototype reports ranges for the number of links, number of words within each text link, and the number of graphics. It also reports on the use of text and link clustering and on-text alignment, which may be helpful to users with partial vision. The WebTango prototype is a rudimentary research system, which not currently robust enough for wide use. Both tools checked the page for Section 508 conformance.

LIFT-NNG checks for small text, too many outgoing links, too short text links or button labels, too small buttons, and too many graphics. Accompanying discussions of guideline violations provide some crude thresholds for these aspects. The tool also checks to see if form fields are stacked in vertical columns,
and if there is white space between vertically aligned text links.

**Hearing:** Both the W3C and Section 508 guidelines require text equivalents for audio, video, or other multimedia formats. In addition, they require synchronization of these text equivalents with their sources. Most of the tools check for alternative text for objects, which screen readers can read to a vision-impaired user (see preceding discussion). However, there are no checks to see if this alternative text is easily available to a hearing-impaired user. Furthermore, the tools cannot check whether text equivalents or alternative text are consistent with their sources. AccessEnable seems to be the only tool that can flag whether there is some synchronization between the text equivalents and their sources, although it is not clear how this is accomplished, because this is difficult to check automatically.

**Cognition:** W3C and Section 508 guidelines address animation, which can cause problems for users with attention deficits. Bobby, WebSAT, and the Dreamweaver Accessibility Suite check for animated images or scrolling text. The WebTango prototype also quantifies the use of animated images and provides ranges, based on an analysis of over 5300 Web pages. None of the tools can assess whether there are ways to stop animations. To address the needs of users with comprehension impairments, the guidelines suggest that pages be well-structured and consistent within a site. For example, Bobby, the LIFT tools and others check that pages and frames have titles, which can help to orient users. WAVE checks the use of structural tags, such as headings and lists, to determine if they are used properly. The WebTango prototype quantifies and provides ranges for the quantity of text, links, images and other elements as a way to address potential cognitive overload. It also assesses download speed, consistency, the use of link and text clustering, and reading complexity—all of which may impede Web site use by users with comprehension impairments. Similarly, Doctor HTML checks for spelling errors and analyzes images (e.g., image size and number of colors). WebSAT also reports the total size of graphics.

LIFT-NNG, WebSAT, and the WebTango prototype evaluate other text, link, and image formatting that may impede Web site use. For example, WebSAT reports the following: Web pages that use non-default link colors, the use of fewer than 2–5 words in text links, line breaks before or after links, horizontal rules, or separate browser windows. WebSAT also reports a ratio of links to words, which may provide some indication of text density. LIFT-NNG reports on the following: use of small text, text links or button labels with fewer than six characters, too many outgoing links (more than twenty) or images, long cascading menus, separate browser windows, and inadequate space between vertically aligned text links.

The automated tools assess many aspects of web pages that could have negative effects on comprehension, but the key aspect that is missing is an
analysis of the content itself. None of the tools provides concrete insight about the content’s quality, cohesion, appropriateness, readability and the like. Although the tools may be able to improve the presentation of good content, they cannot improve on poor content. Humans need to make such assessments, but automating some aspects could be beneficial.

In Summary: As discussed in the preceding sections, automated evaluation tools address many user abilities. However, the emphasis of these tools is heavily on supporting blind users and users with comprehension impairments, at least to some degree. The tools currently provide little or no support for users who cannot use a mouse or who have hearing impairments. The tools do not appear to be robust enough to handle non-HTML implementations of Web sites, such as implementations in Macromedia Flash, dynamic HTML and other technologies. Furthermore, one of the major limitations of these tools is that they do not analyze content, even though content is considered most important (Brown and Robinson 2001).

Although it appears that the evaluation tools support a range of users, the survey does not report on the effectiveness of automated evaluation tools. An evaluation of three popular tools is discussed in Section 2 below.

Although automated evaluation tools can address some issues at design time, it cannot address all issues (e.g., accommodating users for whom keyboard input is limited or slow). Thus, integrating automated transformation and evaluation tools may be a viable solution to design time challenges. For example, if a transformation tool could dynamically collapse navigation levels and expose only one level at a time, then this transformation might improve Web site use by vision-impaired, cognition-impaired and other users. On the other hand, it is still necessary to evaluate the usability and accessibility of transformed pages; thus, there needs to be a tight integration between automated evaluation and transformation tools. This integration would enable designers to determine how robust their sites are to such transformations, for instance.

1.2 Automated Transformation Tools

Automated transformation tools are software applications that help end users, rather than Web site developers. These tools dynamically modify Web pages to better meet accessibility guidelines or the needs of specific users. Unlike automated evaluation tools, automated transformation tools do not make recommendations. They simply make changes to the content the user is accessing. The most common, automated tools were developed to support blind users (Betsie 2002; textualize; Fuller 1999; James 1997; Asakaw and Itoh 1998; Kennel et al. 1996; Miura et al. 2001; Oogane and Asakawa 1998 and Takagi and Asakawa 2000). Next most common are tools developed to maintain privacy or to remove unwanted advertisements and flashing and animated...
Fecha del documento: Winter 2003

Our survey also includes:

1. A set of accessibility tools for navigation and reading from the Trace R&D Center (Novak 2002);
2. The Internet Explorer plug-in that supports zooming, highlighting, and other features (Internet Explorer web accessories 2002);
3. Two projects supporting users with low vision (Brown and Robinson 2001; Hanson et al. 2001);
4. Our work on low-bandwidth accessibility (Mankoff et al. 2002);
5. Zellweger et al.’s (1998) work on dynamic link annotation;
6. Christian et al.’s (2000) voice-controlled Web browser; and
7. Two systems that focus on general mechanisms that can potentially support a wide range of users (Huang and Sundaresan 2000; Hermsdorf 1998).

Our intent is not to mention every tool, but rather to highlight the gaps and illustrate how some needs have been addressed. Additionally, this discussion leaves out technology that is not specific to Web browsing.

Automated transformation tools fall into two major categories, client-side tools and server-side tools. Client-side tools are special Web browsers (e.g., Internet Explorer web accessories 2002). They may modify HTML content directly, but more commonly, they provide alternative input mechanisms or render content in new ways. Server-side tools are commonly proxies—Web servers that dynamically modify content from other Web sites before it is sent to the client (e.g., Mankoff et al. 2002).

Mouse use: Mouse use may be reduced for technological reasons (e.g., neither a UNIX shell nor a cell phone supports mouse input) or for physiological reasons (either a motor impairment or blindness may make it difficult to use a mouse). Standard Web clients almost unanimously assume mouse input (an exception to this is text-only clients such as Lynx (Lynx Web Browser 2002). Without a mouse, it can be time-consuming to select a link of interest (access time can be approximated as a linear function of the number of links on a page). Some links are difficult or impossible to get to with only a keyboard. Image maps are also difficult to access. Moreover, even for accessible pages, usability problems arise, because most browsers are not designed with keyboard access in mind. Suppose users open a Web page, use the space bar to page down, and identify a link of interest. They will have to tab through all of the links starting at the top of the page to get to the destination link, because use of the space bar does not change the current link focus, which defaults to the first link of the page. Users may even lose their spot, as the browser moves the viewpoint to match each link they tab through.

Automated tools appear not to have solved this last problem, excluding the work presented in the section on “Improving Automated Transformation
Tools.” Tools developed for blind users have addressed other navigational problems, because blindness also implies difficulty using a mouse (Lynx-me 2002; Brown and Robinson 2001; {textualize;} 2002; Betsie 2002; James 1997). For example, table navigation is a problem germane to blind users, who must navigate a table to play sounds describing its elements (Novak 2002; Table Navigation JavaScript). For nested tables, finding a particular cell, or even simply browsing a table, can be quite difficult, especially with no visual feedback.

Tools may support faster navigation by providing a summary window containing document headings or document links (Novak 2002; Kennel et al. 1996; Internet Explorer Web accessories 2002). Document headings may allow the user to navigate to portions of a document with speeds based on the number of headings in the document, rather than on its number of links. Because the number of headings is usually smaller than the number of links, this can help to speed up navigation to a known target. Navigation based on a summary containing only links can speed up access for blind users, who may wish to jump to a specific link, without hearing all the intervening text before it. Another alternative that requires no mouse input is speech-based navigation, wherein users send voice commands to control browsers (Christian et al. 2000). Lastly, displaying information about link targets can help save users the time of laboriously selecting a link that turns out to be a dead end (Zellweger et al. 1998).

Note that very few of these tools directly address the problems of users with partial mouse capabilities. For example, URLs are notoriously difficult to click on because of their tiny height (Fitts’ law highlights this issue, even for users with full mouse capabilities). Worden et al. (1997) and Fraser and Gutwin (2000) have demonstrated techniques for handling this in graphical user interfaces for users with motor impairments and low vision, respectively. WebAdapter provided some support for near misses (Hermesdorf 1998), but apparently, no Web client has experimented with the full range of possible support features or applied them to clicking on links.

**Keyboard use:** As stated above, standard Web browsers assume both mouse and keyboard use. Most Web browsing naturally does not require a keyboard, but on the occasions when a keyboard is required for Web browsing, completing forms for instance, browsers do not provide alternatives. An exception is Microsoft Pen for Windows™, an operating-system-level solution that translates handwriting to text anywhere typing is normally expected as input. Also, Christian et al. (2000) have experimented with speech-based Web browsing. Both of these solutions require users to deal with recognition errors, and the former requires as much dexterity as typing. For users with an accent or speech impairment, as well as a motor impairment, these tools are not a solution.

Most commonly, users deal with these problems with the help of a generic, on-screen keyboard that knows little about the context of use. Only a
few server-side tools currently exist to eliminate the need for keyboard input. Our tool, developed to support this need, is discussed in Section 2 below.

**Vision:** The majority of automated transformation tools focus on some or no vision (Betsie 2002; Lynx-me 2002; textualize; 2002; Huang and Sundaresan 2000; Takagi and Asakawa 2000; Hanson et al. 2001; Mynatt and Edwards 1992; Trewin and Pain 1997; Oogane and Asakawa 1998; Kennel et al. 1996 and Miura et al. 2001). Such tools make it possible for screen readers to more easily render entire Web pages in audio by dynamically creating ALT tags for images that do not have them, and the like. Tools that support users with some reading (i.e., screen magnifiers) focus on making text larger and more legible (Brown and Robinson 2001, (textualize;) 2002 and Internet Explorer Web accessories 2002).

This is an active research area, and many unsolved problems remain, particularly for users with low vision. Magnification significantly reduces the amount of text that can be displayed, leading to problems similar to those faced by users of small-screen devices (Kaasinen et al. 2000). In general, automated tools are not good at generating usable interfaces (Myers et al. 2000), a necessity in this domain, where user interfaces must be translated into entirely different modalities or deal with radically different space constraints.

**Hearing:** There appear to be no automated tools that focus explicitly on hearing issues. This is due in large part to the lack of audio in use on Web pages. Additionally, without transcriptions of audio (the equivalent of ALT tags for images), it would be almost impossible, given current technology, to automate the presentation of audio data in other modalities for people who have difficulty hearing. Where such transcriptions are present, the task is of course much easier.

**Cognition:** For users who experience difficulty comprehending content (because of unfamiliarity with the language or because of a reading disorder, for example) or who have attention deficits, some of the most useful tools are those that simplify content, for example, by reducing distractions.

For those with attention-deficits, flashing, animation and other techniques commonly used by advertisers may be a serious distraction. Tools that remove advertisements—and simplify Web pages by disabling animated GIFs and by removing background images and music—include Muffin, Proxomitron and Web Cleaner (Muffin filtering proxy server 2002; The Proximitron—Universal Web Filter 2002; Webcleaner 2002).

For people with reading difficulties, highlighting and color modifications can both be helpful (Internet Explorer Web Accessories 2002; Brown and Robinson 2001). Color modifications may include changing font and background color, which can help some people with dyslexia to read more comfortably (Gregor and Newell 2000). Readers may perform highlighting as they scan a
document to support comprehension, as reading occurs or when text is revisited. Some GUI tools designed to support dyslexics read passages of text aloud, so it is conceivable that tools designed for those with no vision may also be useful for people with that condition. Summarization may also be useful for people who have difficulty comprehending large volumes of text. Current publicly available tools may summarize links or headings (Internet Explorer Web Accessories 2002; Novak 2002).

None of the tools listed above was intentionally designed to support people with cognitive impairments, and because of this, many of them represent only partial solutions. For example, a Web page that must be heard, but is not seen, generally includes no images, and may not support mouse-based navigation through the audio rendering. In contrast, someone who has difficulty reading might benefit from images and certainly would want to jump around what is read spatially rather than linearly. Thus, existing screen readers would not represent an adequate solution. Sites that require responses or task completion within a certain amount of time may also cause problems for some people with cognitive impairments; automated transformation tools do not address this issue.

**In Summary:** There are automated transformation tools available that support a wide range of user abilities, at least to some degree. The area most heavily investigated is support for users with no vision, and if authors have taken small steps at authoring time to support accessibility, these tools are reasonably functional. However, to become more than research prototypes, automated transformation tools must be robust across a broad set of Web pages and services. Even if it were possible to identify the content of any image without ALT tags, or to provide access to arbitrary plug-ins, Javascript, and Java applets, existing tools would still be limited. In terms of robustness, recent work by Fairweather et al. (2002) has begun to address this issue by creating a system that combines server-side and client-side techniques.

Outside the realm of blind users, even more work remains. As noted above, there is no direct support for users with limited mouse use on the Web, little support for users with limited keyboard use, and no tools that directly target people with cognitive impairments (within the confined domain of Web browsing). In actuality, some of the techniques developed for one category of user are often applicable to others, but investigation is required to determine if a tool can be used as is, or if it requires modifications. Additionally, because most users require an integrated support package, technologically it would be very difficult for them to combine multiple proxies or clients to select some techniques from each of them.

1.3 Conclusion of this Survey
This survey revealed that neither existing automated evaluation tools nor existing automated transformation tools cover the full range of user abilities. But how do the tools perform in practice for the abilities they do support? Next, data on the performance of automated tools is presented. It will be shown that the data supports the authors’ claims that the automated tools are not comprehensive.

SECTION 2: ASSESSMENT OF AUTOMATED EVALUATION TOOLS

As far as is known, there is little evidence about the efficacy of the existing automated Web site evaluation tools—guideline review tools in particular. Of the few that have been studied, researchers on the WebTango project (Ivory and Hearst 2002) showed that participants preferred Web pages that were modified to conform to the researchers’ statistical models of highly rated Web designs over the original ones. Participants also rated sites modified based on the researchers’ models higher than the original ones. Differences were significant in both cases. As another example, UsableNet’s LIFT—Nielsen Norman Group Edition (UsableNet 2002) assesses a site’s conformance to guidelines that were developed from studies involving users with and without visual and mobility impairments (Coyne and Nielsen 2001). The premise is that the validated guidelines are inherently effective.

The authors have recently conducted a study of three automated evaluation tools—WatchFire Bobby, UsableNet LIFT, and the W3C HTML Validator (This section presents an expanded discussion of the study in Ivory and Chevalier (2002) Although the first two tools explicitly address accessibility and usability aspects, the latter tool does not. However, the premise is that conforming to standard coding practices may eliminate usability and accessibility issues that invalid HTML code may cause, such as overlapped or nested lists (Paciello and Paciello 2000). The study consisted of two phases: (1) designers used the tools to modify subsections of five Web sites, and (2) users with diverse abilities used a selection of modified sites to complete information-seeking tasks.

Study 2.1: Designers’ Usage of the Bobby, LIFT and Validator Tools

The study was conducted with nine experienced designers, each of whom modified subsections of the following five sites—http://www.section508.gov, http://www.govbenefits.gov, http://spam.abuse.net, http://www.irs.gov and http://www.usatoday.com. Site contents were of general interest to a broad user community, with three of the sites—1, 2 and 4—being government sites. Site designs ranged from a simplistic design with few graphics (Site 1) to a text- and graphics-intensive design (Site 5). Designers’ modifications were intended to improve each site’s universal accessibility (i.e., both the usability or “the extent to which a product can be used by specified users, to achieve specified goals,
with effectiveness, efficiency and satisfaction, in a specified context of use” (International Organisation for Standardisation 1998) and the accessibility or “usability of a product, service, environment or facility by people with the widest range of capabilities” (International Organisation for Standardisation 2002). Designers completed modifications with or without the assistance of one or more automated evaluation tools (WatchFire Bobby, UsableNet LIFT, or W3C HTML Validator). The objective was to examine designers’ usage of these frequently used tools and to assess whether using the tools was beneficial to them.

**Study design:** The authors asked nine designers, who had at least two years of design experience, to participate in a study, wherein each designer modified subsections of five Web sites. They modified each site in one of five evaluation conditions. Designers completed the first site modification without the assistance of an automated evaluation tool (manual). Designers completed the second-through-fourth site modifications using one automated evaluation tool (Bobby, Validator or LIFT). Designers completed the fifth and final site modification using all three tools in combination. Both the order of site modifications and the usage of individual tools were counterbalanced.

Designers completed the following tasks for each site and evaluation condition:

1. Explore the site to become familiar with the subsection. Designers were provided with a usage scenario and disabled links from the subsection to other site areas or to external sites.
2. Evaluate the universal accessibility of the site subsection. Designers used a five-point scale (very low to very high) for these ratings, although it was not specified how designers were to make these judgments; in some cases, designers inspected both the rendered pages and the HTML code.
3. Modify the site subsection to improve its universal accessibility. In conditions without an automated evaluation tool, designers enumerated a list of problems and used their own expertise to guide modifications. Otherwise, they could only address problems identified by the tool. Designers had twenty minutes to modify the first four sites and forty minutes for the final site (combo condition). They made modifications using either the Microsoft FrontPage or the Macromedia Dreamweaver authoring environments on PCs running Microsoft XP.
4. Evaluate the universal accessibility of the modified site subsection, according to the same rating scale used in step 2.
5. Evaluate the automated evaluation tool (conditions with one tool).

Designers completed a debriefing questionnaire about their experiences with using the tools. They were asked to think aloud throughout the study session, and videotaped sessions and captured screen activity summarize the site subsections that designers rated and modified.
Tools used in the study: WatchFire Bobby, UsableNet LIFT and the W3C HTML Validator tools represent the state-of-the-art in automated Website evaluation tools (Ivory and Chevalier 2002). These frequently used tools offer different functionality and enable subsequent evaluation of site changes. Designers used the desktop version of WatchFire Bobby (WorldWide version 4.0), which determines if Web pages conform to the W3C or Section 508 guidelines and provides guidance for correcting problems. Example guidelines include the following: provide alternative text for all images; explicitly associate form controls and their labels with the LABEL element. Figure 1 depicts an example Bobby report, and Figure 3 depicts a Web page overlay. The W3C HTML Validator (September 13, 2001 version) is a Web-based tool that determines if Web pages conform to W3C coding standards and provides guidance for correcting violations. Example guidelines include the following: end tag for element “A” which is not open; and element “TABLE” not allowed here.

The UsableNet LIFT tool 1) determines if Web pages conform to accessibility and usability guidelines (e.g., W3C and Section 508); 2) provides guidance for correcting problems; and 3) in some cases, provides support for repairing the HTML code for images, tables, and forms. Unlike with Bobby and the Validator, designers used versions of the LIFT tool embedded within FrontPage or Dreamweaver (versions 1.1 and 0.94, respectively). The developers claim that the Dreamweaver version, LIFT Nielsen Norman Group Edition (LIFT-NNG), contains the same core guidelines as the FrontPage version (LIFT-FP), as well as additional guidelines derived from usability studies involving participants with and without visual and mobility impairments (Coyne and Nielsen 2001). Example guidelines include using ‘skip links’ (so that users can skip links or navigational elements in both versions) and avoiding too many outgoing links (LIFT-NNG). Figure 2 depicts a typical task flow for evaluating and repairing a Web page within LIFT-NNG.

Whether the three tools enabled more comprehensive evaluations or enhanced designers’ throughput was examined based on quantitative measures. These subjective evaluations of tool effectiveness were then summarized.

Evaluation coverage: This analysis of the number of problems identified in conditions with and without automated tools revealed that, as expected, the automated tools identified significantly more potential problems than the designers had identified—especially when considering the total number of errors identified within each site as in Figure 4. However, as shown below, designers made more design changes when they did not use an automated tool. The numbers do not show that the tools, excluding Bobby, mostly report the same errors (once for each occurrence). Figure 4 shows that there is not much difference in the number of Bobby-reported problems and the number of designer-reported problems (manual condition), possibly because Bobby summarizes the number of unique problems along with the number of occurrences.
Figure 5 depicts the three types of problems that the designers and tools identified—accessibility (top graph), usability (middle graph), or other (bottom graph). Designers primarily identified higher-level accessibility and usability problems that were applicable to the entire site—rather than to individual pages, such as “Spell things more clearly...,” “...make clearer what the section is describing and where the links will go,” and “...multicolored links (bad for vision impairment)....” There were only a few problems identified by both designers and one or more tool, including the following: 1) add attributes or labels for form controls (Bobby and LIFT tools), 2) reduce the number of links (LIFT-NNG), 3) increase text size (LIFT-NNG), 4) add or remove alt text (Bobby and LIFT tools), and 5) add skip links (LIFT tools). Designers did not enumerate any problems that the Validator identified (primarily other types of coding errors), but they repaired tags when they modified sites in the manual condition.

**Designer throughput:** Although the automated tools identified more problems than the designers identified, study results suggest that they did not enhance the designers’ throughput (i.e., their ability to resolve identified design problems within a short amount of time). Figure 6 shows that designers made significantly more changes in the manual condition than in the tool conditions ($p < 0.05$). Designers modified a median of three pages on each site in the manual condition—two pages in both LIFT conditions, one page in the Validator condition, and no pages in the Bobby condition; differences were significant ($p < 0.05$). Nonparametric analyses of variances (ANOVAs) did not show significant effects due to prior tool usage or design experience.

Surprisingly, designers modified a total of only three pages in the Bobby condition. Participants’ comments suggest that they did not think that any modifications were needed, based on the reported problems; this is evident from the relatively higher proportion of designers reporting that they completed site modifications and were satisfied with them in the Bobby condition (See Figure 7). It is even more surprising that designers made significantly more modifications in the manual condition than in the LIFT conditions, because the LIFT tool can assist designers in making some repairs—such as changing ALT text or image dimensions. There were no significant differences in the number of problems that tools reported before and after modifications.

Figure 6b shows that the designers made more holistic changes in the manual condition than in the individual tool conditions. Their modifications addressed usability, accessibility and other problems fairly equally. The figure also shows that designers introduced more errors while making modifications in the manual condition than in the individual tool conditions; designers also introduced errors in the combo condition. The tools did not flag these new errors in the combo condition; designers in the manual and combo conditions were
either unaware of the new errors, or they did not have enough time to correct them.

Analysis of the changes made across designers for the same site and the same tool condition revealed that designers made similar types of changes more often in the manual condition than in the tool conditions. There was typically one page within each site for which multiple designers made one or more similar change. Results suggest that the tools may or may not result in deterministic changes: designers could interpret reported problems or solutions quite differently.

Tool effectiveness: Designers rated the universal accessibility of sites both before and after modifications. A nonparametric, paired-samples test revealed no significant differences in these ratings, possibly because designers often did not have enough time to complete modifications (See Figure 7a) or thought that sites needed to be redesigned.

For each tool, designers responded that they strongly agreed to strongly disagreed with nine statements. Here are four examples: 1) assessments were accurate and relevant, 2) the tool was easy to use, 3) the tool was extremely helpful in improving universal accessibility, and 4) the tool helped them to learn about effective design practices. There were significant differences in responses to the latter two statements. Responses suggest that Bobby was the most helpful and that the Validator was the least helpful. Similarly, responses suggest that Bobby and LIFT-NNG helped designers to learn about effective practices. These responses also illustrate the need to validate the design guidelines embedded within automated evaluation tools to ensure that conforming to them will improve user experiences.
Figure 5: Number of Pages with Accessibility (Top), Usability (Middle) and Other (Bottom) Problems
Maximum, mean, and total number of changes made on individual pages were computed. The median of these site-level measures across sites is also shown. Nonparametric ANOVAs revealed that all differences in the number of changes are significant ($p < 0.05$). Nonparametric ANOVAs also revealed that the accessibility and error differences are significant ($p < 0.05$).

On the debriefing questionnaire, half of the designers identified Bobby as the most effective tool, and over half of them identified the Validator as the least effective tool. Surprisingly, half of the designers reported that neither using all three tools nor using one tool was the best usage scenario. Several designers commented about the need to rely on designers’ expertise or usability analysis (e.g., “...there are usually embedded corporate reasons for placement of objects...needs to be taken into consideration.”).

### 2.2 Study 2: Users Usage of Modified Sites

The preceding study demonstrated that although the automated evaluation tools identified more errors than designers identified without using them, designers made more design changes in the condition without tools. To
determine if the changes resulted in measurably different site experiences, modified versions of each of the five sites in Table 2 were selected for usability testing by users with and without disabilities.

**Study design:** This study was designed to determine if there would be any differences in user performance or in subjective ratings for the original and modified versions of the sites, as shown in Table 3. To understand the changes that produce significant differences, 22 users were asked to participate in a ninety-minute or two-hour study session1 during which they were given a scenario for each site and then asked to browse the site to find specific information (see Table 2). Both the site order and the site version (i.e., modification condition) were counterbalanced, and participants completed tasks on only one version of each site.

Participants' ages ranged from 18 to 55, with an equal number of males and females participating. Although participants had very diverse ethnic backgrounds, the majority were native English speakers and intermediate computer and Internet users with a college or post-graduate education, who typically spent ten or more hours on-line a week. Some 13 participants had one or more of the following disabilities: visual (6), hearing (1), moderate mobility (3), learning (1), physical (5) or other (1). Three participants actually had multiple disabilities.

Participants mainly used Netscape Navigator to complete study tasks. Blind participants experienced problems with the JAWS screen reader software and needed to use Internet Explorer instead. Two participants with mobility disabilities used the keyboard exclusively. For participants with visual and physical disabilities, testers read the study tasks aloud and assisted participants with completing the post-task questionnaires. Nearly all participants had never used the study sites.

**Results:** Since there was no control for user abilities in the study, it is not considered in this analysis. Analysis of data for individual sites revealed a few significant differences in results for sites 2, 3, and 5 (as shown in Tables 4, 5 and 6). It is interesting to note that for these significant differences, the modifications that designers made to the sites in the manual condition improved user performance, yet this was not the case for the modifications designers made in the tool conditions.
## Table 2: Five Study Sites and Scenarios

<table>
<thead>
<tr>
<th>Site</th>
<th>Id</th>
<th>#Pages</th>
<th>Subsection</th>
<th>Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 508</td>
<td>1</td>
<td>8</td>
<td>About 508</td>
<td>Making a company’s site EEOC compliant</td>
</tr>
<tr>
<td>GovBenefits</td>
<td>2</td>
<td>8</td>
<td>GovBenefits Program List</td>
<td>Finding funding sources for a clinical geriatrics professor</td>
</tr>
<tr>
<td>Spam Abuse*</td>
<td>3</td>
<td>6</td>
<td>Help for Users</td>
<td>Complaining about rude spam emails</td>
</tr>
<tr>
<td>IRS</td>
<td>4</td>
<td>7</td>
<td>Accessibility</td>
<td>Finding a Braille version of the 2001 Education Credits form</td>
</tr>
<tr>
<td>USA Today</td>
<td>5</td>
<td>6</td>
<td>FAQ</td>
<td>Getting the crossword puzzle to work in AOL</td>
</tr>
</tbody>
</table>

*Three pages on the Spam Abuse site were ASCII text rather than HTML code.

## Table 3: Number of Participants Who Evaluated Site Versions (Number with Disabilities in Parentheses)

<table>
<thead>
<tr>
<th>Site</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Original</td>
</tr>
<tr>
<td>1</td>
<td>6 (4)</td>
</tr>
<tr>
<td>2</td>
<td>4 (2)</td>
</tr>
<tr>
<td>3</td>
<td>5 (4)</td>
</tr>
<tr>
<td>4</td>
<td>4 (2)</td>
</tr>
<tr>
<td>5</td>
<td>4 (2)</td>
</tr>
</tbody>
</table>
### Table 4: Mean Task Completion Times and Standard Deviations (in Parenthesis)

<table>
<thead>
<tr>
<th>Site</th>
<th>Original</th>
<th>Manual</th>
<th>Bobby</th>
<th>Validator</th>
<th>LIFT-NNG</th>
<th>Combo</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>393.5 (494.9)</td>
<td>–</td>
<td>–</td>
<td>401.3 (250.0)</td>
<td>366.0 (350.5)</td>
<td>329.9 (224.3)</td>
</tr>
<tr>
<td>2</td>
<td>157.0 (54.6)</td>
<td>335.0 (120.1)</td>
<td>–</td>
<td>–</td>
<td>416.5 (286.6)</td>
<td>234.0 (78.4)</td>
</tr>
<tr>
<td>3</td>
<td>478.4 (331.1)</td>
<td>242.1 (162.0)</td>
<td>–</td>
<td>229.0 (195.9)</td>
<td>–</td>
<td>466.5 (285.9)</td>
</tr>
<tr>
<td>4</td>
<td>475.9 (473.6)</td>
<td>364.5 (226.8)</td>
<td>482.2 (264.0)</td>
<td>–</td>
<td>–</td>
<td>303.2 (157.8)</td>
</tr>
<tr>
<td>5</td>
<td>257.4 (161.4)</td>
<td><strong>240.7 (125.7)</strong></td>
<td>274.0 (123.6)</td>
<td>223.1 (125.1)</td>
<td><strong>74.8 (71.3)</strong></td>
<td>–</td>
</tr>
</tbody>
</table>

**Bold entries denote significant differences between task completion times in the original or manual conditions and the tool conditions, based on nonparametric ANOVAs (p < 0.05).**

In terms of these three sites:

**Site 2:** Participants spent more time completing tasks on the LIFT-NNG version of site 2 than the original one (mean of 417 seconds versus 157; Table 4). The designer modified all pages in the LIFT-NNG version, but the modifications addressed coding issues (e.g., fixing tags and adding a document type definition) instead of usability and accessibility issues. The designer also introduced an error on the first page, which affected its layout. The tool did not flag this error, and the designer was either unaware of it or did not have enough time to correct it.

**Site 3:** Participants rated the manual version of site 3 higher than the original one (3.3 versus 1.4 on a five-point scale; in Table 5). The designer made extensive visible modifications to four of the six pages in the manual version. Most of the changes addressed usability issues: wrapping long text links in the navigation bar and adding vertical space between them; increasing the font size for links; formatting headings so that they appear more distinctly; adding vertical space between links in a
bulleted list; and removing an acronym scheme to reduce potential cognitive overload. The designer did not introduce errors on any pages.
### Table 5: Mean Site Ratings and Standard Deviations (in Parenthesis)

<table>
<thead>
<tr>
<th>Site</th>
<th>Version</th>
<th>Original</th>
<th>Manual</th>
<th>Bobby</th>
<th>Validator</th>
<th>LIFT-NNG</th>
<th>Combo</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Designers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Users</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>2.4 (1.2)</td>
<td>2.2 (1.0)</td>
<td>–</td>
<td>–</td>
<td>2.8 (0.5)</td>
<td>2.1 (1.4)</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2.4 (0.9)</td>
<td>2.8 (1.5)</td>
<td>2.0 (1.6)</td>
<td>–</td>
<td>–</td>
<td>2.0 (0.7)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>2.1 (0.6)</td>
<td>1.4 (0.9)</td>
<td>3.3 (0.5)</td>
<td>–</td>
<td>2.1 (1.1)</td>
<td>–</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>2.6 (0.9)</td>
<td>2.0 (0.8)</td>
<td>2.1 (0.6)</td>
<td>1.4 (0.6)</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>1.3 (1.1)</td>
<td>1.0 (1.4)</td>
<td>2.2 (1.3)</td>
<td>2.3 (0.6)</td>
<td>2.0 (1.4)</td>
<td>2.2 (0.8)</td>
</tr>
</tbody>
</table>

Ratings provided by the designers for the original sites (i.e., before making modifications), which are very consistent with the users’ ratings. Bold entries denote significant differences between the users’ site ratings in the original or manual conditions and in the tool conditions, based on nonparametric ANOVAs (p < 0.05).

### Table 6: Task Completion Percentages

<table>
<thead>
<tr>
<th>Site</th>
<th>Version</th>
<th>Original</th>
<th>Manual</th>
<th>Bobby</th>
<th>Validator</th>
<th>LIFT-NNG</th>
<th>Combo</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>17%</td>
<td>–</td>
<td>–</td>
<td>20%</td>
<td>29%</td>
<td>50%</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>100%</td>
<td>60%</td>
<td>–</td>
<td>–</td>
<td>40%</td>
<td>100%</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>60%</td>
<td>75%</td>
<td>–</td>
<td>63%</td>
<td>–</td>
<td>40%</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>75%</td>
<td>50%</td>
<td>40%</td>
<td>–</td>
<td>–</td>
<td>40%</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>25%</td>
<td>80%</td>
<td>67%</td>
<td>0%</td>
<td>40%</td>
<td>–</td>
</tr>
</tbody>
</table>

Bold entries denote significant differences between task completion percentages in the original or manual conditions and the tool conditions, based on nonparametric ANOVAs (p < 0.05).
Site 5: No participant was able to complete the task on the Validator version of site 5, but 80% of participants completed tasks on the manual version as shown in Table 6. (Only 25% of participants completed tasks on the original version.) The designer modified only one page in the Validator version: changes consisted of adding alternative text to indicate advertisements. The blind user in this condition did not complete the task. The designer modified three pages in the manual version: changes addressed usability and accessibility issues, such as moving relevant links to a prominent position at the top of the page, adding headings to the table of contents for an FAQ, increasing font size and adding vertical space around links. Unlike the Validator version, the manual version had visible errors on two pages.

Participants spent only a mean of 75 seconds completing tasks on the LIFT-NNG version of site 5 versus 241 seconds on the manual version (Table 4). However, twice as many participants (80%) actually completed the task on the manual version, which explains the extra time. The designer modified only alternative text to indicate spacer images on pages in the LIFT-NNG version. Due to the busy nature of the initial page, participants typically gave up on completing the task. As discussed above, the designer of the manual version made considerable changes to reduce page complexity and consequently, cognitive overload.

In Summary: Within the narrow parameters of this preliminary study, results suggest that the three automated evaluation tools were not very effective in helping designers to improve Web site usability and accessibility. Perhaps a more rigorous study—one using less mature sites; studying novice or expert designers; or giving designers more time to learn the tools, understand the guidelines, and modify sites—might produce a different outcome. Nonetheless, the findings do have implications for both researchers and Web professionals. More research needs to be done to validate the guidelines embedded in automated evaluation tools and to make them more usable. Fundamentally, the results suggest that the guidelines do not examine higher-level issues that negatively limit Web site use, such as page complexity or legibility of text with the color combinations used. This limitation was also suggested by the survey in Section 1 on Supporting Diverse User Abilities.

Although it should be self-evident, the results show that Web professionals cannot rely on the automated evaluation tools alone to improve sites. The studies suggest that the guidelines embedded in the tools, or perhaps the way designers interpret them, may not necessarily improve the usability and accessibility of Web sites—at least not more so than when experienced designers rely on their own expertise. However, the results do suggest that the tools play an important role in educating Web professionals about effective design practices.
2.3 Improving Automated Evaluation Tools

Both the survey and the study of automated evaluation tools suggest the need for more coverage of diverse user abilities than tools currently support (Shneiderman and Hockheiser 2001). For example, users who cannot use a keyboard or who have hearing impairments are typically not addressed by today’s tools. However, it may not be possible to address some user needs at design time (e.g., accommodating users who cannot use a keyboard or mouse and who have a hearing impairment). There appears to be a need to merge advances in the areas of automated evaluation and transformation tools to enable Web use by a broader range of users. Ideally, the evaluation tools could encode some knowledge about automated transformation tools (including those other than screen readers), such that they could run pages through such tools and then evaluate transformed pages. This integration has occurred to some degree with tools, such as WAVE and A-Prompt, which depict linearized text or pages for the developer to analyze. The tools could take this one step further and actually assess the linearized pages or tables. Integrating automated transformation and evaluation tools would enable designers to determine how robust their sites are to such transformations, for example.

The study of evaluation tools also suggests that more research needs to be done to examine higher-level issues, such as page complexity or text legibility with the color combinations used. The tools should not apply guidelines independently; for example, increasing font size may require a subsequent reduction in the amount of text on a page, but this problem is not addressed by the tools. The WebTango approach (Ivory and Hearst 2002), described briefly in Section 1 above, provides an example of how to address these issues.

More research needs to be done to make the guidelines and the tools themselves more usable (Shneiderman and Hockheiser 2001). Guidelines are often too long or too vague for designers without a human-factors background to understand. Thus, designers experienced difficulties interpreting them, which in turn interfered with their ability to implement appropriate changes. Chevalier and Ivory’s (2002) studies on the effects of guideline training sessions provide some guidance on helping designers to apply guidelines. More studies need be conducted to examine the cognitive difficulties Web designers encounter in using these tools.

More work also needs to be done to improve the presentation of guideline violations, because designers find it overwhelming to wade through long lists of violations, especially when they consist of the same types of violations. Bobby provides a good model for how to minimize overload when presenting this information. Lastly, guidelines need to be context-sensitive and perhaps even extensible, so that designers can adjust them based on the context in which sites are designed.
SECTION 3: ASSESSMENT OF AUTOMATED TRANSFORMATION TOOLS

Few evaluations of automated transformation tools have been conducted. However, those tools that have been evaluated provide important insights. Tools that have been evaluated include one in the domain of motor impairments, one that tests the efficacy of a speech alternative to keyboard and mouse input, and two that support users with no vision.

In the domain of motor impairments, Christian et al. (2000) compared a voice-controlled and a mouse-controlled Web browser, 1) to determine the time cost of using voice (due to errors and slow recognition speeds) and 2) to experiment with different ways of representing links to reduce recognition errors. The results showed that voice-controlled browsing was 50% slower than traditional mouse-based browsing (Christian et al. 2000). Additionally, users preferred to refer to links by their textual name rather than by their number.

In the domain of visual impairments, Fuller (1999) conducted an informal evaluation of a tool designed to support navigation within a table. The author demonstrated the tool to conference attendees and followed the demonstration with verbal, qualitative feedback. User comments indicated the need for more personalization, additional navigation pathways (such as repeat and skip capabilities), and better position/location information.

Another study in the visual domain compared the effectiveness of non-speech audio to speaker changes in spoken audio for indicating information about document structure. James (1998) found that novices preferred more explicit markings than either speaker changes or non-speech audio could provide; they requested explicit structural descriptions. Trained users found relative changes, such as differences in pitch, were difficult to differentiate. More natural, meaningful sounds were preferred. James also found that speaker changes could effectively indicate structure.

The majority of other evaluations also fall within the vision domain. Hearing and cognition are notably absent from the evaluation literature, probably because automated tools for Web access that focus directly and intentionally on these domains do not appear to exist. The evaluations described above focus on specific tools, users and constrained sets of Web pages. A next step would be to compare different general tools that address broad sets of users and problems, in order to determine whether they cover the full set of needs of their users or can adapt appropriately to differing needs.

3.1 Improving Automated Transformation Tools: Supporting Users with Motor Impairments

As highlighted in at the start of this section, there are many gaps in the set of issues supported by these tools, especially for users who have motor impairments that limit their ability to use a mouse and keyboard. Input capabilities can be characterized in terms of the amount of “bandwidth” required of users (where bandwidth is defined by the speed at which users can produce
input) and the range of inputs available to them (Mankoff et al. 2002). For example, a mouse has much higher bandwidth than a single, eyebrow-controlled switch. Thus, in this document, low-bandwidth input refers to almost non-existent mouse and keyboard use, whereas medium-bandwidth input refers to some mouse and keyboard use. Low-bandwidth users may type below 10 words per minute (usually using an on-screen keyboard) and have no direct mouse control, while medium-bandwidth users may type about twice that fast, and they can use a mouse. Both sets of users may have difficulty with errors in movement or selection.

A summary of work in Mankoff et al. (2002) on Web accessibility for low bandwidth input follows, highlighting some requirements for low-bandwidth input and describing the server-side tool built to meet those needs. The findings about the needs of users with medium-bandwidth input are then discussed, based on a recent series of interviews.

Low-bandwidth input: Efforts in this domain were focused on the needs of people who use one or two switches (i.e., devices that transmit a fixed set of signals) to control a computer. From a technical perspective, browsing the Web is simply not possible without either 1) an on-screen keyboard interface, which allows the user to laboriously select keys that are then sent to the Web browser, or 2) some changes to Web pages/browsers themselves (Colven and Lysley 2000). Seven issues from Mankoff et al. (2002) that must be addressed to make single-switch browsing feasible are listed here as requirements.

1. The currently selected link is visible when tabbing through links.
2. The user can read and navigate text even when it contains no link.
3. The user can traverse the history list forward and backward.
4. The user can access bookmarks and add to them.
5. The user can go quickly to a point of interest with a minimal number of input signals.
6. The user is given alternatives for entering text and dealing with form elements.
7. The user is given enough information about link targets to make informed decisions about whether to follow them.

Currently, six of these have been implemented using a proxy and four of them in a client-side tool. Two images showing how the proxy addresses six guidelines are shown in Figure 8. The numbers visible on the figure are each located near a modification that addresses the guideline referred to by the number:

- [1] shows a link highlighted.
- [2] shows “Page Down” and “Page Up” buttons to navigate when there are no links.
• [3] “Back” can be used to traverse history.
• [4] “Bookmark this page” is one of the functions supported regarding bookmarks.
• [5] “Page Down” and “Skip to Content” jump quickly to a point of interest.
• [6] Requirement 6 is not currently supported by this proxy.
• [7] The quote in parentheses foreshadows the start of the page being linked to.

The browser shown in Figure 9 is our client side tool. It meets the following guidelines:
• Guideline 1: the currently selected link is visible (see the first link under “Welcome to the College”)
• Guideline 3: the “Next” and “Prev” buttons provide access to the history list
• Guideline 4: the “Favorite” button provides access to the bookmarks.
• Guideline 7: the image and bottom right shows a preview of the page associated with the currently selected link

Note that although an automated tool could in theory support all seven requirements, only three of the requirements (numbers 1, 2, and 3) can realistically be dealt with at page authoring time. Support for bookmarks is generally browser specific, and certainly user specific, and thus not appropriate to handle at this time. Moving to a point of interest is user specific, and difficult to support. Alternatives for entering text are likely to be cumbersome for users without motor impairments, and thus still more appropriately dealt with dynamically. Finally, information about link targets remains difficult to specify since the contents of those targets may change with time and may not be in the control of the author linking to that target.

Medium-bandwidth input: Users with some keyboard and mouse use still encounter problems when browsing the Web. Five interviews of people with advanced Parkinson’s disease and other mild to severe motor impairments were recently conducted. All respondents had difficulty with both the mouse and the keyboard, but they preferred and could use the mouse. Many respondents perceived the mouse to be non-problematic, despite obvious difficulties observed by the interviewers. For example, several had trouble releasing the mouse button without dragging. Keyboard trouble included hitting multiple keys and related problems.
Figure 8: Two Web Pages Modified by a Proxy to be Accessible with Two Switches
(The Numbers Refer to Guidelines that are Addressed by the Proxy)
These specific low-level motor issues have been described before, in depth by Trewin and Pain (1997) and Keates et al. (2002). However, the contribution here is to consider their impact on Web browsing. In the context of Web browsing, these low-level problems translated into difficulty scrolling and slow typing. Because all respondents reported using the Web for tasks that require both keyboard and mouse input (such as on-line email and shopping), these difficulties have implications for Web page accessibility that could be addressed through automated tools.

In particular, past work has shown the utility of an expanded mouse cursor for selecting small targets such as URLs (Worden et al. 1997). This is clearly a client-side technology solution. Other techniques, such as manipulating the presentation of content to reduce the need for scrolling, could be addressed with client- or server-side techniques.

One issue that does not require additional technology was also identified. Respondents often did not know what tools were available to them. For example, one common problem was the negative impact of pop-up windows. Respondents found these particularly laborious to remove simply because they often did not know about the keyboard shortcut that closes a window. This highlights the need for further education about existing computer technologies that can help to mitigate access problems.

In summary: Existing tools do not fully solve the problems faced by users with motor impairments. This finding is not surprising, considering that the most well-known guidelines cover only some of the issues. For example, the W3C Guideline 9 recommendation for device independence in the Web Accessibility Initiative (1999) states that: “Generally, pages that allow keyboard interaction are also accessible through speech input or a command line.
interface.” This contains implicit assumptions about the capabilities of users, and for example does not directly address the needs of users who are limited to single-switch input. Our new guidelines address these issues. However, they have not yet been validated directly through studies with low-bandwidth users.

Despite the lack of adequate tools, many of the problems faced by users could be adequately addressed by existing technology, with the help of some education about what options are available. Given the medium-bandwidth respondent’s expressed preferences for their existing technology, and the wide differences between the needs of people with motor impairments, it is important to note that any complete solution must be compatible with current existing options, such as tabbing and shortcut keys.

4 CONCLUSIONS AND FUTURE WORK

Automated tools have the potential to support designers and to make Web pages accessible to a much broader swath of end users. In particular, automated evaluation tools may help designers to identify specific problems related to accessibility and usability, while automated transformation tools can dynamically modify Web content to meet the accessibility needs of end users. Despite the obvious potential benefits of these tools, the above survey of existing tools shows that there are large categories of users whose needs are not yet adequately addressed. Even in the areas where the most work exists, such as accessibility issues facing blind users, there are still many open questions. Groups such as people with low vision or people with motor impairments face an even larger gap in terms of available technological or evaluation support.

In addition to analyzing the coverage of existing tools, the question “Do they work?” was asked. The study of automated evaluation tools described in this article showed that improvement is still needed in this domain. In general, developers, who relied on their own expertise to modify sites, produced sites that yielded some performance improvements. This was not the case for the sites they produced with evaluation tools.

In the realm of automated transformation tools, most work has focused on vision-related tools and tools that remove distracting animations and advertisements. The above research shows that some classes of users are not supported, among them users with motor impairments or difficulty with mouse or keyboard usage.

An important future goal is support for combining automated evaluation and transformation tools. This could enable designers to determine how robust their sites are not only as originally designed, but also when transformed to meet the needs of specific user sub-populations. Additionally, the authors are working towards developing additional discount and early-phase evaluation methodologies that can provide better support for designers than the best tool found in this work-designer intuition.
REFERENCES


Center for Information Technology Accommodation. 2002. Section 508: The Road to Accessibility. Available at http://www.section508.gov/index.cfm


October


Available at [http://validator.w3.org/](http://validator.w3.org/)

Available at [http://jigsaw.w3.org/css-validator/](http://jigsaw.w3.org/css-validator/)

Available at [http://www.w3.org/WAI/EO/Drafts/PWD-Use-Web/](http://www.w3.org/WAI/EO/Drafts/PWD-Use-Web/)

Available at [http://www.w3.org/WAI/ER/existingtools.html](http://www.w3.org/WAI/ER/existingtools.html)


---

1 Automated evaluation and transformation tools are discussed as they existed at the time of this article (September 30, 2002). We realize that these tools are constantly evolving and possibly incorporating changes that might affect our findings. We apologize in advance for any discrepancies due to more recent tool versions.

2 Section 508 is a U.S. federal ruling. The Web site accessibility requirement appears to have been recently reversed.

3 Designers enumerated problems that typically applied to the entire site in the manual condition, which is not reflected in the graphs. No designer modified site 1 in the manual condition. The Bobby priority 1, 2, and 3 errors have been aggregated. The Validator failed on sites 2, 3, and 5.

4 The two-hour sessions were to allow extra time for the 13 participants with disabilities to complete the study.