A World Wide Web Mediator for Users with Low Vision

Silas S Brown & Peter Robinson
University of Cambridge Computer Laboratory
New Museums Site Pembroke Street
Cambridge CB2 3QG England
+44 1223 334 446/637
{silas.brown,peter.robinson}@cl.cam.ac.uk

ABSTRACT
World Wide Web mediators, which intercept and modify Web traffic, are becoming increasingly popular as a means of assisting Web users with special needs. This paper describes the first author's Access Gateway mediator which is primarily intended for users with low vision.

KEYWORDS
World Wide Web, visual handicap, mediator.

INTRODUCTION
The ability to customise the size and colour of text on a computer screen can assist users with low vision, as well as some dyslexic users [9], to read independently. Although most Web browsers allow such customisation, it is not always honoured, because browsers have bugs, and because content is often delivered in a graphical, rather than textual, form. Moreover, the page layouts that are used in many Web sites become difficult to use when the print size is significantly increased; they can also be difficult for blind people using speech synthesisers or Braille displays.

Special browsers and access software can assist [3]. However, users are often restricted in their choice of software by institutional policy, financial circumstances, and compatibility requirements. There can also be problems with changing the software and settings on a computer that belongs to another person or is publicly available. It is therefore beneficial to seek a means of assistance that is independent of the client software.

There are guidelines for page authors on making their pages accessible [5]. However, it is difficult to make these guidelines effective for all client software, especially given the non-standard behaviour of some browsers. Moreover, many authors are unaware of the guidelines or view them as irrelevant, overly restrictive, or too time-consuming to implement. In some countries, certain types of website are required by law to be accessible, but it is not feasible to enforce this for large numbers of sites.

Another means of assistance involves modifying the page data at some point between the server and the client, by means of a *mediator* [13] – a server that retrieves pages from other servers on behalf of a client, like a proxy, but that performs some useful transformations on the pages before returning them to the client. This has the advantage of being independent of both client software and server-side techniques.

RELATED WORK
The World Wide Web Consortium maintains a list of evaluation and repair tools [4] that includes several mediators, and there are others. Some of the mediators that have similar objectives to Access Gateway are briefly described here.

The first mediator was probably Shodouka [14], a service for displaying Japanese web pages on browsers that do not have Japanese fonts, by using graphics to show the characters. Entry to Shodouka is by means of an HTML form, and when a page is retrieved, all the links on it are modified to point back through the mediator, so links can be followed without leaving the mediator. Many later mediators also used this concept. Shodouka is designed to give a rapid response; it does not have to wait for the complete page to be retrieved from the server before it can begin processing it and delivering it to the client. Shodouka's author Ka-Ping Yee has since used its code in three other mediators.

Four months after the initial release of Shodouka, Kennel, Perrochon and Darvishi released WAB [10], an HTTP proxy (based on CERN httpd) that modifies HTML to assist users who are totally blind. It was not customisable, and its being a proxy meant that it could not be used by individuals who were only allowed to use their institution's proxy, unless they could have a local copy installed. It was WAB that gave us the idea of writing Access Gateway.

BETSIE [11] is a simple Perl script written by Wayne Myers for the British Broadcasting Corporation (BBC) in conjunction with the UK's Royal National Institute for the Blind (RNIB). It was originally intended to make the BBC's website more accessible, but is now available for download. The intention is that a webmaster of an inaccessible site can install BETSIE on that site, possibly adding code that is specific to the site (as is done on the BBC's site). It is therefore assumed that there is some co-operation between the mediator and the webmaster, and thus the mediator does not have to handle every possible HTML technique. The BBC's website is large and rapidly

changing, with many webmasters and a large amount of dynamically generated HTML, so it is difficult to check that BETSIE always works.

The concept of using knowledge of specific websites to enhance the processing of those sites is also used in Asakawa and Takagi's proxy for blind users [2, 12], which can use volunteer-supplied knowledge of a site's house style to determine the order in which the sections of each page should be presented. Although this knowledge does have to be acquired and is liable to become obsolete by site redesign, the proxy can in some cases acquire it automatically by using heuristics on a single page, or by comparing different pages of the site or different versions of a page. This latter technique is similar to that used by Ebina, Igi and Miyake [7] in their extractor of updated content. Since site-specific knowledge takes time to acquire, it is more appropriate if the mediator has a large user base and the site is popular. For this reason, Access Gateway does not use such techniques.

Asakawa and Takagi's proxy can be customised; each user's customisation is stored on the server, rather than being encoded in the page's links. Proxy authentication is used to identify each user. Where users do not have suitable client software or cannot change their proxy settings, a second mediator can be used, which encodes the user's identification in the page's links and mediates between the client and the proxy.

A popular mediator is Muffin [8], which is intended for use as a personal proxy. Muffin can only be customised by the proxy's administrator, but the intention is that each user administrates his or her own copy. This is a client-side approach and is therefore not always possible for the reasons given in the introduction.

A little-known mediator for users with low vision is AlterPage [1], intended for users of the 'WebTV' set-top box service in the USA. AlterPage has limited capabilities, but some WebTV users find its use of WebTV-specific markup to be helpful (Petro Giannakopoulos, private communication).

OBJECTIVES

The objectives of Access Gateway were based on the first author's personal experience as an individual with low vision. It was intended to be practical as a tool for allowing people with low vision to access *any* Web page that is usable by fully-sighted people, without needing the co-operation of webmasters or volunteers. It should therefore be able to handle frames, tables, images and maps with missing or meaningless ALT attributes, lists of adjacent links, forms, authentication, SSL, Java, JavaScript, cookies, Flash, plug-ins, Cascading Style Sheets, HTML errors, referer tracking, browser checks, automatic refresh, cluttered layouts, ASCII art, double-spaced characters, formatting directives that depend on the display dimensions, and any of the world's commonly-used character sets. In addition, it should be usable in any Web browser on any operating system, without having to install additional software or change any of the browser's settings, and it should be usable over low bandwidth.

The program should be customisable by individual users, and able to support many sessions simultaneously, each with its own settings. Customisation by individual users is important because special needs are diverse; every feature should be optional. The customisation interface should itself honour the user's preferences, responding to changes immediately if possible, and should be easily usable. The user's customisation should be preserved between pages, so that users can visit other pages (such as by following links) without having to re-enter their customisation. However, the mediator should be stateless, so that the settings are stored by the client rather than the server, and so are not subject to space restrictions or timeouts. Users should be able to pass on their settings to others, publish them as presets, use them on different installations of the program, and store them persistently in bookmarks.

Attempting to make pages accessible by removing unwanted HTML markup is not sufficient. For example, removing table markup can leave unterminated links and unnecessary "spacer" objects. Access Gateway should prevent these unwanted side-effects. In addition, it should be able to compensate for the information loss that might occur when markup is removed. For example, if a page author uses colour or layout to convey information, then that information might be lost when the display parameters are changed to the user's preferences. The mediator should be able to convey some of this information in another way, but without unnecessarily cluttering the output. For example, the mediator could use multiple colours but select them from a user-supplied list, or insert various punctuation characters into the text.

It is often necessary to display URLs to the user, either in the status line of certain browsers as the pointer passes over a link, or because no other textual information about the link is available. The gateway should be able to present these URLs in an abbreviated form so that the reader can more easily extract meaningful information.

The implementation should be portable across different server platforms, and secured against remote exploits and denial of service attacks, but without unnecessarily compromising functionality. It should be hosted on servers that do not insert advertisements or other clutter. Some establishments may require access to the program to be restricted to internal clients, and some may wish to disable certain features, perhaps to reduce traffic. It should also be possible for different installations of the program to be aware of each other, so that they can offer to re-direct users to a closer mirror if it is sufficiently capable.

The implementation should be extensible with new features as the need arises.

IMPLEMENTATION

Access Gateway has been implemented as a CGI script in C++. The current implementation fulfills *most* of the

objectives. At the time of writing, it can be used at the following URL: `http://www.csse.monash.edu.au/cgi-bin/cgiwrap/~ssb/access`

The program is easily demonstrated by some examples. Consider the extreme case of a user who uses a text-only browser such as Lynx. Here is the home page of an organization "dedicated to providing innovative consumer products that make everyday living easier":

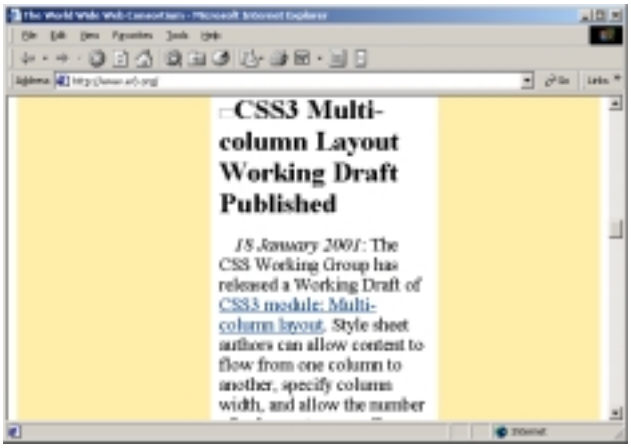
```
                                    OXO International
                [INLINE]
                [INLINE]
                [INLINE]
```

Even using the *list* command to reveal links only gives:

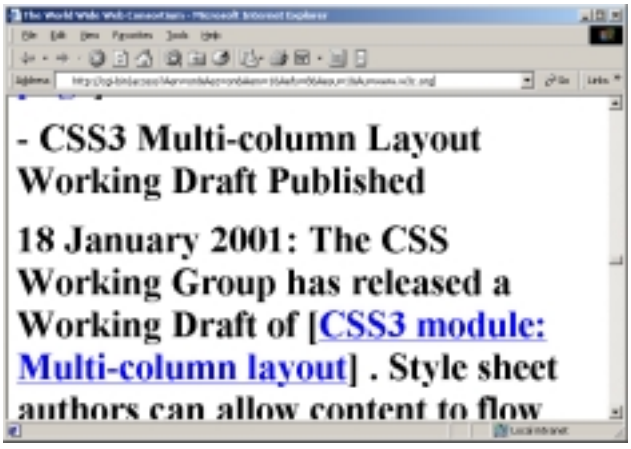
```
Hidden links:
1. javascript:redirect()
2. javascript:redirect()
3. javascript:redirect()
```

However, Access Gateway successfully extracts the links from the JavaScript and allows them to be followed.

More general problems arise in graphical browsers. Even organizations with the very best intentions may make tacit assumptions about the font sizes used to view their pages and waste space with unnecessary layout:



Access Gateway utilizes the full width of the window, as well as allowing larger text than any of the browser's normal settings:



EVALUATION

The URL of the page to be processed, and any preferences, are submitted to the mediator through an HTML form, and the response is modified so that all links and forms point back through the mediator and specify the same settings. An increasing number of sites are using client-side scripts and plug-ins for their navigation, and the automatic modification of these is non-trivial; many mediators make no attempt to process them. Access Gateway can extract URLs from some Javascript, but it fails with more complex scripts and with applets that use `java.applet.getDocumentBase()`. It uses Javascript itself for simplifying the status line messages and for making some customisation changes immediately visible, but Javascript support is not a requirement.

Since cookies are part of the user's state, Access Gateway treats them in the same way as user preferences, so sites that require cookies work even if the browser does not support them. However, the need to encode the state (including the cookies) in link URLs limits the total size of stored cookies. Further, encoding data in link URLs significantly increases the size of a page containing many links, which can cause problems when the client has low bandwidth. Access Gateway compresses its URLs by abbreviating certain combinations of options, although further compression is conceivable. It can also make use of client-side cookies if the client supports them, but client-side cookie support is not required.

It is difficult to mediate secure sites without compromising security. An encrypted connection can be made between the mediator and the remote site, and possibly between the client and the mediator, but the mediator itself must be trusted. Access Gateway contains no cryptographic code, in order to remain legal if strong cryptography is banned. However, it can run on a secure server and it can use the SSL version of Lynx (if available) to fetch pages. Basic authentication, being insecure, is trivial to mediate.

The customisation options in Access Gateway are mostly for enabling or disabling selected features of the mediator, and sometimes for setting parameters. Most features are enabled by default. In order to make the options more navigable, they are arranged in a hierarchy, and online help is available. Text is currently in English but work is in progress to add translations.

Access Gateway uses Unicode internally and can read a number of character encodings; it can detect which one to use based on frequency tables, so long as the user specifies a language (e.g. Chinese) and the page does not use multiple character encodings. Indeed, the most popular use of Access Gateway is as a character set converter that uses images for characters that are not supported by the client, so long as they are available on the server (not all of Unicode is supported). This works best in ideographic languages, the most popular being Japanese – the gateway has mirrors under guises like "Japanese Viewer" and "ACCESS-J", while other mirrors have this feature disabled

to reduce traffic. Shodouka (discussed earlier) usually outperforms Access Gateway when rendering Japanese.

Dealing with images that do not have alternative text (ALT attributes), or that have unhelpful ALT attributes like "image", requires heuristics; it is especially important to provide text for navigational links. Some mediators (e.g. Asakawa and Takagi's proxy) substitute the title of the page that is being linked to; Access Gateway tries to extract text from the URL, since this avoids the need for fetching further pages during processing. A problem is that both URLs and titles can be meaningless. Dardailler [6] proposes a repository of ALT attributes for frequently-used websites; however, access to other sites can still be important.

Layout changes are non-trivial because the page's reading order is difficult to determine automatically. Access Gateway tries to output meaningful content first by using a simple heuristic based on link density; it often fails. Further work is needed in this area.

One consequence of features like layout changes is that the mediator must retrieve the entire document before it can send any of it to the client. This makes large pages seem slow, since the client cannot begin to display the page until the mediator has finished processing it; pipelining mediators, such as Shodouka, appear faster. Pages with images have the additional problem that they often cannot be displayed until the dimensions of all images are known; although the gateway specifies the dimensions of any character images it adds, this is not always true of images that are already on the page, so ideographic pages sometimes take a very long time to display. Once the ideographic characters are in the client's cache, this problem is reduced.

CONCLUSION

Mediators can do much to alleviate the problems with Web pages that are experienced by users with special needs, but it is very difficult for mediators to eliminate these problems completely.

Mediators are coming into mainstream use for Web-enabled mobile telephones, where the client's processing and bandwidth is limited; both WAP and iMode make use of mediators. The small size of mobile displays produces layout problems similar to those associated with large print on a desktop screen, so the mobile market might generate more interest in researching this area.

Ideally, a mediator should allow its users some means of specifying their own transformations, rather than relying on those already implemented, but this is difficult if the users are not programmers.

ACKNOWLEDGMENTS

The first author is supported by a studentship from the UK Engineering and Physical Sciences Research Council.

REFERENCES

1. Anonymous ("Agent 33"). *AlterPage*. `http://www2.crecon.com/agent33/alter.page`.
2. Chieko Asakawa and Hironobu Takagi. Annotation-based transcoding for nonvisual web access. In *Proceedings of the Fourth International ACM Conference on Assistive Technologies ASSETS 2000*, pages 172-179, Nov 2000.
3. World Wide Web Consortium. *Alternative Web Browsing*. `http://www.w3.org/WAI/References/Browsing`.
4. World Wide Web Consortium. *Evaluation, Repair, and Transformation Tools for Web Content Accessibility*. `http://www.w3.org/WAI/ER/existingtools.html`.
5. World Wide Web Consortium. *Techniques for Web Content Accessibility Guidelines*. `http://www.w3.org/TR/WCAG10-TECHS/`.
6. Daniel Dardailler. *The ALT-server: An accessibility collaboration project proposal*. `http://www.w3.org/WAI/altserv.htm`.
7. Tsuyoshi Ebina, Seiji Igi, and Teruhisa Miyake. Fast web by using updated content extraction and a bookmark facility. In *Proceedings of the Fourth International ACM Conference on Assistive Technologies ASSETS 2000*, pages 64-71, Nov 2000.
8. Mark R. Boyns et al. *Muffin World Wide Web Filtering System*. `http://muffin.doit.org`.
9. Peter Gregor and Alan F. Newell. An emperical investigation of ways in which some of the problems encountered by some dyslexics may be alleviated using computer techniques. In *Proceedings of the Fourth International ACM Conference on Assistive Technologies ASSETS 2000*, pages 85-91, Nov 2000.
10. A. Kennel, L. Perrochon, and A. Darvishi. Wab: World-wide web access for blind and visually impaired computer users. In *New Technologies in the Education of the Visually Handicapped, Paris, and ACM SIGCAPH Bulletin*, June 1996. `http://www.inf.ethz.ch/department/IS/ea/blinds/`.
11. Wayne Myers. *BETSIE (BBC Education Text to Speech Internet Enhancer)*. `http://www.bbc.co.uk/education/betsie/`.
12. Hironobu Takagi and Chieko Asakawa. Transcoding proxy for nonvisual web access. In *Proceedings of the Fourth International ACM Conference on Assistive Technologies ASSETS 2000*, pages 164-171, Nov 2000.
13. Ka-Ping Yee. *Definition of a Mediator*. `http://www.lfw.org/ping/mediator.html`.
14. Ka-Ping Yee. *Shodouka*. `http://www.lfw.org/shodouka/`.