A Different Approach to Real Web Accessibility

António Ramires Fernandes, Fernando Mário Martins, Hugo Paredes, Jorge Pereira

Universidade do Minho, Portugal
Email: sim@di.uminho.pt

Abstract

Accessibility is sometimes taken for granted, however for people with certain disabilities there are still a number of obstacles between them and the Information Society. At a time where the Information Society is becoming more and more important it is imperative to make sure that no one is left out. The web is becoming the centerpiece of this new information age; therefore if the sentence "Information Society for All" is to have any meaning then the web must be accessible to everyone. Nevertheless the web is turning into a showcase for designers and technology experts. Although this is pushing many people to the web it does drive some sectors away. People with specific disabilities, namely the blind community, find it harder and harder to read common web pages. This paper reports on a work on a talking browser showing some features that either are not yet widely implemented or, in some cases, have never been reported. Furthermore a new approach is presented, based on information extraction scripts for template urls, to provide real accessibility for blind users. This latter approach, although restricted to a small subset of the web by its nature, aims at giving blind users further views of a web page which will provide them with a clean extraction of the main information on a web page.

1. Accessibility Today

When browsing the web today, we're amazed at the graphical quality and the richness of content with Flash animations, Java Applets, and other fancy multimedia plug-ins. It is very common to see HTML being used as a design language, instead of a content language. HTML tables are frequently used for layout instead of data. The design industry is clearly enjoying this new media. That's OK, we've got enough bandwidth for this, and in the future bandwidth will keep increasing so there are no technical problems in doing this. As a matter of fact we believe it's great. This approach drives people to the web making it grow. Unfortunately it also drives people away from the web.

Blind people are at a severe risk of being left out of the information society since the extraction of information from web pages can be extremely hard. Does this mean that we shouldn't use rich data types, or even simple images in a web page? We don't think that trying to discourage users and developers to use these data types is the way to go. In our understanding this would only create a wider gap between the blind community and the Information Society.

Recently, in 1999, the W3C released a first version of the accessibility guidelines [1]. This document defines a set of guidelines for web page construction. The guidelines are built around the idea that web designers can still use all that technology provides, including Flash animations, Java, 3D, and other rich media types. The main purpose of this document is to provide the web page designer with suitable, and of course accessible, ways of making that content available to users with disabilities. The guidelines are a step in the right direction towards having an information society for all. Nevertheless its implementation requires goodwill, and maybe even funding, from the content publishers.

The guidelines by W3C specify how the pages should be built in order to be accessible; the next step is to get a tool to actually read the page. Two different approaches are available today: screen readers and talking browsers. In here we will focus mainly on the talking browsers. This latter approach has seen some good examples as, for instance, the IBM Home Page Reader [2], Brookes Talk [3], and pwWebSpeak [4]. The talking browsers have a demanding task on their hands: to convert into meaningful text a HTML document regardless of how compliant it is with the W3C guidelines.

The screen reader is only concerned with the look of the document. If the document's graphical layout is clear then a good screen reader will be able to read the page. However the screen reader has no access to the document's structure, and this is where the talking browsers do have some advantage. If the W3C guidelines are followed then the talking browser ought to do a better job than the screen reader. Furthermore the talking browser can take advantage of the knowledge provided by the documents structure to present the user with different views of the document. There is one disadvantage though, if the structure of the document does not reflect its visual appearance then the talking browser can be misled, and provide an interpretation of the document which was not intended.

The issue is no longer to find a technology to read a web page. The real issue is reading it in a reasonable amount of time, allowing the blind user to grasp the information at a speed comparable to that of a sighted user.

There are still problems which have to be solved for talking browsers, namely how to deal with plugins, Java and JavaScript. In Earl et. al. [5], a review of two talking browsers, IBM HPR and pwWebSpeak, both manufacturers acknowledge this point (HPR release 3.0 claims to reads pages with JavaScript). Perhaps the ideal tool will be a merge between a talking browser and a screen reader.

In section 2 we'll describe our talking browser, the AB browser, focusing mainly on the features that we believe are either not yet widely implemented, or are new to accessibility tools. Section 3 explores a new approach for providing real accessibility for specific sites.

2. The Audio Browser: Standard Issues

The Audio Browser, AB hereafter, is a talking browser developed at the University of Minho under the support of the research program CITE IV. Although there are already a few good talking browsers, there was none at the time that spoke standard Portuguese (IBM is planning on launching a Brazilian Portuguese version in March 2001). Furthermore we believed that we could explore some directions yet unseen in other similar products. While we don't intend to establish a direct overall comparison to other similar products, we do think that launching new features is required to keep the ball rolling. In this paper we give a more comprehensive treatment to the features that are not common in other talking browsers, as opposed to common or basic features.

One of the main disadvantages a blind user has when accessing a web page, is the inability to quickly find and/or move to a position in a document. The ability to see gives the user the power to locate the desired information in a quick glance. For instance, the second table, the e-mail of the Webmaster, or the list of anchors in a long document with a table of contents. Providing, as much as possible, a fraction of that power to disabled users quickly became our main goal, giving the user the required content as fast as possible. Although we're clearly a long way from reaching our goal we believe that we have implemented some uncommon, yet useful, features for talking browsers.

In AB the web page is presented in three different ways: audio for blind users; visually as in a standard browser; and in a large hi-contrast text box for those with some usable vision. In this way, our browser allows for different users to share the web experience together. This feature is desirable for full inclusion of the blind community in the Information Society and as far as we know only the latest version of HPR shares this feature.

In AB four views of the web page are provided: the document itself, the forms of the document, the links, and the tables. Each view will be explained in detail in the next subsections.

The browser also has common features such as history of links visited and bookmarks. When book marking a document, we give the user the possibility to store not only the page's address, but also the current position in the document. This is a useful feature in long pages since the user can stop at the middle of a page and later, in a future visit to the page, restart from the same position.

2.1. The Links View

When the user swaps to the links view, it is presented with the full set of links in the document divided in six categories:

- Anchors to the current page
- Internal links, i.e. links within the same domain
- External links, i.e. links to other domains
- Internal files, i.e. files downloadable from the current domain
- External files, i.e. files downloadable from other domains
- E-mail list

We believe that this classification narrows down the search to locate a link. The categories are presented initially and the user can then navigate between them with up and down arrow keys, pressing [return] opens up a category. Navigation within a category is done in a similar fashion with the up and down arrow keys. When the user is upon a link it has the option of having the document's text read, immediately before and after the link, to provide a context to the user. After the text is read the user can stay on the document or return to the previous location in the links list. This is mainly useful when web pages have very small or even meaningless link text, as in for example "…please press here", where the link is the word "here".

The list of anchors in the current page is extremely useful because it can provide an easy navigation in those long documents that have a table of contents. In this case the list of anchors is almost the table of contents.

Separating between internal and external links helps the user to navigate in the results page of a search engine like Google. In this case the internal links provide the user with the search engine options whereas the external links are a quick shortcut to the results. The advantages of the e-mail list become obvious when, for instance, the user is looking for the technical support e-mail.

2.2 The Tables View

In this view the tables of the current web page are provided in a list, similar to the links view. The user can then select a table and jump to it in the document. Therefore, if the user knows that the football scores are at the end of the page in the second table, the user can jump there with a small number of keystrokes. As in the links list the user can listen to the text immediately before and after the table with a single key press.

As opposed to links, tables have no identifiers so the table view is of little use if you're not familiar with the structure of the page. On the other hand if you already know which table you want to look at, then this is a quick way of getting to those scores.

Table navigation is done with the arrow keys. Pressing the modifier keys provides further options like reading the first/last non-empty cell of a row/column. There is also the possibility of navigating inside the table without loosing the current position. If the user presses Alt and the arrow keys, navigation is performed as usual with the arrow keys, but upon releasing the Alt key the user is sent back to the previous position.

2.3 The Forms View

The forms view displays the forms in the web page. As in the other views the idea is to provide the user with a quick shortcut to information. For instance in a search engine, the forms view gives direct access to the input query. Again this feature aims at providing yet another shortcut to the information.

2.4 The Document View

This is the main view, where the whole web page is read. Three modes are available to the user, in each a different perspective of the web page is given. Each mode was built based on our experience during the browser's implementation. We soon realized that there was no unique way if reading all web pages, in a clear way. Furthermore some users will want to have full knowledge of the pages structure whereas others will just want to have the text read to them.

In order to cater for these situations and requests from the users we do provide different ways of reading a web page. One mode gives the user total knowledge of the web page structure, including all the tags, their relation and hierarchy. Navigation is done in the hierarchical structure of the web page using the arrow keys. However this mode adds too much extra text to the page's text, therefore getting the actual text of the page can be cumbersome.

In order to provide a smoother reading, but still have all the information available we also provide a mode where the tags text is eliminated. The text is presented in a hierarchical tree based on the headers (if available). Tables are linearized and the reading is row based. Nevertheless no information is lost because the user can still jump to the tables view and have a non-linearized version of the tables.

The third mode available is for those users who just want a piece of text on the web page. For instance, in a news web site, when jumping to a page for a particular piece of news, it is common to have the links to other sections on the left side. In a talking browser these implies that before reading the actual text, all the links must be read because they appear first in the html code. This third mode removes all the stand-alone links from the text, i.e. a link that has no text before or after the link tag will be excluded from the reading. In practice this generally implies that all navigation links tend to disappear and the text that is read is just the main text on the web page.

In the example of a news web site almost only the text for the piece of news is read. Note that the user can still jump to the links section. However, in some sites using JavaScript some extra text might also be read, therefore delaying access to the information.

At any time the user can switch between the different modes using the keyboard, and select the mode that suits him/her better.

As in the tables view the user can also navigate without loosing the actual position. When pressing Alt together with the arrow keys a temporary cursor is used to move in the document, and the cursor item will be read. Upon the release of the Alt key, the user is sent back to the previous position.

3. Using Scripts for Template URLs

Accessibility has two different meanings for blind and sighted users. These latter users have the ability with a quick glance at a web page to identify the locations where the information is present. Sighted users can go directly to the relevant information avoiding paying any attention to the list of links on the left of the page, or the banner at the top. Blind users, on the other hand, have to cope with all the text of a web page when using a talking browser or a screen reader. In a typical news page, full of navigation elements, links to sponsors, and to many other entries, the blind user must either have all this text read, or do a bit of navigation in the web page, before the tool gets to the actual information.

Even with the different views we already provide in our browser there are pages where the main text of the page is not readily available to the user. For instance, we've tried with pages of a local newspaper that uses JavaScript and this causes a lot of garbage to be presented to the user.

Since site design is very diverse it is impossible to cater for all cases. On the other hand it is extremely easy to define a script to extract the information from a page, or even a set of pages from a particular site. The ability to incorporate scripting in a talking browser, gives the user the best of both worlds. The user can still travel freely on the web, and for a restricted number of selected sites the user is presented with the information right away, without any extras. The fact that this feature is incorporated in a talking browser allows the user to switch to the default views, without scripting; therefore the extras are not lost. Scripting provides the user with a new view of the web page.

Scripts can be written for template URLs, i.e., the URL is defined using wildcards in a similar fashion to how a list of folder or filenames is specified. When the talking browser detects a URL for which there is a script the document is presented using the script to extract information. An audio message is provided to the user in order to let him/her know that a script has parsed the page's content. Using a single keystroke the user can change to the default view, i.e. the full web page.

Writing a single script is a very simple process. However to get to a significant number of sites, time is required to implement all those scripts. Basically this is a never-ending process, where scripts are added in a continuous process. Furthermore web sites do design updates and this may require a script update. With time a large database of scripts can be build, covering the major news sites, search engines, and sites with relevant specific information for blind users.

Because of the dynamic nature of the scripts database we needed a way of keeping the previously installed talking browsers up to date. The solution we found is based on a central server that contains the database. When a talking browser is activated, a connection to the server is made where it is checked if there were any updates to the database, either new or modified scripts. If this is the case then the talking browser will download the new and updated scripts, and store them locally.

4. Conclusions and Future Work

There's still a lot of work to be done before the blind community can fully experience the power of the Internet. The graphical nature of the web implies that they will always be outsiders to some extent. Nevertheless this is no reason to quit, quite the opposite, these problems drive us to improve our browser.

While we believe that we've provided the blind user with a good set of document views, we realize that there are still a considerable number of pages where unwanted text is read to the user. One direction of research to cope with this problem is the study of heuristics to provide new views of a web page.

We also are looking into ways of building a script editor that could be used by the blind user. This may be just a dream, but if we could make it come true then the blind user could define his/her own view of specific web pages. Different users could have different views of the same web page, and they would even be able to share those views. Obviously this feature would be only for power users, still we think that this is a step in the right direction towards increasing the level of accessibility for blind users.

Another issue is multi language support. Right now our browser only speaks in Portuguese, but its implementation was designed having this goal on mind. Furthermore, the interface and pre defined messages can also be translated with very little effort, so having English, Spanish, French, and other versions is a straightforward process.

Acknowledgements
The research reported in here was supported by SNRIP: The National Secretariat of Rehabilitation and Integration for the Disabled under program CITE IV.

References

[1] "Web Content Accessibility Guidelines 1.0", http://www.w3.org/TR/WAI-WEBCONTENT/

[2] "IBM – Home Page Reader", http://www-3.ibm.com/able/hpr.html

[3] "Brookes Talk", http://www.brookes.ac.uk/schools/cms/research/speech/info.htm

[4] "pwWebSpeak", http://www.prodworks.com

[5] Crista Earl, Jay Leventhal, and Koert Wehberg, "A review of IBM Home Page Reader and pwWebSpeak", http://www.onlinejournal.net/afb/AW/1999/1/0/prod_eval2.html

[6] Mary Zajicek, Chris Powell, and Chris Reeves, "A Web Navigation Tool for the Blind", 3rd ACM/SIGAPH on Assistive Technologies, 1998, California.