International Journal of Web Information Systems

Enhancing web accessibility by implementing context aware proxy

Najd Al-Mouh Hend S. Al-Khalifa

Enhancing web accessibility by implementing context aware proxy

Najd Al-Mouh
King Abdul Aziz City for Science and Technology, Riyadh, Saudi Arabia, and
Hend S. Al-Khalifa
Department of Information Technology, King Saud University, Riyadh, Saudi Arabia

Received 14 November 2015
Revised 1 March 2016
Accepted 2 March 2016

Abstract

Purpose – Millions of visually impaired people (VIP) in the world still face difficulties browsing the Web and accessing information. This paper aims to present a proxy service that takes advantage of the concept of context-aware to help contextualizing web pages for visually impaired users.

Design/methodology/approach – The VIP-aware proxy combines five components to utilize the user preferences, adapts the requested web page and reorganizes its content to best match the preferences set by the user. This new scenario will assist VIP in browsing the Web more effectively.

Findings – A preliminary evaluation of the system resulted in general user satisfaction.

Practical implications – The VIP-aware proxy will provide users with a clean, accessible web page, save them time when screen readers examine content related to their preferences and save them money when unnecessary content is not downloaded.

Originality/value – The VIP-aware proxy presented in this paper is the first of its kind targeting VIP.

Keywords Segmentation, Accessibility, Web, DOM, Proxy, Visually impaired people

Paper type Research paper

1. Introduction

People surf the Web daily because it provides information and services, connects people and allows them to communicate (Atzori *et al.*, 2010). Making the Web accessible to all people is very important; any person should be able to navigate, browse and interact online easily. Accessible web pages should be well constructed, flawlessly designed and easy to edit. The content of web pages needs to be accessible to any user, even those with impairments.

Consequently, Web accessibility has aroused the interests of a large number of organizations and working groups in the world of Web technology who all work toward one goal: making the Web accessible to all people regardless of their ability. One of these organizations is the World Wide Web Consortium (W3C), initiated in 1999. W3C has created the Web Accessibility Initiative (WAI) to facilitate Web access to visually impaired users. One of WAI's projects is to develop the Web content accessibility guidelines (WCAG). WCAG has established a number of references to help provide better Web access; it has a total of 14 guidelines that, when followed, will enable Web content to be accessed by a large group of users with impairments. To ensure that a web

International Journal of Web Information Systems
Vol. 12 No. 2, 2016
pp. 201-214

1744-0084
DOI 10.1108/IJWIS-11-2015-0037

page follows these guidelines, Web designers can use validation tools[1] and professional audits[2]. Another alternative is using semi-automatic website repair tools (Asakawa and Takagi, 2000). The best method of ensuring Web accessibility is using Web adaption tools that alter the web page content and present it to visually impaired users in a more usable format.

However, with advancement in technology, web pages are becoming more complex: they contain pictures, fancy designs and rich content. Services provided by the Web have also widened. In fact, people go online for not only browsing but also shopping, searching and communicating. Yet, even with these advancements, web pages are not fully accessible to visually impaired users. In this paper, we present a proxy service that adapts web pages and presents them in a usable way to visually impaired users. Our visually impaired people (VIP)-aware proxy (Al-Mouh *et al.*, 2013) functions as an agent and presents a service that can quickly retrieve key information from any web page.

The rest of this paper is organized as follows: first, we present the motivation for this paper and then summarize the background and other related work. Next, we explain VIP proxy implementation requirements and components. Finally, we discuss the evaluation of the proxy, its results and future directions.

2. Motivation

A web page is usually designed to attract readers' attention. It consists of main information at the top of the page, advertisements in the margins, a banner and other information at the bottom and pictures at different places on the page. However, the situation is totally different when using a screen reader. A screen reader simply parses through the web page and converts what is in the page to speech. When a blind person enters a website using screen reading software, he or she first listens to understand how many lines there are and comprehends the basic structure. If the web page is full of context, there may be 300-400 lines of code, which the software will go through line by line. This process takes a very long time, going until the user finds the desired information.

Screen reading software, such as VoiceOver in OS X or JAWS for Windows, makes the process of reading a web page a little more convenient. Such programs search the web page for headers, give a literal description of the layout and reflect all punctuations. These small features are of use only when the web page designer considers them carefully while designing, although most designers do not. Designers, for example, need to divide large blocks of texts by headers, use alternative text for images and use tags accurately. These are differences one probably cannot see but that some will hear.

Moreover, web page accessibility still has many challenges that need to be solved. It is important to develop Web accessibility evaluation tools that can simulate the users' behavior and help Web developers to overcome accessibility issues (Asakawa, 2005). Also, developers should put their efforts together to create a solution to these accessibility problems.

Based on the previously mentioned challenges, we developed a VIP-aware proxy service that:

- adapts a web page and makes it easy and fast to access its content for VIP;
- converts a web page into HTML5 (HTML: hypertext markup language), the latest version of the hypertext markup language, which can deal with semantic markup and enhance web page's accessibility; and
- allows VIP to choose the best organization of the page's main content (header, content and footer) to best serve his or her needs.

3. Previous work

For the benefit of visually impaired users, there have been many techniques to enhance Web accessibility and extract content from web pages. Recently, researchers have developed several techniques to serve that goal, such as natural language processing, name entity tagging and ontology-based information extraction. In this section, we look at some of these techniques and explore how they have been used to enhance accessibility.

Many tools help developers assess and evaluate Web accessibility and have aided them in meeting accessibility standards, such as the W3C WCAG 2.0. The automatic evaluation of Web accessibility has been a useful method, which allowed developers to detect issues in Web accessibility (Ivory, 2003); however, techniques such as automatic evaluation tools still need development (Bigham and Ladner, 2007). According to the comparisons between different assistive methods that (Mankoff *et al.*, 2005) have been discussed, the most productive one was that of developers reviewing web pages with a screen reader.

Retrieving web page content for visually impaired users is a necessity that has led to the development of several tools and software that mostly use screen readers. Zajicek *et al.* (1998) introduced a prototype browser called "NavAccess" that can assist visually impaired users in browsing the Web. NavAccess was designed on the basis of User Agent Accessibility Guideline and WCAG. It is a navigating tool with a search mechanism that searches web pages and keeps a cache of related pages. On the other hand, Lee (2004) presented an accessibility scaffolding technique that helps in dividing the content of a web page for visually impaired users. His work uses the visual structure of a page to divide its content; that is, the page is divided according to the visual properties of the blocks and the spaces that exist there. Another tool is BrookesTalk (Brown *et al.*, 2001), which uses a speech mechanism to summarize web pages to a visually impaired user. This tool can help the user browse the page easily, and what distinguishes it from other tools is its use of different voices to mark the different parts of a web page. It was built on the basis of "pwWebSpeak" (De Witt and Hakkinen, 1998), the standard tool for the visually impaired.

For more precise extraction, Huang and Sundaresan (2000) presented Aurora. Aurora is a system that extracts web page services rather than content or information. Moreover, it is a semantic transcoding system that assists visually impaired users in extracting services using a semantic model paradigm.

From the previous work, we can see that several tools and systems are available in the field of Web accessibility. Some tools focus on evaluating accessibility, whereas others focus on extracting specific information from a page. However, these tools have not yet solved the problem of slow access to the page's content by screen readers. In our work, we will try to build a system that converts web pages' content into a flexible and accessible form.

4. Visually impaired people-aware proxy system

Our VIP-aware proxy system links the user with the server and controls the communication between them. When the user requests a web page, the proxy processes the requested web page and adapts it to best match the preferences set by the user, assuming the proxy works on a machine with sufficient resources that can correspond to the user's preferences.

The proxy consists of five key modules to perform its job as shown in Figure 1. These modules are basically the adaptation technique followed by the proxy to adapt the original web page and present it in a more suitable way to the end-user. The following sections describe the implementation requirements, the implementation environment and the system components in detail.

4.1 Requirements

The VIP-aware proxy implementation process has some requirements. The proxy is implemented on an adapted interface, which allows it to load a web page, then add and customize JavaScript.

The proxy, on the other hand, performs two main methods: the first is getting the website's uniform resource locator (URL) to access the components of the web page. In this step, the proxy parses the HTML of the requested page and creates a document object model (DOM) which will be cleaned of inaccessible elements. The second method is dividing the final web page into three parts: header, content and footer, which will be reorganized according to the user's preferences to give the user an adapted web page.

The interface where the proxy is implemented needs to provide users with selection boxes or buttons that reside at the top of the display interface. This can simplify browsing for visually impaired users and allow them to switch between the three parts, switch the usage modes and save changes.

4.2 Environment

The VIP-aware proxy was implemented using CSSBox[3], which is an HTML-rendering engine that is mainly concerned with web page layout and content. CSSBox is written in pure Java, which makes the contents of the rendered page and its layout available for

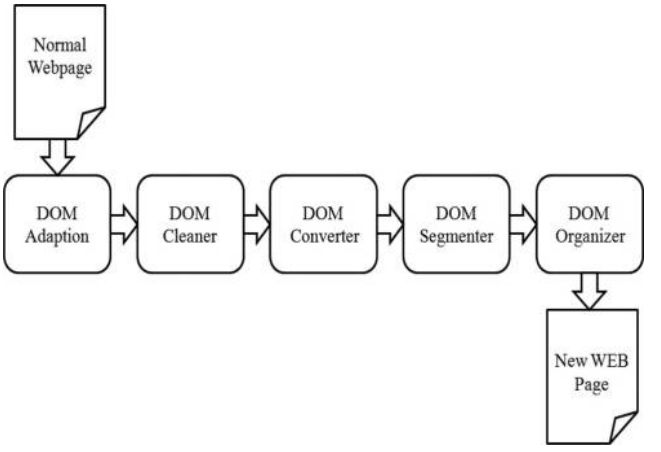


Figure 1.
Process pipeline of web page adaptation in VIP-aware proxy

additional processing. It renders the DOM tree and produces an object-oriented model of the web page, which can be processed further by the layout analysis algorithms.

The accessibility of the output web page of the VIP-aware proxy was tested using Firefox with its add-ons that allow it to be customized and personalized. In this project, two add-ons were used: the accessibility evaluation toolbar[4] and firebug. The former assessed the proxy's accessibility based on Illinois Center for Information Technology and Web Accessibility (iCITA) HTML best practices[5], whereas the former assessed the DOM tree and allowed us to adjust it.

The accessibility of the output page was also tested using nonVisual Desktop Access (NVDA)[6]. NVDA is a free desktop screen reader that mocks both speech and braille. It works on Microsoft Windows systems and with different applications such as Web browsers.

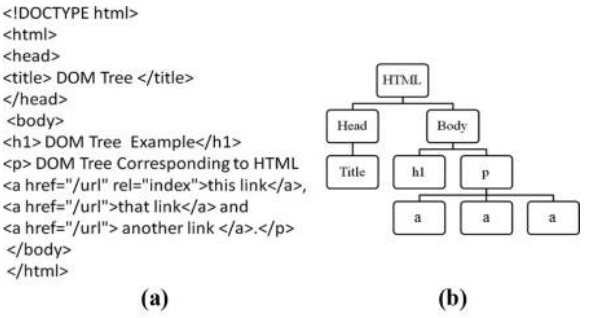
4.3 Components

VIP-aware proxy consists of five key modules to perform its job. The modules are as follows.

4.3.1 Document object model adaption. Based on HTML standards (Raggett *et al.*, 1999), the DOM adaption module analyzes a normal web page into more easily processed components, fills the missing HTML tags and creates a DOM tree that matches the page hierarchy; and is based on DOM criterion (Thors and Wilson, 2000). The DOM tree includes the web page's components, structure and elements that build up the page, such as each element's name, content and attributes, and their relation with each other (Figure 2). The DOM tree makes HTML elements easier to access and operate compared with their text form.

4.3.2 Document object model cleaner. This component removes scripts, cascading style sheets, images and any content that is mostly not read by the screen reader. It is obvious that such information will not help visually impaired users; on the contrary, it consumes more time and costs extra money to be downloaded or located. To clean a web page of unreadable content by screen readers, the DOM tree is used to locate sub-trees with inaccessible content (e.g. images and JavaScripts).

4.3.3 Document object model converter. In this module, the cleaned DOM tree is converted into an HTML5 page. HTML is one of the most common and easiest ways to mark up the content of a web page, and its improvement in the latest version, HTML5, has made it the perfect choice for this module. HTML5 has many enhanced features,



Notes: (a) HTML document; (b) DOM tree

Figure 2.
DOM tree corresponding to HTML document

such as providing content information; compatibility with different devices, especially mobile; and dealing with semantic markup. Furthermore, HTML5 has introduced elements that can more precisely describe their content, unlike general elements, such as . Because these elements have semantic values, they can make accessibility a stand-alone structure. These elements include header, footer, section, article, aside and nav. In this module, the elements of the web page that result from the preceding module are changed to corresponding elements in HTML5.

To facilitate Web access to VIP, W3C has set up WAI-Accessible Rich Internet Applications (WAI-ARIA)[7], which allows a web page, or parts of a web page, to act as an application server instead of a static document by providing the role, attribute and information of the situation which can enhance the web page's accessibility. Therefore, WAI-ARIA has been a proper approach for facilitating access to Web applications for VIP and others with disabilities; it makes it easier for them to use screen readers to navigate and use Web applications.

As a result, our proposed VIP-aware proxy contains both HTML5 specifications and WAI-ARIA support, which will provide it with clearer, more accessible content for VIP.

4.3.4 Document object model segmenter. By using the DOM, this module identifies the content and converts it to blocks, which will be turned into segments. In this module, the vision-based segmentation algorithm is used to segment a web page. It acquires the structure of the web page by using both DOM structure and the visual structure of a web page. It first extracts the visual blocks by applying twelve heuristic rules proposed by Cai *et al.* (2003) and then separates the blocks and builds a vision-based content structure of the web page. Figure 3 shows an example of vision-based content structure for ksu.edu.sa.

Extracting the visual blocks uses the visual properties of the various nodes in the DOM tree structure. These nodes are defined by Akpinar and Yesilada (2013) and Cai *et al.* (2003) as follows:

- *Inline node*: This is any node applied to change the appearance of the text but it does not start a new line. Nodes such as <B>, <BIG>, <I>, <STRONG> and <U> are considered inline nodes.

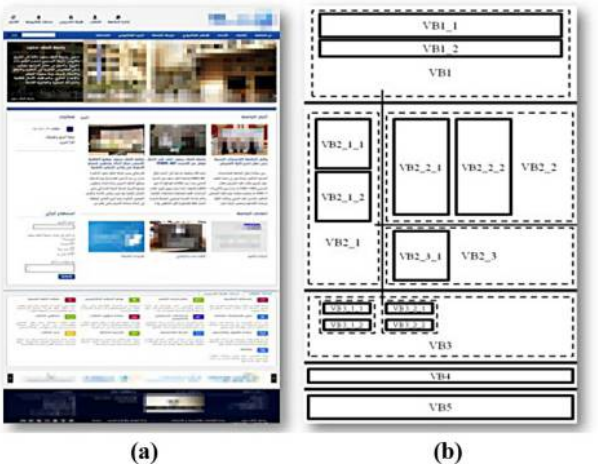


Figure 3.
The web page layout (a) and vision-based content structure (b) for the ksu.edu.sa portal

- *Line-break nodes*: The opposite of inline nodes, these are usually applied to start a new line, such as <TABLE>, <TR>, <UL> and <P>.
- *Invalid nodes*: Any node that does not appear in or affect the appearance of a web page's layout is considered an invalid node, for example, SCRIPT, STYLE, TITLE, etc.
- *Partially valid nodes*: These are nodes that only appear when they have a text node, similar in inline and some line-break nodes, or any visible children. Therefore, when the node is only visible when it is with a child, it is considered a partially valid node. Partially valid nodes might have nodes such as FORM, HEAD, TABLE, DD, DT, etc.
- *Valid nodes*: All the nodes that are visible in the web page's layout are valid nodes, including the partially valid ones with visible children and line-break nodes which are not part of partially valid nodes. For example, UL, LI, DL and BLOCKQUOTE are considered valid nodes because they appear in the page layout as empty spaces when they do not have any children.
- *Margin*: Margins, when their value is not zero, are useful to break up different blocks with space.
- *Font size and font weight*: The ability to find font size and font weight can be useful in identifying where a new topic starts.
- *Float*: Float values can be helpful in placing different nodes in the same block.
- *Images*: They can be used to find headers especially when the header contains a logo.

This module creates three main segments, header, content and footer, which are considered the main parts of any page and, consequently, are used in the following module to get the final adapted web page.

4.3.5 Document object model organizer. The DOM organizer organizes the main segments that result from the previous module (header, content and footer) according to the user's preferences (Schiaffino and Amandi, 2009), which the user has set previously in the application interface. In the preference, the user sets which part (header, content and footer) of these three he or she prefers to be read first by the screen reader. The three work steps of this module are as follows:

4.3.5.1 Getting the user's preference. The first step is to get the user's preference according to which the three most important parts of the page shall be organized. The preferences are as follows:

(1) If the user wants the header to be the first part of the page, he or she chooses the following structure: header, content and footer.

(2) If the user wants the content to be first part of the page, he or she chooses the following structure: content, header and footer.

(3) If the user wants the footer to be first part of the page, he or she chooses the following structure: (footer, content and header).

4.3.5.2 Grouping document object model nodes. The DOM tree nodes' properties are used to group similar nodes and identify the main parts. Chen *et al.* (2003 and 2005) described the flat shapes (shapes with small height/width ratio) at the top of the page as

the header, whereas the ones at the bottom of the page as the footer. Accordingly, a threshold N is defined, and the upper N pixels of a web page are set as the header region, whereas the bottom N pixels of the page are set as the footer region. The threshold N is used in the following formula (Chen *et al.*, 2003; 2005):

$$N = \text{base\_threshold} + F(\text{heightwidth}), \text{ where } F(x) = a/(b * x + c) \quad (1)$$

In this formula, A, b and c are constants with the base threshold:

$$\text{base\_threshold} = 160, a = 40, b = 20 \text{ and } c = 1.$$

4.3.5.3 Changing DOM node position. Finally, this module goes over the whole DOM tree to reposition the main parts in their preferred place. For that to happen, a temporary node value is used to move the wanted node into the requested position. The result of this step is the new adapted layout of the requested web page, organized in accordance with the user's preferences. The output of this module is the final adapted web page, which can be used by visually impaired users and allows them to navigate easily.

These five modules are key in the VIP-aware proxy and will be applied each time a web page is requested. Using the five modules together has added an extra unique feature to specifically help VIP. For example, there are many applications that use the DOM segmenter module alone for different reasons; but using it with the DOM cleaner adds an extra feature to the segmenter, and so for all the other modules.

4.4 Visually impaired people-aware proxy scenario

To show how the VIP-aware proxy can be adopted in realistic environments, we developed a graphical user interface for VIP-aware proxy that is compatible with the Windows operating system. The user first inputs the website URL and chooses how he or she wants the page's parts to be organized, namely:

- Header, content and footer.
- Content, header and footer.
- Footer, content and header.

The interface takes the user's preference and sends it to the proxy, which, in turn, processes the web page and displays it in a reorganized fashion (See Figure 4 for an example).

5. System evaluation

We conducted two experiments to evaluate the result of the VIP-aware proxy: we manually tested it with 50 websites from different domains and asked five VIP who have experience browsing the Web to test the output of the proxy and provide feedback on its quality.

5.1 Manual evaluation

In the first experiment, 50 popular websites in Saudi Arabia, with English and Arabic versions, were chosen to test the performance of our proxy. These websites are from different domains.

At first, we write the URL of each website in the user's proxy interface. Then, after the adaption of the web page by our proxy, we evaluate the results manually and provide a

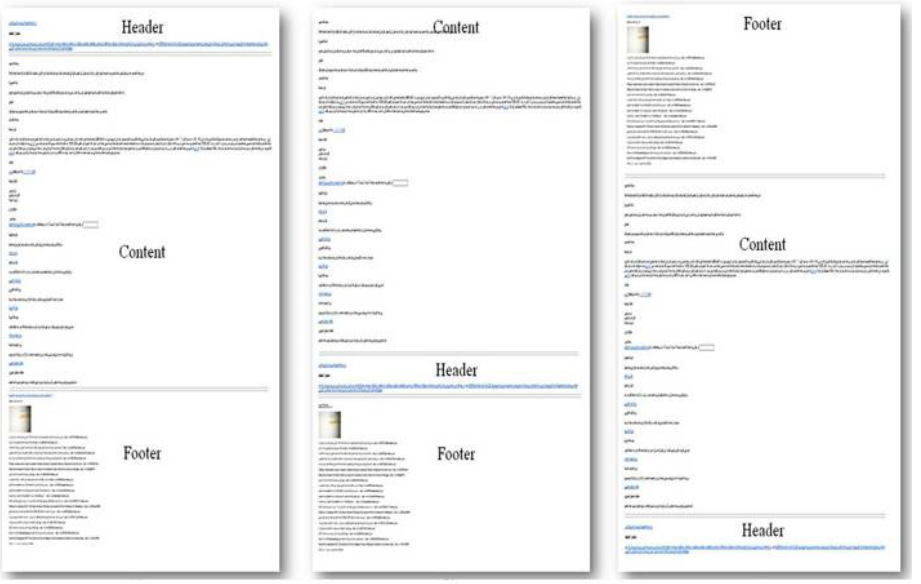


Figure 4.
The VIP-aware proxy output for the ksu.edu.sa portal in different organizations

degree (perfect, good or error) to the output. Perfect is given when the organization is excellent and there is no error in the page. Good is given when the segmentation and organization are correct but there are some small errors which can be neglected because they do not corrupt the view of the resulted page. Error is given when the segmentation and organization process are not correct and some information is lost during the process. For this experiment, we used a personal computer (1.65-GHz central processing unit (CPU) and 4-GB random-access memory (RAM)). The average time of page adaption and processing is 90 s (time varies according to the web page itself).

Table I shows the URLs of the 50 websites, their genre and corresponding results. The result showed that 26 per cent of the websites received a "perfect" result; their segmentation process and their header, content and footer identification were excellent. The websites with the best results either have accessibility characteristics or are W3C certified (marked with * in the table). Next, 38 per cent were rated "good"; this result was for websites with dynamic content that made it difficult to identify their content. The rest of the websites, 36 per cent, got an "error" result, which means content in the page was unidentifiable. Some of the websites that received errors were written in JavaScript, which is difficult for screen readers to read. Other websites were some Arabic websites; although the segmentation and organization processes were successful, the Arabic letters were changed into unrecognized symbols because of the library used in the segmenter, and we could not find an alternative that can recognize all Arabic websites.

5.2 User testing

The second experiment tested how VIP can access the content of the requested page and if he or she can depend on the proxy output to browse a web page. This experiment was conducted by five blind people with knowledge of technology (one male and four

Website	Genre	Result
1. bbc.com/news	News	Perfect
2. ksu.edu.sa	University	Perfect
3. Yahoo.com	Mail	Error
4. aljazeera.net/news	News	Perfect
5. alriyadh.com	News	Good
6. amazon.com	Shopping	Perfect
7. tech2click.net	Blog	Error
8. ar.wikipedia.org	Encyclopedia	Error
9. en.wikipedia.org	Encyclopedia	Good
10. cnn.com	News	Perfect
11. google.com	Search	Error
12. bbc.com/news*	News	Perfect
13. boot.com*	Shopping	Perfect
14. anu.edu.au	University	Error
15. noaa.gov*	Weather	Good
16. mohe.gov.sa*	Government	Perfect
17. moe.gov.sa	Government	Error
18. mopm.gov.sa	Government	Good
19. saudinf.com	Government	Good
20. aawsat.com	News	Error
21. spa.gov.sa	News	Error
22. dubaicustoms.gov.ae	Government	Good
23. citc.gov.sa	Government	Perfect
24. al-madina.com	News	Perfect
25. hasanews.com	News	Error
26. pnu.edu.sa	University	Good
27. flowgo.com	Kids	Error
28. minervation.com	Health	Good
29. samba.com	Bank	Good
30. alrajhibank.com.sa	Bank	Good
31. tripadvisor.com	Travel	Good
32. uqu.edu.sa	University	Error
33. kau.edu.sa	University	Error
34. psu.edu.sa	University	Good
35. cksu.com	Forum	Perfect
36. magrabihospitals.com	Health	Good
37. kfshrc.edu.sa	Health	Error
38. kfmc.med.sa	Health	Error
39. sghgroup.com.sa	Health	Good
40. mep.gov.sa	Government	Error
41. moh.gov.sa*	Government	Perfect
42. alhayat.com	News	Good
43. eyoon.com	Search	Error
44. grants.gov	Government	Good
45. grantsonline.rdc.noaa.gov	Government	Good
46. skysports.com	Sport	Good
47. kooora.com	Sport	Error
48. infospace.com	Search	Good
49. ebay.com	Shopping	Perfect
50. cnet.com	Shopping	Error

Note: *denotes websites with the best results

Table I.
List of tested websites and their corresponding results

females) and aged between 19 and 25 years. The participants were college students with total and partial vision loss. The experiment was conducted on King Saud University portal (both the desktop portal and the mobile version), and participants used their smart phones for the process. Using a Likert scale (5, strongly agree; 1, strongly disagree), participants evaluated the following criteria:

- *Content accessibility*: If it was easy to access the content of the website.
- *Content availability*: If all the web page's content is available.
- *Content organization*: If the organization of the web page's content is as preferred by participants.
- *Navigation*: If it was easy to move through the web page.
- Users' total satisfaction with the proxy's output.

Table II shows the results of the five participants. The results showed users' general satisfaction with the proxy's output and their ability to reach the content in a better way in comparison with the mobile version and the desktop version. Also, the navigation process was smooth and easy. The participants were generally satisfied with the organization of the output page. The criterion with the lowest result was content availability and it was due to the fact that the information on the website was made shorter, that is, only the main content is displayed.

5.3 Discussion

The two experiments we conducted aimed to evaluate the proxy's performance, the satisfaction of visually impaired users with the proxy's output and whether we achieved our goal. VIP-aware proxy can generally process different websites in different domains, layouts and languages, and its output is accessible through mobile devices or personal computers.

The result of the first experiment showed that 64 per cent of the 50 tested websites received "perfect" and "good" results. Most of the websites that failed this experiment did not have accessibility features and were very complicated for visually impaired users to browse. This result has led us to conclude that not many websites, especially Arabic ones, adhere to accessibility guidelines; thus, there is a great need for accessible websites to widen visually impaired users' resources and enable them to deal with any website. In the second experiment, which tested King Saud University portal, VIP

Users	Content accessibility	Content availability	Content organization	Navigation	Comparison with mobile version	Total satisfaction
User 1	5	3	5	5	5	5
User 2	5	5	4	5	5	5
User 3	5	5	5	5	5	5
User 4	4	4	4	5	5	5
User 5	4	5	5	5	4	5
Average	**4.6**	**4.4**	**4.6**	**5**	**4.8**	5

Table II. The result of the second experiment

expressed their general satisfaction with the proxy's output. They also liked the reorganization feature, which saved them time.

6. Conclusion, limitations and future work

The VIP-aware proxy presented in this paper is designed to facilitate Web browsing for VIP. Utilizing context-awareness, this proxy adapts a web page and presents users with a reorganized version that contains concentrated content arranged according to users' preferences and set previously by the user. When users request a web page through the proxy, it goes through five modules: its content will be analyzed to build a DOM tree; the DOM tree will be cleaned of inaccessible content and will be converted to an HTML5 format; and the page is then split into blocks and segments, which will be finally arranged according to the user's preference.

When building this VIP-aware proxy, we faced some challenges, especially during the implementation of the VIP-aware proxy modules. Some of these challenges are listed below:

- Visual presentation can create ambiguity; as a result, it would be hard to identify a segment's boundary.
- Web pages vary in their layouts.
- Web pages are not always tidy and marked properly.
- Processing a web page takes time.
- Dynamic web pages and JavaScript can be an issue to process.
- Dynamic content on a web page also creates some challenges.

Other challenges are due to the proxy's algorithm; although it can give the desired results, there are still some flaws. One of the proxy's flaws is the fact that semantically related segments are not always clustered, they can be scattered all over the page. Also, the segmentation result depends on how the author of the page uses visual cues in the document; the result can be very poor if the visuals are badly used.

These challenges and limitations have resulted in some deficits in the proxy's performance, which lets us consider each website as a case by itself. Also, because of these challenges, it was not easy to reach our goal. We tried to overcome and avoid the challenges by using an appropriate means to get the best possible result to adapt a web page for VIP.

For the future, we plan to improve three main aspects: the interface of the proxy, the segmenter module and the organizer module. First, the proxy's interface needs improvement to make it easier for VIP to use it on mobile devices. Second, we are planning to expand the segmenter by taking more contexts, such as keywords, so that we get more segments related to the user. Third, the organizer module will use a personalization method, which will be based on the user's behavior rather than customization. This would further strengthen our VIP-aware proxy and would add a lot to its value.

Notes

1. www.cast.org/learningtools/Bobby/index.html
2. http://wave.webaim.org/

3. http://cssbox.sourceforge.net/
4. https://addons.mozilla.org/En-us/firefox/addon/accessibility-evaluation-toolb
5. http://html.cita.uiuc.edu/
6. www.nvaccess.org/
7. www.w3.org/WAI/intro/aria

References

Akpinar, E. and Yesilada, Y. (2013), "Vision based page segmentation algorithm: extended and perceived success", *Current Trends in Web Engineering*, Springer International Publishing, pp. 238-252.

Al-Mouh, N.A., Al-Khalifa, H.S. and Al-Salman, A.S. (2013), "Proxy service to contextualize web browsing for the visually impaired", *Proceedings of International Conference on Information Integration and Web-based Applications & Services*, *ACM*, p. 634.

Asakawa, C. (2005), "What's the web like if you can't see it?", *Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A)*, *ACM*, pp. 1-8.

Asakawa, C. and Takagi, H. (2000), "Annotation-based transcoding for nonvisual web access", *Proceedings of the Fourth International ACM Conference on Assistive Technologies*, *ACM*, pp. 172-179.

Atzori, L., Iera, A. and Morabito, G. (2010), "The internet of things: a survey", *Computer Networks*, Vol. 54 No. 15, pp. 2787-2805.

Bigham, J.P. and Ladner, R.E. (2007), "Accessmonkey: a collaborative scripting framework for web users and developers", *Proceedings of the 2007 International Cross-Disciplinary Conference on Web Accessibility (W4A)*, *ACM*, pp. 25-34.

Brown, M.K., Glinski, S.C. and Schmult, B.C. (2001), "Web page analysis for voice browsing", *Proceedings of the 1st International Workshop on Web Document Analysis (WDA'2001)*, *Citeseer*, pp. 59-61.

Cai, D., Yu, S., Wen, J.-R. and Ma, W.-Y. (2003), "VIPS: a vision-based page segmentation algorithm", Microsoft Technical Report MSR-TR-2003-79.

Chen, Y., Ma, W.-Y. and Zhang, H. (2003), "Detecting web page structure for adaptive viewing on small form factor devices.", *Proceeding of the 12th International World Wide Web Conference*, *Budapest*, pp. 225-233.

Chen, Y., Xie, X., Ma, W.-Y. and Zhang, H.-J. (2005), "Adapting web pages for small-screen devices", *IEEE Internet Computing*, Vol. 9 No. 1, pp. 50-56.

De Witt, J.C. and Hakkinen, M.T. (1998), "Surfing the web with pwWebSpeak", *Proceedings of the Technology and Persons with Disabilities Conference*.

Huang, A.W. and Sundaresan, N. (2000), "A semantic transcoding system to adapt web services for users with disabilities", *Proceedings of the Fourth International ACM Conference on Assistive Technologies*, *ACM*, pp. 156-163.

Ivory, M.Y. (2003), "Automated web site evaluation: researchers' and practitioners' perspectives", *Springer Science & Business Media*, Vol. 4.

Lee, A. (2004), "Scaffolding visually cluttered web pages to facilitate accessibility", *Proceedings of the Working Conference on Advanced Visual Interfaces*, *ACM*, pp. 90-93.

Mankoff, J., Fait, H. and Tran, T. (2005), "Is your web page accessible? A comparative study of methods for assessing web page accessibility for the blind", *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, *ACM*, pp. 41-50.

Raggett, D., Le Hors, A. and Jacobs, I. (1999), "HTML 4.01 specification", *W3C Recommendation*, Vol. 24.

Schiaffino, S. and Amandi, A. (2009), "Intelligent user profiling", *Artificial Intelligence An International Perspective*, pp. 193-216.

Thors, A.L.T. and Wilson, C. (2000), "Document object mode 1 (DOM) level 2 HTML specification (Version 1.0)", *W3C Working Draft*, Vol. 13.

Zajicek, M., Powell, C. and Reeves, C. (1998), "A web navigation tool for the blind", *Proceedings of the Third International ACM Conference on Assistive Technologies*, *ACM*, pp. 204-206.

Corresponding author
Hend S. Al-Khalifa can be contacted at: hendk@ksu.edu.sa