Toward Enhancing Web Accessibility for Blind Users through the Semantic Web

Bernard Semaan
LE2I Laboratory UMR-CNRS
University of Bourgogne
21000 Dijon, France
semaanbernard@gmail.com

Joe Tekli
School of Engineering, ECE Department
Lebanese American University
36 Byblos, Lebanon
joe.tekli@lau.edu.lb

Youssef Bou Issa
Faculty of Engineering
Antonine University
40016, Baabda, Lebanon
youssef.bouissa@upa.edu.lb

Gilbert Tekli
R&D Department
Pearl Advisers Offshore
Houston Office, 77251 Houston, USA
gtekli@pearladvisers.com

Richard Chbeir
LIUPPA Laboratory
University of Pau and Adour Countries
64000 Anglet, France
richard.chbeir@univ-pau.fr

***Abstract*—The problems of Web data accessibility and navigation for blind users have become an active research field for the past decade. Many techniques have been created to solve them, some are hardware based and others are software based. Yet, the Web is rapidly evolving toward the far-anticipated Semantic Web (SW): a revolutionary vision extending Web information with well-defined meaning so that it becomes more easily accessible by human users and automated processes. As a result, SW technological breakthroughs such as ontologies and semantic data description, as well as data representation and manipulation technologies (i.e., RDF, OWL, and SPARQL) are being recently explored to improve data accessibility for blind Web surfers. In this paper, we briefly explore existing studies targeting Web data accessibility for blind users, ranging from traditional techniques (Braille output, screen readers, etc.) toward semantically enhanced techniques (using SW technologies). Then, we present an ongoing framework, exploring SW technologies (namely RDF, RDFa and OWL) in order to improve the representation of heterogeneous Web pages (typically pages not conforming to the W3C recommendations for Web page design), adapting the document's contents and presentation so that it best fits the blind user's needs.**

Keywords: Blind users; Web navigation; Braille; Screen readers; touch-screen semantic navigation; Multi-axial navigation; Semantic annotation.

I. INTRODUCTION

Nowadays, the World Wide Web (WWW) has become ever more popular and ubiquitous. More than 800 million users share personal profiles and pictures on Facebook [26], 500 million share personal and professional messages on Twitter [30], and 4 billion data search operations are executed by Web surfers on Google per day [29]. Furthermore, the Web has become a major market of services, where people can buy clothes, book hotels and flights, and even pay their bills and taxes online. This overwhelming success of the Web has been the result of a very fast evolution, gradually adapting and answering the user's needs. It went from the Web 1.0: a simple medium for data access where users merely consume static information, to the Web 2.0: an interactive communication and interchange platform providing the user with an interface to produce and consume information simultaneously using social media (such as chatting, blogs, social networks, and collaborative systems, e.g. wikis), and entering the more sophisticated Web 3.0, known as the Semantic Web, where information is giving well-defined meaning (using ontologies, and description languages such as RDF [17, 24] and OWL [39]) so that it becomes more easily accessible by human users and automated processes.

In this context, blind users, among others, have access to this virtual world, where special purpose techniques like screen readers or talking browsers have been used to facilitate Web data accessibility. Yet, visually impaired users still have difficulty accessing the Web because most webpages are increasingly based on visual information and graphical design [9]. While screen reader and/or talking browser techniques can be used to transform visual input into audio output adapted to the blind user, still for the latter techniques to produce effective results requires "properly-designed" webpages, i.e., pages created following the World Wide Web Consortium (W3C) recommendations and guidelines for webpage design [13] (the page has to be *perceivable*, e.g., providing text alternatives for non text content, *navigable*, e.g., using headings and section headings to organize the page, etc.). Unfortunately, few Webpages (only 8.6 % [28]) are designed with these recommendations. This causes problems for screen readers and/or talking browsers (e.g., text picture buttons are not understandable when alternative tests are missing, missing headings often result in confusing/merging paragraphs) which makes webpages hardly accessible on different devices.

In this context, the objective of our study is to transform any (HTML) Webpage, especially if it does not conform to the W3C recommendations, into a semantically augmented (RDF-based) document aiming to improve the presentation and organization of its contents and make it easily accessible for blind users (regardless of its original coding and/or graphical design). To do so, we introduce a framework entitled SEE: Semantically Enhanced webpage Explorer, allowing to automatically annotate a webpage using RDFa[1] [38], based on a reference ontology (described in OWL [39]) defining the relations between the main blocks of information that can be found in a webpage and their

[1] An W3C recommendation allowing to integrate complex structured information - such as relationships between places, people, and events in a typical HTML webpage

semantic meanings. The automatic annotation process takes into account the input/output modalities of the device at hand (e.g., touch-screen input, voice input, audio output, vibration output, etc.) and user preferences (e.g., accessing the page menu first, then reading paragraph headings, etc.), using mashups [36]. The Webpage will be hence transformed into an RDF-based representation (e.g., Paragraph_1 *Has_Title* "Semantic Web"; Paragraph_1 *Has_Subsection* Paragraph 2; Paragraph_2 *Has_Title* "Ontologies", etc.) which can be more easily rendered in a semantically meaningful way on the blind user's device of choice.

The rest of the paper is organized as follows. Section 2 briefly presents Web accessibility guidelines. Section 3 covers some of the main existing techniques allowing Web data accessibility for the blind. Then, Sections 4 and 5 cover Semantic Web techniques and their usage in Web data accessibility. Section 6 discussed the pros and cons of existing (traditional and semantic-based) solutions, before introducing our framework in Section 7. Section 8 describes our experimental prototype, before concluding in Section 9.

II. WEB ACCESSIBILITY & PROBLEMS

In this section, we first briefly discuss the main W3C guidelines for web accessibility, and then highlight the main problems facing Web accessibility for blind users.

A. Web Content Accessibility Guidelines

Various Web content accessibility guidelines have been proposed by W3C to facilitate accessibility and navigation between Web contents [13], which we attempt to summarize hereunder. In short, web developers are required to:

- Add text alternatives to describing the multimedia object (as some people may not be able to visualize audio, video, pictures and even mathematical charts, based on their device and/or physical disability).
- Properly use markup and style sheets (using markup languages when available, e.g., XML for structured information, MathML for math equations, and using style sheets as CSS to control layout, markup lists and list items properly with "ol", "ul","dl" tags, and use header elements to convey the document structure).
- Ensure that pages featuring new technologies are transformed gracefully (the page should remain accessible on browsers that do not support the latest technologies. Hence, it is recommended to include equivalent text describing dynamic contents, as well as scripts, applets, or other programmatic objects).
- Ensure user control of time-sensitive content changes (allowing users to control flickers and animations on the screen by pausing them, in order to facilitate reading moving text).
- Design webpages which are device-independent (such as the user may access the webpage with a preferred input or output device, e.g., mouse, keyboard, voice, etc. Hence, defining logical event handlers is recommended, instead of using device-dependent event handlers).
- Avoid pop-ups or other windows to appear without informing the user (which may disrupt a blind user).
- Provide context and orientation information (representing the different groups of information in the webpage in a way to preview the relationship between them, such as adding titles for each frame to simplify navigation between them).
- Provide clear navigation mechanisms (adding some orientation and navigation tools, such as: navigation bars, site map, breadcrumbs, etc.) and clearly identify the target of each link (especially if the link consists of a text picture).
- Ensure that the web document layout is clear and simple (containing recognizable graphics and easy to understand language, making them easily accessible).

B. Accessibility Problems for Blind Users

W3C's accessibility guidelines are most of the time disregarded or ignored by Web developers (only 6.8% of the web pages worldwide are conforming to these guidelines [28]). For instance, developers usually do not respect the headings, as well as the ordered and unordered lists tags provided by HTML. Also, they may create complex forms for authentication or information filling purposes, such as the forms are not well designed (following W3C's accessibility guidelines), to be correctly accessible in a certain order while using a talking browser or the *tab* key from the keyboard.

Other problems face a blind user while navigating the web such as: locating herself in the page, understanding character diversity especially for ASCII pictures (i.e., pictures that are made from ASCII characters), and understanding complex structures like tabulations and tables used for webpage design, which makes the document unintuitive for blind users and automated processes alike. Furthermore, many webpages utilize disordered virtual keyboards for password entering (the keyboards are disordered on purpose for security reasons) which may cause confusion for a visually impaired user. Another major problem arises when text images are used for buttons or links, where alternative texts are hardly ever provided, which makes them hardly accessible using traditional web navigation and accessibility techniques (such as speech synthesizers or talking browsers).

In the following, we provide a brief overview of existing web data accessibility techniques for blind users, ranging over traditional hardware techniques and software technologies as well as more sophisticated semantic approaches, highlighting their pros and cons, before introducing our enhanced accessibility framework.

III. EXISTING WEB DATA NAVIGATION

Data navigation techniques for visually impaired users have existed for many years now, like *Speech synthesizers* developed in the 1950's [35], and the *JAWS* screen reader developed in 1989 [20] designed to facilitate data access for a blind user using a computer.

A. Hardware based Techniques

These techniques provide the user with hardware interfaces using the universal language Braille[2] (used by 2 out of 10 blind people [27]). These are implemented through devices like the *Braille keyboard* (data input) and *Braille display* (data output) [41] (cf. Figure 1). Yet, such devices are generally expensive, and need to be mounted and configured manually on each computer system (e.g., PC, tablet, smartphone, etc.) to be utilized by the blind user.

a. Braille keyboard b. Braille display

Figure 1. Main hardware accessibility techniques for blind users.

B. Software based Techniques

These techniques can be implemented on any basic computer system and do not require special hardware:

- **Speech synthesizers** are software tools that transform text-based input into (human) speech output.
- **Screen readers** (e.g., VoiceOver (mac), YASR (Linux), Tiny Voice (Dos) [25]) are software tools that attempt to identify and interpret what is being displayed on the screen of a computer: the text shown on screen is straightforwardly transformed into audio (speech) output, whereas images and other multimedia objects are confined to their alternative text descriptions which are transformed into speech.
- **Talking browsers** [12] are internet browsers that transform an input Webpage into an audio (speech-based) output using speech synthesizers, providing some shortcuts (e.g., list of headings (insert+F6), list of links (insert+F7) [12]). While some of these tools have been shown efficient (such as WebTalkster, Wordread, Browsealoud, etc.) [49], yet they remain confined to reading text-only input that appears on the screen.

While the above software based techniques have been shown practical and efficient, they were designed to provide data output only (transforming and presenting data to the user in audio or sound-based modality). They do not however target data input.

C. Touch-screen based Techniques

More recently, some new techniques have been developed for touch-screens and smartphones highlighting an increasing technological trend nowadays (more than 75% of the world population has cell phones [48]):

- **iPhone VoiceOver** [4] (Figure 2) is one of the main accessibility options provided by the iPhone operating system (iOS). It uses gestures for navigation such as the text on the screen is spoken allowing the user to locate herself on the screen while using the smartphone (e.g., touch or drag finger around the screen and VoiceOver tells what is happening; tap a button to hear a description; double-tap to activate, etc.). While seemingly practical (based on several interviews the authors made with iPhone blind users), this tool is only available on iPhone (iOS) systems
- **Braille Touch** [14] (Figure 3) is another innovative technique using touch-screens, designed to facilitate input text entry by typing in Braille. It is based on six points located on both sides on the screen, representing the Braille matrix that can be used to type letters and numbers in Braille code. This application is practical for Braille writers as no additional hardware accessories are required to type in Braille on the touch-screen. Yet, it is only accessible for users who are familiar with Braille writing, and is restricted to text input (it does not provide an output medium). Also, similarity to *VoiceOver*, *Braille Touch* is only available on iPhone systems.

Similar techniques were developed for Android-based smartphones. A prominent tool is **Slide Rule** [45]: a gesture-based application allowing users to browse messages, emails, music files and contact information. It was specifically designed to overcome two main difficulties when using touch-screens: i) determining the current state of the touch-screen device, and ii) selecting the desired item on screen. The solution is based on 4 main gestures (Figure 2):

(1) Scanning with one finger a list of titles,
(2) Taping with another finger to select the wanted item,
(3) A one finger sliding gesture to the right or to the left in order to switch pages of selected items,
(4) A one finger "L" shaped gesture used to browse hierarchical information such as music artists and corresponding albums, menus, etc.

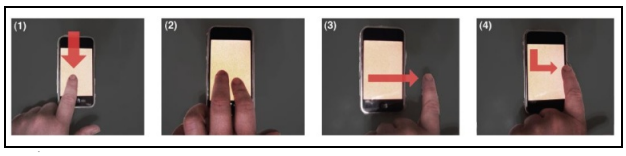

Figure 2. Gestures-based touch-screen manipulation (*SlideRule*).

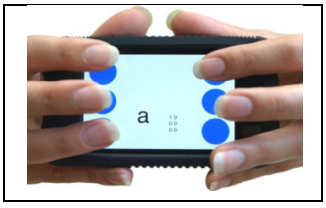

Figure 3. Snapshot of *Braille Touch*.

[2] The Braille system is a method based on tactile contact, widely used by visually impaired people in order to read and write text. A Braille character is made of six dot positions, arranged in a rectangle such as dot may be raised at any of the six positions to form sixty-four (2^6) possible subsets (character encodings).

Some less adopted techniques exists like *Perkinput* [5] for Braille input using touch-screens (developed for android phones, android pads, and also allows to connect two android devices in order to benefit from their adjacent screens simultaneously), and *VBraille* [32] for Braille output: it uses a 6 dots Braille matrix such as when the user touches a dot, the phone vibrates indicating that the dot exists in the character being read.

To sum up, existing Web navigation techniques combine screen reader and talking browser techniques – transforming input webpages into human speech (audio) output [12, 25], with touch-screen (gesture-based) navigation – allowing blind users to navigate between paragraphs, sections and Web links [4, 45], and to type input data using Braille coding [5, 14]. However, most existing methods share two major limitations:

i. They show limited functionality in handling graphical (visual) information (in comparison with text data),
ii. They are not designed to handle heterogeneous webpages, i.e., pages that do not necessarily conform to W3C's accessibility guidelines.

This is due to the fact that most existing navigation techniques target the graphical design and visual coding of webpages, instead of processing the semantic meaning of data contents and related visual information.

IV. SEMANTIC WEB TECHNOLOGIES

In this context, we can generally distinguish between two kinds of files on the Web: i) data documents (e.g., text files, media files, maps, graphs, etc.), i.e., pages presenting raw data designed to be accessed and understood by human users, and ii) information documents (e.g., calendars, contacts, registration info, traveling info, etc.), i.e., pages presenting data to which we associate semantic meaning, which can be stored and manipulated automatically by machines.

This is why the original version of the Web (Web 1.0) was known as the *Web of Documents,* where documents are written in HTML (Hyper-Text Markup Language), uniquely identified by a URI (Uniform Resource Identifier) and linked together through hyperlinks [8], such as the documents are destined to be accessed by users. The traditional Web then gradually grew to meet the requirements and the needs of its users allowing them to better interact with the data and information published online. Websites became increasingly interactive, allowing users to easily exchange ideas, discuss topics, and publish information, which soon drove the Web to another level: the *Social Web* (Web 2.0) where people are involved in publishing and also interacting with other users' published materials [2].

Nonetheless, these large amounts of Web 2.0 data need to be carefully organized, to be more accessible and readable by humans and machines [15]. Hence, the Web 3.0 or Semantic Web (SW) has been promoted as an extended version of the current Web: giving an information a well-defined meaning, i.e., machine-readable semantic descriptions, in order to improve automatic data accessibility and manipulation. The SW is based on two major technological breakthroughs: i) knowledge bases (such as taxonomies and/or ontologies [7, 43]), which provide predefined semantic information references (similarly to dictionaries for human users) to allow the identification and extraction of semantic meaning from raw data, and ii) data representation and manipulation technologies (namely RDF [17, 24] and RDFS [11, 33] for resource description, OWL [3, 16] for ontology definition, and SPARQL [23, 42] for semantic data manipulation and querying). In the following, we briefly describe both of these technological paradigms.

A. Knowledge Bases

An ontology usually comes down to a *semantic network* which is basically a graph consisting of nodes and arcs, organizing words/expressions in a semantic space [44] (Figure 1). Each node represents a concept underlining a group of words. Arcs underline the semantic links connecting the concepts, representing semantic relations (synonymy, hyponymy, etc. [40, 44]). Typical examples of lexical ontologies are Roget's thesaurus [50] and WordNet [40]. For instance, the semantic network in Figure 4 describes an extract of the reference ontology that will be used to annotate webpages. Concepts like *Menu*, *Submenu*, *Advertisement*, *Image*, etc. are linked with semantic relations defining their existence and role in the webpage, e.g., `Submenu` *issubClassOf Menu*, *Advertisement isComposedOf hyperlink*, *text* and *image*, etc. The ontology can be viewed as a formal and explicit specification of a shared representation (conceptualization) of the world for a given purpose [22]. In the example of Figure 4 for instance, our purpose is to represent the relation between HTML elements and visual groups in a webpage.

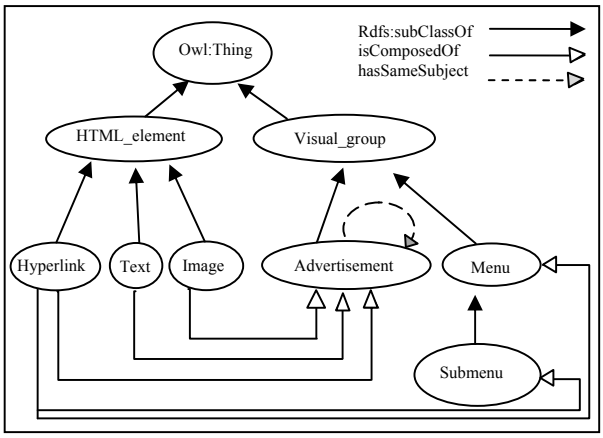

Figure 4. Extract of our reference ontology utilized to semantically annotate webpages.

B. Description Languages

Many languages can be used to represent and describe *knowledge*, to be stored in Knowledge Bases (KBs), available in machine-readable format to be processed automatically by software agents, hoping to achieve semantically enhanced (intelligent) processing capabilities.

Here, we restrict our presentation to RDF [37] and OWL [39] which are the most commonly used on the Web.

RDF (Resource Description Framework) [37] is an XML-based data model designed to standardize the definition and use of metadata, in order to better describe and handle data semantics. Its data representation model is based on triplets (*Object*, *Attribute*, *Value*), more commonly known as *A*(*O*,*V*). A triplet binds an attribute value to an object, giving the relationship a semantic meaning. Objects, attributes and values underline any kind of Web resources, identified using URIs. Values can also contain literal (text) contents. For example, consider two resources: *Hyperlink* and *Advertisement* and property *IsComposedOf*. These can be instantiated as follows: *Hyperlink: www.dummy.com* and *Advertisement*: *Dummy_Ad_1*, *IsComposedOf*(*www.dummy.com*, *Dummy_Ad_1*), such that *www.dummy.com* is a *Hyperlink* and *Dummy_Ad_1* is one of its *advertisements*. This is called an RDF *resource-property-value* triplet. While RDF has a formal semantics with a predefined namespace (i.e., *rdf*) and elements prefix tags (such as *rdf:type*, *rdf:Property*, *rdf:XMLLiteral*, etc.), notions of semantic relations and dependencies [24], yet, it is just a data model describing data instances, and requires the use of a solid, rigorous and well defined vocabulary: data constructs, in order to highlight the intended semantics behind the model.

In this context, OWL (Ontology Web Language) [39] was introduced as a standard for describing semantic data constructs: ontologies on the Web. It models and manipulates classes, defining a class in terms of the properties its instances may have (as briefly mentioned with the *HasName* example above, where the property was defined in terms of its resource). It is built upon RDF and inherits its basic elements (including constructs such as *owl:Class*, *owl:ObjectProperty*, *owl:DataTypeProperty* which extend the expressiveness of their *rdfs:Class* and *rdfs:Property* ancestors) allowing property specialization, the identification of disjoint classes, specifying cardinality or data-type restrictions, etc. [3].

V. WEB NAVIGATION USING SEMANTIC ANNOTATIONS

Navigating the Web using SW technologies has been recently attracting attention in the Web research community, e.g., [6, 19] [9, 10, 34]. In this section, we briefly present the main approaches developed in this field specifically dedicated to facilitate navigation for blind users.

In [6], the authors indicate that a 2D environment is unintuitive to many blind users; blind users prefer stream oriented navigation as if they were reading a book. The authors also indicate that the *search field* (Ctrl+F) and the *table of contents* in a Webpage could help blind users find the information they are looking for. They discuss the concept of multi-axial navigation, where objects with topological, visual, and semantic relations are defined and manipulated at different levels of abstraction, i.e., axis. For instance, one axis could represent the links; another axis could represent the headings in the Webpage, etc. A multi-axial navigation model would allow the user to jump from one axis to another in order to find the information she needs (e.g., navigate all the links in a row, then all the headings). This approach has been applied in the JAWS screen reader for Microsoft personal computers [20], yet it is not fully implemented on mobile devices.

In [19], the authors exploit the HTML document structure to deduce the semantic relations between the components of a Webpage. The proposed technique uses the Document Object Model (DOM) to extract the Webpage organization using the inclusion of components in others (e.g., we can determine the structure of a page from the use of headings, paragraph declarations, etc.). It also identifies blocks structured similarly, and other grouping methods to reorganize the webpage depending on its components' similarities. Nonetheless, this technique depends on the webpage developer's style and her preferences in creating the page (as she may use headings or not, paragraphs or breaks, etc.), and the way she visually partitions her page.

The authors in [34] identify a major problem facing Web data accessibility for blind people: the reconstruction of the Web by transforming it into a text-based stream. They propose a framework based on 3 groups of actors: i) ontology creators, ii) page annotators, and iii) user-agent developers. The ontology creators' job is to create an ontology representing the semantics of a Webpage, the annotators annotate the page using the ontology, and the user-agent developers create the browser or the software that may be used by blind people to access the knowledge in the page. The main limitation of this framework is the amount of manual work required to perform semantic annotation for every single page.

In [9, 10], the authors identify two types of relations between two blocks of information: i) the visual relations that are identified by the page structure (consisting of the text color and font, the neighboring blocks, and visual properties), and ii) the contextual relations that may exist between blocks (e.g., two blocks having the same subject, the same geographic place, such as one represents the details of another, etc.). They propose a framework entitled MAP-RDF [10] based on an RDF representation of visual and contextual relations of a Webpage with a notation language of special symbols (e.g., a menu is represented by ⊙, adds are represented by ⊗, etc.). Three MAP-RDF pages are generated for each Webpage, to be printed and presented to the blind user on embossed paper, allowing the user to better navigate the webpage using tactile perception. While it has been shown effective [9], yet this approach highlights two main limitations: i) embossed paper (and printer) prices are considerable and might not always be affordable, ii) users need to learn the special notation language to understand the webpage. Another study in [18] proposes a unified ontological model for general-purpose websites, including HTML elements (e.g., text, form elements, images, etc.). It presents a multi-axial navigation model for webpages similar to the one used in the JAWS screen reader [20].

On the other hand, a group of studies in [21, 51, 52] have investigated Web data accessibility for the visually impaired based on an analogy between the Web and the real world. The authors consider that a user navigating the Web is spending a "journey" in this virtual world, similarly to a traveler discovering a city (e.g., streets in real world correspond to hyperlinks in a webpage, a city map can be viewed as the map of a Website, etc.). To do so, they use annotations associating Webpage components to real world objects based on their role in the navigation process (e.g., a *hyperlink* can designate *an access* or *a navigation point*; an *image* can indicate a *banner* or *a news add*, etc.). Yet, this technique requires serious training for a user to understand all the travel objects and their annotations. It also requires considerable work for annotating webpages based on their contents and functionality.

VI. DISCUSSION

Accessibility is defined as the minimum time spent by the user to reach a particular piece of information [19]. In this context, blind users face many problems while browsing Internet. As mentioned previously, many hardware and software based techniques have been developed for this purpose. On one hand, most hardware solutions (e.g., Braille Keyboard and Braille output [5, 14]) are generally expensive and require special equipment to be mounted on each computer system. On the other hand, software solutions (e.g., audio-based speech synthesis: screen readers [49] and talking browsers [12], and gesture-based navigation on touch-screens [4, 45] which have been shown especially useful when using smartphones) usually process web contents as they are, i.e., taking into account the syntactic and visual descriptions and tagging of HTML pages, disregarding their semantic properties. In fact, most current websites are based on graphic designs which are not properly tagged. Recently, semantic-based approaches attempt to build on their predecessors: building ontological models [18, 34] to classify information into meaningful semantic blocks, using the page's visual and contextual relations [10], and its DOM structure [19], in order to help the user more easily navigate the needed information without having to read the entire page. Yet, most semantic-based approaches suffer from: i) exploiting complex ontological models, which makes them hardly executable on mobile devices, ii) the considerable amount of manual work required in annotating (preprocessing) the webpages, and iii) most (traditional and semantic) approaches are static in that they do not consider user preferences in adapting the processing/presenting the webpage.

VII. OUR "SEE" SYSTEM PROPOSAL

The objective of our study is to provide an enhanced webpage accessibility approach for visually impaired users. To do so, we introduce a new framework entitled *SEE* (Semantically Enhanced webpage Explorer), which (semi) automatically annotates a webpage, using SW standards (i.e., RDF [11], RDFa [1], and OWL [31]), coupled with Mashup techniques [36], to describe the meaning and organization of its contents, transforming it into a more accessible representation for blind people. For this purpose, we develop an (OWL) ontology model that defines the various semantic concepts and relations describing the different blocks of information in any given Webpage. An extract of the ontology is shown in Figure 4. The ontology is then utilized to automatically annotate and represent the webpage information blocks (in RDF-based coding, namely RDFa), taking into account the user's needs and preferences using mashups, in order to improve user navigation experience (e.g., start the page by presenting the title in Braille coding, then read the introductory text in human speech, after presenting the Website menu by pronouncing menu labels and using gesture-based tactile navigation, etc.). Our method targets the semantic contents of webpages rather than their graphical design and syntactic coding, and thus is expected to efficiently handle visual (graphical) data and heterogeneous webpages, namely pages not conforming to W3C's guidelines, in comparison with traditional (semantic-free) techniques. Also, our approach provides a simple and automated approach, transforming an input HTML page into and RDFa conceptual presentation taking into account the user's feedback and input/output modality preferences (in comparison with existing complex, manual and/or static SW annotation/navigation approaches).

Note that our approach is general and can be used with any traditional navigation technique (Braille-based, audio-based, gesture-based, etc.).

A. Framework Architecture

SEE's architecture model is presented in Figure 5, and consists of four main modules.

1. Automatic Webpage Annotator (AWA): It takes the input OWL reference ontology and the original HTML webpage as an input, and automatically annotates the webpage components and contents with the ontology concepts and relations, producing a semantically rich RDFa page as output.

2. Navigation Protocol Generator (NPG): It generates the navigation protocol (as a set of simple RDF statements) describing the available equipments and how their components are exploited (e.g., touch-screen *Has_Dimensions* 50x50, touch-screen *Used_As* input, accelerometer *Has_Position* slider, etc.). The navigation protocol will be appended, along with user navigation preferences, to transform the semantically enhanced (RDFa) webpage into an output presentation accessible by visually impaired users.

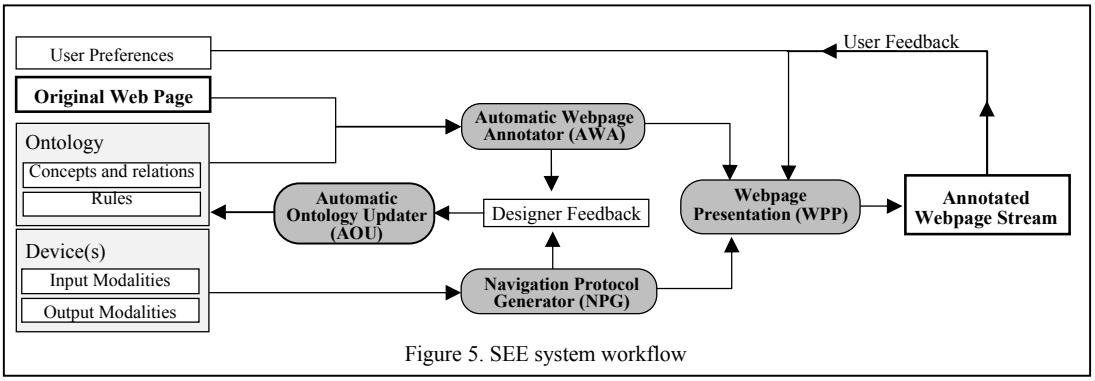

Figure 5. SEE system workflow

3. Automatic Ontology Updater (AOU): It takes the semantically annotated (RDFa) webpage and (RDF-based) navigation protocol, along with the programmer (system designer)'s feedback as input, and updates the reference (OWL) ontology concepts and relations accordingly (modifying, adding, and/or removing concepts/relations).

4. Webpage Presentation (WPP): It takes the annotated webpage, the navigation protocol, and user preferences (expressed as RDF statements) as input, combines them using mashups to create an annotated (RDF based) webpage stream. The stream is then explored on an adapted device browser, considering the device input/output modalities.

B. Creating the Reference Ontology

The reference ontology is initially created manually, and then will be updated automatically depending on the programmer's feedback, given the annotated webpages and navigation protocols. The ontology is written using the expressive OWL language, and includes concepts describing all atomic elements used in a webpage (an atomic element can be a piece of text, a link to another page, an image, form elements, etc.), as well as visual groupings such as menus, the website banner, or any grouping of some visual elements. Data type properties are also included as concept attributes, to provide the descriptions of webpage elements (color, font size, location, etc.), and object properties to express the relation between the elements. Consequently, the visual and contextual relations that may exist between ontology elements are defined. Later on, scripts can also be added to the ontology. These scripts can be used for many purposes like the validation of a form, creating visual effects, and dynamic websites (e.g., receive new emails without refreshing the page, chatting on the web, etc.).

C. Adding Semantic Annotations using RDFa

The reference ontology is then used to automatically annotate the webpage, transforming it into an RDFa document.

We recall that RDFa is a W3C recommendation (released in 2008) providing several features that enable webpage designers to integrate complex structured information (such as relationships between places, people, and events) in typical HTML or XML based documents. RDFa Lite is a minimal subset of the full recommendation, consisting of a simple group of attributes that can be easily learned by webpage designers in less than a day [38]. RDFa Lite attributes are: vocab, typeof, property, resource and prefix. The **vocab** attribute identifies the vocabulary (i.e., the reference ontology, expressed in RDF or OWL) encompassing the concepts that will be used in webpage annotation, the **typeof** attribute expresses the ontology concept representing the type of elements being annotated (add, menu, submenu, etc.), and the **property** attribute includes the information (ontology concepts) related to the current concept we are annotating (stored in *typeof*), as shown in Figure 6. To create an identifier for a specific element in the page, the **resource** attribute is used by adding a hash tag with the resource name. Semantic relations between concepts in the annotated webpage are created using the *resource* attribute to specify the link between different HTML blocks. Finally, the **prefix** attribute can be applied when using different vocabularies to specify the vocabulary path and use the prefix in the current document.

Hence, we adopt the RDFa Lite specification in our framework to link the reference ontology (vocabulary) concepts with webpage elements (or groups of elements) to semantically enrich the webpage, as described above.

Trident competition
The top ten most voted designs get to complete in a real bathub race! Go Tridenters!
Click for more details.

```
<p> <img src="123.jpg" alt="my picture"> The top ten most voted designs get to complete in a real bathub race! Go Tridenters! <a href="www.myadd.com">click for more details <a/> </p>
```

a. Sample HTML code.

```
<p vocab="http://myontology.org/" prefix= "mo: http://myontology.org/"
    typeof="Advertisement" resource="#myadd1">
   <span property="title"> Trident competition </span>
   <span property="image">
        <img src="123.jpg" alt="my picture">
   </span>
   <span property="text"> the top ten most voted designs get
        to complete in a real bathub race! Go Tridenters !
   </span>
   <span property="Hyperlink">
        <a href="www.myadd.com">click for more details<a/>
   </span>
</p>
<span about="#myadd1" property="mo:hassamesubject">
    <span about="#myadd2"></span>
</span>
```

b. RDFa annotation of HTML code.

Figure 6. Sample webpage semantic enriched using RDFa Lite.

Consider for instance the sample webpage in Figure 6.a, describing an advertisement. The resulting semantically enriched page, using RDFa Lite is shown in Figure 6.b. Here, we utilize the *vocab* and *prefix* attributes to specify the ontology path. The *typeof* attribute highlights the HTML block type being annotated, which in the example is an *a*dvertisement. The *property* expresses the elements of the advertisement: an image, a text, and a Hyperlink. Finally, the *resource* attribute defines the relation between the current advertisement (identified as *#myadd1*) and another advertisement (identified as *#myadd2*). Once both of the blocks are nominated, we create an empty *span* element, and use the *about* attribute to specify which block describes the object property domain. Then, we create another span tag to specify the object property range. In our example, the domain is the *#myadd1*, the property is *hassamesubject*, and the range is *#myadd2*.

D. Integrating User Preferences using Mashups

Once the original webpage has been semantically annotated using AWA (Web Page Annotator), producing an RDFa-based document as described previously, and once the navigation protocol has been generated using the NPG (Navigation Protocol Generator) in the form of RDF-based statements, these are forwarded as input to the Mashup layer integrated within the WPP (Web Page Presentation) module.

Mashups underline the concept of reusing and combining existing services, designed for data manipulation and circulation on the Web [36]. Their objective is to allow non-expert users (in our case: non-expert smartphone users/programmers) to manipulate data using dedicated manipulation operations (ranging from simple data selection/projection to data modification: insertion, removal, obfuscation, etc.) [46], which are usually made available using a graphical interface [47]. Typical Mashup applications [36] can include Mashups using maps (i.e., Google maps and Yahoo map3), multimedia content (i.e., YouTube and Flicker videos), e-commerce services (i.e., amazon.com and ebay.com) and news feeds (i.e., RSS and ATOM).

In our current approach, we utilize Mashups in adapting the (semantic) contents of RDFa-based webpages, taking into account RDF-based protocol navigation statements (describing the hardware device's modalities and functionalities), into a webpage presentation which is adapted for the blind user. Here, Machups can be used by both expert and non-expert programmers depending on the mode they choose: manual or semi-automatic, to produce the output page to be perceived by the blind. In Manual Mode: user preferences are graphically defined using concatenations of manipulation operations, creating Mashups to describe their needs. In Semi-Automatic Mode: user preferences are defined as RDF-statements which are then processed automatically in the Mashup layer. Once the Mashup operation is defined, RDF-based webpage streams can be generated from the RDFa page, considering the navigation protocol defined previously, hence producing the output page targeting the blind surfers' preferences.

VIII. EXPERIMENTAL PROTOTYPE

We have developed an experimental prototype to test and evaluate our proposal, in the form of a sophisticated mobile application Web browser entitled SEE. In the following, we briefly describe the prototype implementation, before describing a typical scenario using our running example in Figure 6, and then discussing our ongoing experiments.

A. Experimental Implementation & Operation

We designed our system to run on a basic smartphone Android 2.2 operating system (as minimum requirement). As a preliminary step, we chose to implement the simplest input/output modalities we were able to identify (based on several interviews with blind Web surfers[3]):

- For input: gestures, applied and processed on a typical smartphone touch-screen[4] to provide input commands,
- For output: voice synthesis using the phone's speech synthesizer for reading on-screen text, and vibrations to denote a situation-change action (e.g., alert box, switching to another page, turning off the application, etc.) that could disorient the user.

In this context for instance, the Navigation Protocol Generator (NPG) generates the following group of statements to describe the functionalities of the device:

- touchScreen *Used_As* input
- File(gesturesLibrary) *Defines* inputGestures
- File(gesturesLibrary.xml) *Matches* inputActions
- File(gestureNavigationManual.txt) *Defines* inputManual
- speaker *Used_As* output
- vibrationMotor *Used_As* criticalOutput

To set up our SEE application on the smartphone, we implement the *android.gesture* package to allow using gestures as input on the touch-screen, the *org.apache.http* package for accessing webpages, and the *org.jsoup* library to allow HTML manipulation. The gestures adopted to provide input commands are defined with the Android *GestureBuilder* tool, and saved in a (binary) library file: *gesturesLibrary.bin*. We also create a dedicated XML file *gesturesLibrary.xml* which specifies the correspondences matching gestures with the commands to be performed when each gesture occurs. We typically adopt the following gesture matchings in our initial configuration: i) arc shape to use the axis, ii) slide, left or right, for next or previous item, iii) L shape to access the parent item, iv) double tap to read item or validate a button, v) slide up pauses the speech or cancels an alert, and vi) slide down resumes the speech. We also add a dedicated user guide: *NavigationManual.txt* including descriptions of all gesture navigations, as well as all kinds of Mashup manipulations which could be performed when adapting the webpage presentation.

[3] Over 20 interviews were organized with blind students, aged between 16 and 21, from the National Lebanese School for the Blind and Deaf.

[4] Samsung Galaxy Ace S5830 and Galaxy Tab 10.1 were used for testing.

B. Typical Scenario & Running Example

Consider a typical webpage containing the add shown in Figure 6. The process of accessing the page using our SEE framework prototype is depicted in Figure 7.

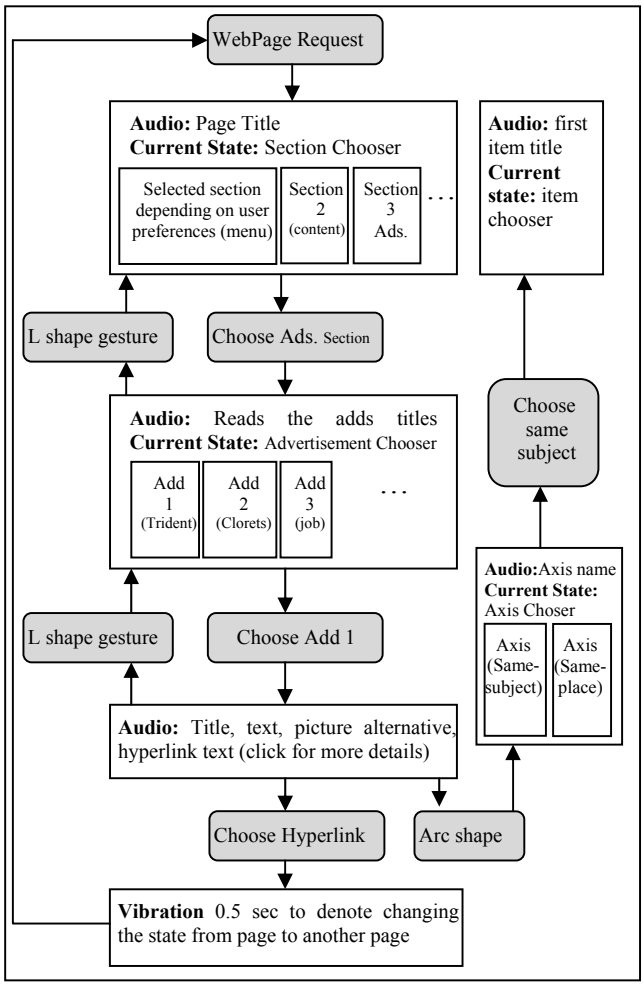

Figure 7. Typical scenario workflow.

Once the user requests the page, the application will read its title in audio (e.g., "Page Title") and order its items depending on the user preferences, which are organized by default as follows: i) "webpage organization", ii) "page content", iii) "advertisements", and iv) "site map". The user then chooses the section she needs using gestures, such as: sliding right or left the application will speak the section titles mentioned above, and a double tap will enter the selected section. To go back one step, the user can draw the L shape gesture which tells the application to read the section titles again (ads, site map, etc.).

In this example, we suppose the user chooses the advertisements section (containing the ad described in the example in Figure 6). Here, the application starts reading the different ad titles: i) "Trident competition", ii) "Clorets", iii) "Job opportunity", etc. To choose an advertisement, the user slides right or left, and then double taps to access. Following the user' choice, the application speaks out the corresponding ad title, then the ad's text, such as: "The top ten most voted designs…", as well as the image alternatives, such as: "my picture", and hyperlink text which is preceded by word "link", such as: "link click for more details". The same process can be used for any other item in the webpage.

To access the semantically related items in the page, the user can switch to the "axis view" by drawing an arc shape, the application will speak the available axis ("sameSubject", "sameGeographicPlace", etc.). Once the user chooses an axis (e.g., "sameSubject" axis) by sliding and double tapping, she accesses the items of the axis. The application will read the titles of these items and the user can slide and double tap to access any of them. If the user chooses to access the advertisement hyperlink, the application will produce a little vibration to indicate a state change (i.e., the user is navigating to a new webpage), the application speaks the page title (e.g., "my second page") and then the same process is repeated in accessing the new page.

C. Ongoing Experimental Evaluation

We are currently performing our prototype evaluation experiments, where we aim to study three main criteria:

i. *Localization*: the blind user's ability to correctly determine her location in a webpage,

ii. *Precision*: the blind user's ability to find specific information in a webpage,

iii. *Performance*: the blind user's speed in finding the information needed in a webpage.

IX. CONCLUSION

In this paper, we studied the problems of web data accessibility and navigation when conducted by blind users, and presented current solutions as well as their drawbacks (mainly related to high costs and low expressive power).

We also presented our framework, called *SEE* (Semantically Enhanced webpage Explorer), designed to annotate webpages using Semantic Web technologies (RDF, RDFa, and OWL), allowing us to focus on the meaning of pages' contents with addressing their syntax and W3C formatting guidelines, and associated to Mashup techniques to describe the meaning and organization of pages' contents, allowing us to ease the output design since mashups can be used by both expert and non-expert programmers. To achieve this, we developed an (OWL) reference ontology to define main semantic concepts and relations describing the different blocks of information in a webpage. The overall process can be summarized as follows: i) annotating the Webpage using our reference ontology, ii) generating the page stream depending on the user preferences, iii) choosing the right modules of the browser depending on the input/output modalities to navigate the stream, iv) updating the reference ontology based on the semantically annotated page.

To validate our approach, we implemented a prototype running on a smartphone Android 2.2 operating system, and conducted a set of tests. First results are very promising. Other tests are currently conducted with different profiles of blind people to evaluate additional criteria related to the localization, precision and performance of our approach.

ACKNOWLEDGMENTS

This study is partly funded by the CEDRE research collaboration program, project *AO 2011*, entitled: *Easy Search and Partitioning of Visual Multimedia Data Repositories*, funded by the French CNRS (National Center for Scientific Research) and the Lebanese CNRS.

We also like to express our deepest gratitude for the testers who participated in this work, namely the students of the Lebanese School for Blind and Deaf, Baabda, Lebanon.

REFERENCES

[1] Adida, B., C. Commons, and M. Birbeck, *RDFa Core 1.1.* URL http://www. w3. org/TR/rdfa-syntax, 2004.

[2] Anderson P., *What is Web 2.0? Ideas, technologies and implications for education.* JISC Technology and Standrads Watch, 2007. pp. 64.

[3] Antoniou G. and Van Harmelen F., *Web Ontology Language: OWL.* Handbook on Ontologies, 2004. pp. 67-92.

[4] Apple, *VoiceOver for iOS,* (Page Accessed June 2013). http://www.apple.com/accessibility/ios/voiceover/

[5] Azenkot S., Wobbrock J. O., Prasain S., and Ladner R. E., *Input Finger Detection for Nonvisual Touch Screen Text Entry in Perkinput.* Proceedings of Graphics Interface (GI'12), Toronto, 2012. pp. 121-129

[6] Baumgartner, R., Ruslan Fayzrakhmanov, Rafael Gattringer, Max Göbel, Wolfgang Holzinger, David Klein, and Bernhard Kruepl, *Web 2.0 vision for the blind.* 2010.

[7] Baziz M. *et al.*., *A concept-based approach for indexing documents in IR.* INFORSID 2005, 2005. pp. 489-504, Grenoble, France.

[8] Bertails A., Herman I., and Hawke S., *The Semantic Web: The Internet and Tomorow's Web.* Industrial Realities (in French), 2010. pp. 80-89.

[9] Bou Issa Y. *et al.*, *Analysis and Evaluation of the Accessbility to Visual Information in Web Pages.* Inter. Conf. on Computers Helping People with Special Needs (ICCHP), 2010. Vienna, Austria.

[10] Bou Issa Y., Mojahid M., Oriola O., and Vigouroux N., *Accessibility for the Blind: Rendering the Layout of the Web Page for a Tactile Experience.* European Conf. for the Advancement of Assistive Technology in Europ (AAATE'09), 2009. Florence, Italie, IOS Press.

[11] Brickley D. and Guha R. V., *RDF Vocabulary Description Language 1.0: RDF Schema.* W3C Recommendation, 2004. http://www.w3.org/TR/rdf-schema/.

[12] BrowseAloud, http://www.browsealoud.co.uk/page.asp?pg_id=80004, Page accessed on Juen 26th 2013.

[13] Caldwell B., Cooper M., Guarino Reid L., and Vanderheiden G., *Web Content Accessibility Guidelines (WCAG) 2.0.* World Wide Web Consortium (W3C), 2008. http://www.w3.org/TR/WCAG20/.

[14] Caleb S., Clawson J., Frey B., Abowd G. D., and Romero M., *An evaluation of BrailleTouch: mobile touchscreen text entry for the visually impaired.* MobileHCI'12, San Francisco, CA, USA, 2012.

[15] Cullot N., Parent C., Spaccapietra S., and Vangenot C., *Ontologies: A contribution to the DL/DB debate.* In Proc. of the 1st International Workshop on the Semantic Web and Databases, in VLDB Conf., 2003.

[16] Dean M. and Schreiber G., *OWL Web Ontology Language Reference.* W3C Recommendation, http://www.w3.org/TR/owl-ref/. 2004.

[17] Decker S., Melnik S., Van Harmelen F., Fensel D., Klein M.C.A., Broekstra J., Erdmann M., and H. I., *The Semantic Web: The Roles of XML and RDF.* IEEE Internet Computing, 2000. 4(5):63-74.

[18] Fayzrahmanov R. R., Göbel M. C., Holzinger W., Krupl B., and Baumgartner R., *A unified ontology-based web page model for improving accessibility.* Proc. of WWW '10), 2010. pp. 1087-1088

[19] Fernandes A. R. *et al.*, *Transcoding for Web Accessibility for the Blind: Semantics from Structure.* ElPub 2006 --- Digital Spectrum: Integrating Technology and Culture, 2006.

[20] Freedom Scientific, *JAWS Headquarters.* http://www.freedomscientific.com/jaws-hq.asp, Page accessed on June 25th 2013. St. Petersburg, Florida, USA.

[21] Goble C., Harper S., and Stevens R., *The Travails of Visually Impaired Web Travellers.* Proceedings of the 11th ACM on Hypertext and Hypermedia (HYPERTEXT '00), 2000. pp. 1-10

[22] Gruber, T.R., *A translation approach to portable ontology specifications.* 1993: p. 199-220.

[23] Hartig O., Bizer C., and Freytag J.C., *Executing SPARQL Queries over the Web of Linked Data.* International Semantic Web Conference (ISWC'09), 2009. pp. 293-309.

[24] Hayes P., *RDF Semantics.* W3C Recommendation, http://www.w3.org/TR/rdf-mt/. 2004.

[25] High Tech Center Training Unit - California Community Colleges, *Introduction to ScreenReaders.* http://www.htctu.net, 2011.

[26] http://www.internetworldstats.com/facebook.htm, *Facebook Usage.* Page accessed on : June 26th 2013.

[27] http://www.perkins.org/vision-loss/gayleunplugged/braille-in-real-life.html, *Braille in Real Life.* 2008.

[28] http://www.standards-schmandards.com/exhibits/validator2/stats/, *Validation statistics.* Page accessed on: June 26th 2013.

[29] http://www.statisticbrain.com/google-searches/, *Google Annual Search Statistics.* Page accessed on: June 26th 2013.

[30] http://www.statisticbrain.com/twitter-statistics/, *Twitter Statistics.* Page accessed on : June 25th 2013.

[31] http://www.w3.org/TR/owl2-overview/, *owl 2.0.*

[32] Jayant C., Acuario C., Johnson W. A., Hollier J., and Ladner R. E., *VBraille: Haptic Braille Perception using a Touch-screen and Vibration on Mobile.* The ACM SIGACCESS International Conference on Computers and Accessibility (ASSETS'10), 2010.

[33] Klyne G. and Carroll J., *Resource Description Framework (RDF): Concepts and Abstract Syntax.* . W3C Recommendation REC-rdf-concepts-20040210, 2004. http://www.w3.org/TR/rdf-concepts/.

[34] Kouroupetroglou C., Salampasis M., and Manitsaris A., *A Semantic-web based framework for developing applications to improve accessibility in the WWW.* Proc. of the 2006 Inter. Cross-Disciplinary Workshop on Web accessibility (W4A '06), 2006. pp. 98 - 108

[35] Lessac Technologies, *Text-to-Speech.* http://lessactech.com/, Page accessed on June 15th 2013.

[36] Lorenzo G. D. et al., *Data integration in mashups* SIGMOD Record, 2009. 38:59-66.

[37] Manola F. and Miller E., *Resource Description Framework (RDF) Primer : Model and Syntax Specification.* W3C Recommendation, 2004. http://www.w3.org/TR/rdf-primer/.

[38] Manu Sporny, D.B., Inc., *RDFa Lite 1.1.* W3C, 2012.

[39] McGuinness D.L. and Van Harmelen F., *OWL 2 Web - Ontology Language Document Overview.* W3C Proposed Edited Recommendation, 2012. http://www.w3.org/TR/owl2-overview/.

[40] Miller G., *WordNet: An On-Line Lexical Database.* International Journal of Lexicography, 1990. 3(4).

[41] Perkins Products, *Braille keyboard, Braille readers.* https://secure2.convio.net/psb/site/Ecommerce/, Page accessed on June 25th 2013.

[42] Prudhommeaux E. and Seaborne A., *SPARQL Query Language for RDF.* W3C Recomm., 2008. http://www.w3.org/TR/rdf-sparql-query/.

[43] Resnik P., *Using Information Content to Evaluate Semantic Similarity in a Taxonomy.* Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1995. Vol 1, pp. 448-453.

[44] Richardson R. and Smeaton A., *Using WordNet in a Knowledge-based approach to information retrieval.* Proceedings of the BCS-IRSG Colloquium on Information Retrieval, 1995.

[45] Shaun K., Bigham J. P., and Wobbrock. J. O., *Slide Rule: Making Mobile Touch Screens Accessible to Blind People using Multi-touch Interaction Techniques.* The ACM SIGACCESS International Conference on Computers and Accessibility (ASSETS), 2008.

[46] Tekli G., Chbeir R., and Fayolle J., *XA2C: a framework for manipulating XML data* International Journal on Web Information Systems (IJWIS'11), 2011. 7(3):240-269.

[47] Tekli G., Chbeir R., and Fayolle J., *A Visual Programming Language for XML manipulation.* International Journal of Visual Languages and Computations, 2013. 24(2): 110-135.

[48] The World Bank, *Mobile Phone Access Reaches Three Quarters of Planet's Population.* http://www.worldbank.org/en/news/press-release/2012/07/17/mobile-phone-access-reaches-three-quarters-planets-population, 2012.

[49] Web Accessibilit in Mind (WebAIM), *Screen Reader User Survey #4 Results.* http://webaim.org/projects/screenreadersurvey4/, 2012.

[50] Yaworsky D., *Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora.* Proceedings of the International Conference on Computational Linguistics (Coling), 1992. Vol 2, pp. 454-460. Nantes.

[51] Yesilada Y. *et al.*, *Ontology based semantic annotation for enhancing mobility support for visually impaired web users.* In K-CAP 2003 Workshop on Knowledge Markup and Semantic Annotation, 2003.

[52] Yesilada Y. *et al.*, *A Foundation for Tool-based Mobility Support for Visually Impaired Web Users.* Proc. of WWW Conf., 2003, 422-430.