VoxBoox: A System for Automatic Generation of Interactive Talking Books

Aanchal Jain and Gopal Gupta

Department of Computer Science
University of Texas at Dallas, Richardson, TX 75080

Abstract. We present the VoxBoox system, a system for making digital books accessible to visually impaired individuals via audio and voice. This is accomplished by automatically translating a book published in HTML to VoiceXML, and then further enhancing this VoiceXML rendering of the book to enable listener-controlled dynamic aural navigation. The VoxBoox system has the following salient features: (i) it leverages existing infrastructure since the book that is to be made accessible need only be published digitally using HTML on the visual Web, (ii) it is based on accepted Web standards of HTML and VoiceXML and thus books can be made accessible inexpensively, and (iii) it is user-centered in that the listener (the user) has complete control over (aural) navigation of the book. In this paper, we present details of the technologies that make the VoxBoox system possible, as well as the details of the system itself. A prototype of the VoxBoox system is operational.

1 Introduction

Even though we have witnessed great leaps in technology in recent times, access to documents for people with print disabilities is still not seamless. Almost everything is digital today: books and papers are published online; even printed books are composed with the aid of the computer first and then they are printed in hard copies. However, even with books and papers available in digital form, there are significant impediments to making them accessible to the visually impaired. In this paper, we report on our efforts to make books easily accessible to the visually impaired readers by automatically turning them into talking books browsed using audio and voice. Our goal is to produce a solution that (i) leverages the existing infrastructure, (ii) uses widely accepted standards, (iii) uses existing inexpensive technology, and (iv) is user-centered. The culmination of our efforts is the VoxBoox system. The design of the VoxBoox system is based on our earlier efforts to make the Web accessible to the visually impaired individuals [7], a significantly harder problem if we observe the criteria mentioned.

The VoxBoox system automatically converts digital books, published on the visual Web using HTML, into talking books. The VoxBoox system does so by automatically translating books published using HTML markup to VoiceXML markup. HTML is the notation used for marking-up documents on the Web for

C. Stephanidis (Ed.): Universal Access in HCI, Part III, HCII 2007, LNCS 4556, pp. 329–338, 2007.

proper visual display. VoiceXML, in contrast, is a notation used for marking-up voice and audio documents. Thus, as soon as a book is published on the visual Web using HTML, with the help of our system it immediately becomes available in voice/audio. The VoxBoox system further enhances the VoiceXML rendering of the book to facilitate interactive, user-controlled voice and audio-based navigation of its content. Thus, using the VoxBoox system, not only can a (blind or sighted) individual hear the book through audio, he/she can also interactively navigate the book via voice commands.

The VoxBoox system leverages the current infrastructure of the Web to make books accessible to visually impaired. The VoxBoox system relies on the technologies of HTML and VoiceXML. HTML is a well-accepted standard for publishing information on the Web. A significant number of books are available in HTML format on the Web (see for example http://onlinebooks.library.upenn.edu), or they are available in formats that can be readily translated into HTML (such as pdf). Likewise, VoiceXML is a standard, well-established technology, albeit most of its usage has been in providing automated Voice based interaction over the phone [10]. With the help of VoiceXML technology, the VoxBoox system makes it possible for books to be interactively and aurally accessible over cell phones and land-line phones.

Our efforts to make published material available to the blind are along the lines of the DAISY (Digital Accessible Information SYstem) project [3]. The rest of this paper describes the VoxBoox system in detail. Finally, note that our Voice based approach not only enables eyes-free reading for the visually impaired, it also enables voice interactions with the document for the sighted individuals and thus is useful for sighted population as well.

2 Talking Books

Digital Talking Books can be regarded as electronic files, which are accessible via audio. According to DAISY [3], an organization dedicated to helping people with print disability, talking books are a collection of electronic books that are presented to visually impaired people through human or synthetic speech [1,2,4]. Currently, there are two common solutions to obtain talking books: recorded cassettes and Digital Talking Books (DTB) by the DAISY consortium.

2.1 Recorded Cassettes

Recorded cassettes are audio files that have specially been recorded on an audio cassette using human voice in order to help blind individuals "read." There are many libraries across the United States such as the California state library, Washington Talking Books and Braille library, Texas State Library, etc., which produce such cassettes. There are about 672,000 such recorded cassettes available around United States libraries [2]. Approximately 1000 copies each are produced annually by the National Library Service for the blind and physically Handicapped (NLS). These are distributed on 4-track 15/16 ips cassettes through 140

regional and subregional libraries [1]. The key point here is that every year there is a significant effort and money that is spent on preparing these cassettes. They are not only expensive but are also played and navigated manually. Also it is not possible to record millions of publications that are produced every year instantaneously. There are still so many books that are published and are not available to blind individuals because they have not been recorded yet.

2.2 Daisy Talking Books

Daisy (Digital Accessible Information System), formed in May 1996 by talking book libraries, is a consortium that works to provide transition from analog to digital talking books. The vision of the DAISY consortium is to make all published material accessible to people with print disabilities inexpensively [3]. The Digital Talking Books (DTB) by DAISY is a PC compatible software audio player. The player plays files that have been developed in accordance with the DTB format and provide facilities like book marking, highlighting, keyword search, etc [4].

The problem with the DAISY's approach is that, first, a book has to be published in the DTB electronic format, which can be a difficult task (however, it is possible for this conversion to be automated) and, second, navigation has to be done via a computer keyboard. The main disadvantage of the DAISY approach is that a computer is needed to use the system. In many developing countries such as India and China, people can afford mobile phones but they still cannot afford computers. Digital talking books available in the DAISY format can also be played using a DTB player (similar to a cassette player), however, this player has to be purchased separately.

3 The VoxBoox System

VoxBoox is a system developed by us to enable HTML-coded digital books to be made accessible to visually impaired individuals on the fly. The book can be navigated aurally (i.e., using audio and voice) using phones, PDA, mobile and even computers. With the VoxBoox system, blind as well as sighted individuals can easily navigate digitally published books via voice and audio. Because the system can be accessed over the phone, the system can be used to aurally browse books even at times when a computer is not available or cannot be operated (such as while driving).

3.1 Navigation of HTML and VoiceXML Documents

As mentioned earlier, we rely on VoiceXML [6,10] to make digitally published books accessible. VoiceXML is a W3C (World Wide Web Consortium) standard markup language for marking up documents that are to be played using audio and that receive input via Voice. VoiceXML is the audio/voice analog of HTML,

the mark-up language used for coding the documents on the visual Web. Just as there are visual browsers for visual reading of HTML documents, there are voice browsers, which interpret VoiceXML files for browsing with audio/voice. However, before we discuss how content can be made available in audio/voice via VoiceXML, we need to consider the following issues.

First, there are billions of HTML coded web pages available today on the visual Web. To make all these pages also available in VoiceXML would not only be expensive but also be very cumbersome, if proper methodology is not used. Indeed, companies have designed voice portals from scratch (such as those by Tell Me Studio; dial 1-800-TELLME to get a flavor) to make certain type of information available in audio/voice (stock quotes, directions, sports scores, etc). Voice portals have not proliferated due to the considerable expense involved in developing aurally navigable pages from scratch. A better (inexpensive) approach, that we have championed, is to automatically translate HTML coded pages into VoiceXML coded pages [8]. Following this path, the current Web infrastructure can be leveraged to make, at least theoretically, the entire Web content accessible [7] to visually impaired individuals.

Second, aural browsing is inherently sequential. This is in contrast to visual browsing where many things can be quickly scanned in one visual glance. Thus, it is of utmost importance to provide extensive navigational control (such as skipping text, moving backwards and forwards) to the user during aural browsing. Along with the inherent limitation of sequentiality, there are other limitations that are present in VoiceXML itself which significantly curtail users' navigational freedom during aural browsing. VoiceXML is a markup language, which has plain text included within "form" tags. A "goto" tag provides navigation from one form to another. Unfortunately, this kind of design leaves the control of how a VoiceXML page is navigated completely in the hands of the writer of the Web page [5]. The user has absolutely no say, apart from providing responses to questions asked during navigation. Given that the author of a VoiceXML page cannot guess all possible ways in which a user might interact with a page, the approach in which the author of the page tries to guess all possible navigation scenarios in advance does not work either. In order to avoid serial reading, which could be boring, as well as giving more flexibility to the listener, adequate navigation controls should be provided to the user. One-way to solve this problem is to enhance the page with additional control facilities during the translation of the document so that the user has a better browsing experience. This is the approach we take [9]. The VoiceXML page is enhanced so that voice commands such as skip (move to next form), back (move control to previous form), start (go to beginning of document), end (go to end of document), repeat (repeat the current form from the beginning) and pause (suspend reading until user says resume) will work during browsing. Additionally, to permit free form navigation, the listener can place speech bookmarks (we term them voice anchors) on various forms (paragraphs). Then by uttering these bookmarks later, users can move to an arbitrary paragraph in the document at their will. The standard

visual web allows user to bookmark certain URLs, but it does not allow them to bookmark a specific paragraph within the document (this can only be done by the page author using HTML anchors). We also allow users to bookmark the whole VoiceXML page, so that they can navigate back and forth between different pages just by uttering these bookmarks. Finally, keyword based search is also permitted: users can utter a keyword and navigation will jump to the form that contains that keyword. Note that in all cases, the user may have to spell the word the first time they speak it while placing anchors or bookmarks, or during keyword based search.

Note that the approach we take satisfies all the criteria under which we wanted to design our system:

- Because contents are obtained by translating HTML pages, existing infrastructure is leveraged;
- Because we obtain content by an automated translation, no effort is required to make a newly published book accessible. As soon as it is available digitally in HTML, it becomes available in VoiceXML as well.
- Because we rely on HTML and VoiceXML, widely accepted standards for the Web, no new language or convention needs to be learned
- Because we enhance the VoiceXML pages with additional control and voice anchors, listeners have complete control over their browsing experience.

3.2 Usage Scenario for the VoxBoox System

To understand the VoxBoox system, consider an example scenario. Let us say the user wants to read the book titled "Oliver Twist." The user will dial a toll free number. which will connect him/her to a Voicebrowser (a Voice browser is the VoiceXML counterpart of the visual HTML browser). The first page encountered by the user is an index page, which seeks input from the user. The user will speak the title of the book (Oliver Twist). Because of the limitations of general speech recognition, however, the inputs may have to be spelled the first time. This is very similar to retrieving books over the visual Web, except that the input is via voice and output is via audio. This query is transformed into a search over the regular Web, from where the requested book (published in HTML) is retrieved. The HTML coded text for the book is then automatically translated into the VoiceXML format by our system, which is then played to the user who can listen to the book. During translation, the generated VoiceXML page is automatically augmented with features to allow dynamic navigation of its contents through speech commands and voice anchors. Speech commands allow the user to move forward and backward, skip and repeat text, etc. They also allow the user to perform a keyword based search. Additionally, users can place speech labels as bookmarks (called voice anchors) on passages and return to them later. Using Voice anchors, a listener can mark important passages, and move back and forth between different passages, much like flipping pages of a physical book.

4 VoxBoox System Architecture

Next we present the architecture of the VoxBoox system. VoxBoox converts an HTML-published into VoiceXML and further enhances it to incorporate voice-based navigation for the user. This is diagrammatically illustrated in figure 1.

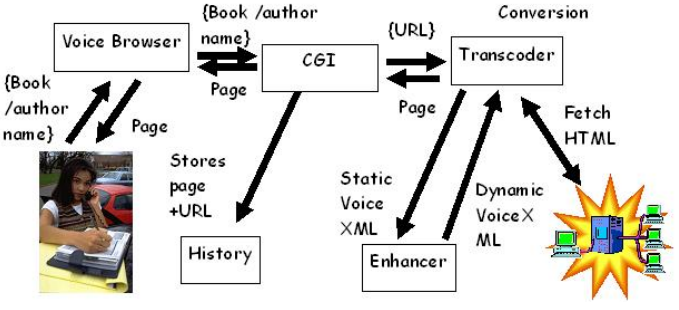

Fig. 1. Architecture of Interactive Talking Books

Before going into further details, let us first discuss the flow of control in the VoxBoox system. When the user dials a toll free number, he/she is connected to the Voice browser and the menu page is played (the names of the books available is read out one by one; the user can of course aurally navigate this menu page as well). The user utters the name of the book they want to access (there are other options available as well, such as searching by author, etc.); the browser then calls a CGI (common gateway interface) executable to handle the request. A URL is generated on the fly and is passed on to the transcoder (the module that performs the translation of HTML to VoiceXML), which fetches the HTML Web page of the corresponding book from the Web server (we use the Web server at onlinebooks.library.upenn.edu). The transcoder then converts the HTML page to VoiceXML and sends it to the enhancer (the enhancer adds control support for skip/pause/etc. as well as voice anchors to the VoiceXML page), which after enhancing sends it back to the transcoder, which sends it back to the CGI. The CGI sends the page to the browser and saves it in the history for further reference. The browser then plays the page to the user who can now navigate it aurally.

The structural design of VoxBoox can be broken down into the Voice browser, CGI (Common Gateway Interface), the transcoder, web server and enhancer. Figure 1 shows the pictorial representation of the interactions between these components. The Voice browser and the web server are independent plug-ins, which have already been developed. Voice browsers have been developed by companies such as IBM, Motorola, Tell Me Studios, etc. Our work thus mainly focuses on developing the CGI code, transcoder and the enhancer modules. To handle different type of user requests, different CGI modules have been developed. Whenever a new page is retrieved, a CGI executable is called which acts

as an interface between the transcoder and the Voice browser. Whenever voice anchors are placed and recalled, respective CGIs are called to place and recall these anchors. The transcoder module works as a translator. It converts an HTML page to static VoiceXML on the fly [7]. Whenever a HTML page is retrieved from the server, it is translated line by line to its VoiceXML version. To allow dynamic, interactive navigation of VoiceXML pages, the VoiceXML output by the transcoder is sent to the enhancer module. The enhancer inserts code and tags to allow users to place dynamic anchors in various paragraphs and retrieve them later on the fly. It also explicitly adds tags and code to allow users to skip through the paragraphs in the document. Users can repeat sections, go to the end or beginning of the document and can pause the document whenever they want. The user can even go back and forth between pages hyperlinked to each other, just as in the visual Web [7]. Next, we summarize the design and implementation of these three components.

Transcoder: As mentioned earlier, the transcoder is responsible for translating HTML pages to VoiceXML. It is implemented using Java by our group [8]. It has 2 phases: a parsing phase and a translation phase. Once it receives HTML pages from the Web Server in phase one, it parses the HTML file to obtain a parse tree. In the second phase, the parse tree is recursively translated to a semantically equivalent VoiceXML page. The transcoder semantically transforms every HTML tag to a corresponding VoiceXML tag. For this, the HTML pages has to be well-formed and should be free of extraneous information such as Java scripts [7]. To overcome this difficulty we first pass it through a filter, which removes all the unnecessary details from the HTML pages and also makes it well formed before passing it to the transcoder.

Enhancer: The foremost advantage of visual Web is to have multi directional reading. The user can skip the irrelevant material and read only the pertinent text. To provide such facility to the aural Web, the enhancer [9] adds certain tags and code to the static VoiceXML output by the transcoder. This enhancement will allow skip, back, start, end, pause, and repeat voice commands to work, along with voice anchors and keyword based search. The enhancement process is briefly described next.

The texts inside VoiceXML document are categorized into “form” tags. Each form has a unique form id. The flow of the document is determine by the “goto next =” tags. In static VoiceXML pages, the “goto next” tag is set to point to the next form tag, hence each form is played sequentially, and therefore the content of a VoiceXML page is heard serially. The writer of the document is responsible for incorporating navigation and giving users some control over the order in which forms are read. Our system incorporates the utility to skip, repeat, back, pause, resume, end and start in a document. These utilities have been implemented by explicitly adding certain tags, by manipulating the “goto next” tag and adding calls to respective CGI executable to handle different types of requests (place anchor, recall anchor, etc.). These CGIs are really code transformers that take a VoiceXML document as input, and produce a VoiceXML document as output, with the desired feature added. Thus, as user navigate

dynamically, the VoiceXML document is constantly being regenerated, with the same textual contents but with different tags, to achieve the desired control feature.

When the VoiceXML documents are enhanced, two global variables, namely, 'currentName' and 'nextName' are added, that respectively store the id of the current form being played and the id of the next form to be played. By manipulating the values of these variable and the "goto next" tag during the regeneration process, the forms can be played in any sequence. This allows the feature of skip (skipping through paragraphs), repeat (repeating the current paragraph), start (reading the page from the start), and end (going to the end of the document). Another variable called "url" stores the URL of the current Web page in the VoiceXML document. When the user wants to go to any link on the current Web page, the URL of the current web page is stored on the stack (on the server, where the CGI is executed), hence when users says the keyword "back", the last URL on the stack is popped out and the page is retrieved. The forward keyword (the opposite of back) works in a similar manner.

Anchors or Bookmarks: In visual Web, users bookmark the pages, which they think has pertinent material. Those pages also have less important text which a user cares less about. In order to give users more control over the flow of the document and avoid impertinent text, we have introduced the concept of *voice anchors*. Anchors give the user the capability to select only those paragraphs which they want to hear, hence allowing them to have full control and have complex interactions, which they did not have earlier.

Currently Voice XML technology cannot interpret arbitrary utterance of a word from the user and convert it into text. Current Voice recognitions techniques are not powerful enough to recognize any arbitrary word spoken by an arbitrary user. It can only recognize words that are encoded in the *grammar* contained in the document in advance. In order to overcome this limitation, voice browsers support dynamic grammars such as GSL (Grammar Specification Language) or SRGS (Speech Recognition Grammar Specification). By dynamically generating the VoiceXML enhanced pages, we are able to incorporate dynamic grammar into the web page hence enabling dynamic voice recognition. Our system uses GSL grammar. GSL is a dynamic grammar, which grows over time. In GSL grammar, the user has to spell out the word character by character, but this only happens once. The next time the user needs to utter the same word, he/she does not have to spell the word again, as it is already added into the grammar. The grammar recognizes the words that it encodes. There are 26 letters and 10 digits recognized by our grammar. When a user spells a word, its characters are concatenated into a string and once the user completes the word, it is added into the grammar. After user's confirmation, it is stored permanently in the grammar. Our grammar is an offline grammar, which can grow indefinitely, can recognize words that exist in it, and can store any word possible.

Whenever a user wants to mark a paragraph, it places anchors on it. Our CGI is invoked with the values of currentName, nextName and anchor name. As mentioned earlier currentName and nextName are the global values, which gives

the unique current form id number and the next form id number of the document and anchor name is the user defined name. These values are stored in the offline memory through the CGI. Once the anchors are placed in the document, any kind of sophisticated navigation is possible. When the user wants to recall those paragraphs, another CGI is called, which regenerates the VoiceXML page with the control shifted to the paragraph that was marked by the anchor. If multiple paragraphs are marked with the same anchor name, they will be read out one by one in sequence (this feature thus allows subdocuments to be created on the fly). With Voice anchors, users can freely navigate through this enhanced document containing only the pertinent information.

5 Future Work

A prototype of the VoxBoox system is operational (for a demo, please visit the following URL http://www.utdallas.edu/∼gupta/demo.html). However, considerable work needs to be done. This work primarily involves adding features to the system so as to enrich the aural browsing experience of the listener. These features include:

- Obtaining summary of a paragraph or of the book: if a user wishes to obtain the summary of a paragraph, then they just utter the word "summary", at which point the first and the last sentences of the paragraph will be read out. Uttering "book summary" will yield the summary of all the paragraphs in the whole book.
- Storing subdocuments: As explained earlier, a user can mark multiple paragraphs with the same anchor name. When this anchor name is uttered, all marked paragraphs are read in sequence. Users can thus create hierarchies of subdocuments using this feature. At present, a subdocument created in this way cannot be stored and played again in a future session. We would like to give the user the ability to store the information about such subdocuments and retrieve them later. Such stored subdocuments can be conceivably made available to other users as well (e.g., instructor's collection of important passages of the document that the students should have access to).
- Creating an index on the fly: Most of the books come with an index, but many HTML-published books (especially novels and works of fiction) may not. For rapid navigation, we would like to allow the user to create an index by just giving a simple voice command. The index will be a VoiceXML page that will be synthesized from the VoiceXML book and played to the user. The user can then use the index to navigate through the book.
- Implementing voice-commanded scripting language [10] on top of our system to allow users to program simple navigation strategies using voice. For example, a user can mark a passage using the voice anchor "plot," another passage using voice anchor "climax", and then verbally program the navigation strategy "repeat climax after plot until stop" which will cause the passage marked "climax" to be played after the passage marked "plot" in an infinite loop, until the user utters "stop."

6 Conclusions

In this paper we presented the VoxBoox system: a system for making books published using HTML on the Web and making it accessible to visually impaired individuals via voice and audio. The advantage of our system is that it leverages existing standardized technology and Web infrastructure. It also allows dynamic, user-controlled aural navigation of the book. The advantages of our system over existing approaches (audio cassette books and digital talking books by DAISY) are the following: (i) navigation is via voice commands, rather than manual (pressing keys of the cassette player or using computer keyboard); thus, to use our system, one does not need a computer: a cell phone or a land-line phone suffices. (ii) New books can be rapidly added to the collection of accessible books, since all our system needs is for the book to be digitally available in HTML on the Web. Thus, as long as the book is published or composed digitally, and the publisher can provide an HTML markup of the book, the book can be made immediately accessible. (iii) More sophisticated aural browsing is possible, compared to audio cassettes and DAISY's digital talking books.

Acknowledgments. Authors have been partially supported by NSF, Dept. of Education and EPA. We thank Dr. Dipendra Manocha of National Association for the Blind, New Delhi, India, for suggesting us to build the VoxBoox system.

References

1. Bingham, H.: Digital Talking Books Expanded Document Type Definitions (2002) http://www.loc.gov/nls/z3986/v100/dtbook110doc.htm
2. Cookson, J. et al.: Digital Talking Books: Planning for the Future (1998) http://www.loc.gov/nls/dtb.html#activity
3. The DAISY Consortium http://www.daisy.org/about_us/default.asp
4. American National Standards Institute. Specification of the Digital Talking Book (2002) http://www.niso.org/standards/resources/Z39-86-2002.html#Strategy
5. Nichols, M., Wang, Q., Gupta, G.: A VoiceXML-based Spoken Scripting Language for Voice-based Web Navigation. In: Human Computer Interaction Conference, July 2005, Lawrence Erlbaum, Mahwah (2005)
6. McGlashan, S., et al. (eds.) Voice Extensible Mark Language (Version 2.0) http://www.w3.org/TR/VoiceXML20/
7. Gupta, G., Sunderraman, S., Nichols, M.: DAWN: Dynamic Aural Web Navigation. In: Proceedings of, International Conference on Human Computer Interaction (2005)
8. Annamalai, N., Gupta, G., Prabhakaran, B.: An Extensible Translator for translating HTML to VoiceXML. In: Proc 9th International Conference on Computers Helping People. LNCS, vol. 3118, pp. 339–346. Springer, Heidelberg (2004)
9. Reddy, H., Annamalai, N., Gupta, G.: Listener-controlled Dynamic Navigation of VoiceXML documents. In: Proc 9th International Conference on Computers Helping People. LNCS, vol. 3118, pp. 337–354. Springer, Heidelberg (2004)
10. VoiceXML Review http://www.voicexmlreview.org