Structured Audio Podcasts via Web Text-to-Speech System

Giulio Mori
IIT-CNR
via Moruzzi, 1
56124 Pisa, Italy
+39 050 315 2195
Giulio.Mori@iit.cnr.it

Maria Claudia Buzzi
IIT-CNR
via Moruzzi, 1
56124 Pisa, Italy
+39 050 315 2632
Claudia.Buzzi@iit.cnr.it

Marina Buzzi
IIT-CNR
via Moruzzi, 1
56124 Pisa, Italy
+39 050 315 2631
Marina.Buzzi@iit.cnr.it

Barbara Leporini
ISTI-CNR
via Moruzzi, 1
56124 Pisa, Italy
+39 050 315 2034
Barbara.Leporini@isti.cnr.it

ABSTRACT
Audio podcasting is increasingly present in the educational field and is especially appreciated as an ubiquitous/pervasive tool ("anywhere, anytime, at any pace") for acquiring or expanding knowledge. We designed and implemented a Web-based Text To Speech (TTS) system for automatic generation of a set of structured audio podcasts from a single text document. The system receives a document in input (doc, rtf, or txt), and in output provides a set of audio files that reflect the document's internal structure (one mp3 file for each document section), ready to be downloaded on portable mp3 players. Structured audio files are useful for everyone but are especially appreciated by blind users, who must explore content audially. Fully accessible for the blind, our system offers WAI-ARIA-based Web interfaces for easy navigation and interaction via screen reader and voice synthesizer, and produces a set of accessible audio files for Rockbox mp3 players (mp3 and talk files), allowing blind users to also listen to naturally spoken file names (instead of their spelled-out strings).

In this demo, we will show how the system works when a user interacts via screen reader and voice synthesizer, showing the interaction with both our Web-based system and with an mp3 player.

Categories and Subject Descriptors
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *Graphical user interfaces (GUI)*. K.4.2 [**Social Issues**]: Handicapped persons/special needs. K.3.1 [**Computer Uses in Education**]: Distance Learning.

General Terms
Algorithms, Design, Human Factors

Keywords
Audio Podcasting, e-Learning, blind, mp3 files, TTS, WAI-ARIA

1. INTRODUCTION
Recently introduced in distance learning, podcasting is a flexible tool that adapts to users' habits and facilitates acquisition of knowledge anywhere, anytime, at any pace [6]. Numerous studies have shown the effectiveness of podcasting: many people learn better by listening to educational material than by accessing written learning objects [1], [4], [7], [9].

WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

This way of reading/studying educational documents is different from listening to a story. When podcasting is used for educational purposes, content should be accessible and easy to navigate. A continuous reading is not appropriate for effective learning, while well-structured content facilitates exploration and internal searches. Previous studies have shown that short podcasts (max 10-15 min) are more effective for learning than a single long unit. Long podcasts may decrease attention, thus reducing comprehension [3], [9]. Establishing rules and methods for achieving this is a challenge.

Educational audio podcasts can be produced by recording live events but this requires time, costs and resources. In contrast, a cheaper alternative is to use a text-to-speech (TTS) system that converts normal language text into speech [2]. However, available text-to-audio converters (such as Text2mp3, DSpeech) can break a document down into several mp3 files based on regular time intervals (e.g., 5/10/15 minutes) or on a "break string" inserted by the user in the document. The latter possibility is interesting but must be performed manually, requiring user time and effort.

Our system extends traditional TTS capabilities to obtain well-structured audio content according to the structure of the source document (e.g. one audio file for each detected section). Structured audio content offers many advantages over sequential listening: the user can listen to (download) individual files containing the audio parts of interest, and can better orient him/herself during a quick search for specific information within a podcast, improving the effective learning process.

Podcasting is ideal for blind persons, who use audio files as an alternative accessible format for written documents. It is fundamental to be able to rapidly 1) obtain an overview of the content in order to orient themselves in the whole document 2) skip parts of the content and 3) seek out important document parts (for instance, in order to go over the lesson again). In short, structured audio improves usability compared to the flat sequential version.

2. RELATED WORKS
A preliminary system prototype is described in [6]. The novel contribution of this paper is: 1) description of the system parsing algorithm; 2) statistics of document conversion; 3) the new UIs which include user preferences and new WAI-ARIA elements (file tree, regions) [12] (The ARIA - Accessible Rich Internet Applications - suite developed by the Web Accessibility Initiative (WAI) group of the World Wide Web Consortium (W3C) defines a way to make Web content and applications more accessible to people who rely on screen readers or cannot use a mouse [12]); 4) new features for enriching audio content

structure (tones are inserted into podcasts for delivering info in bold/highlighted text). To facilitate the preparation of audio materials, several tools have been proposed that transform a text document into a spoken version using TTS technology and a voice synthesizer. The output generated can be heard immediately while the audio is being produced, or it can automatically be recorded in audio files. Typically, these audio files are in mp3 format and can be listened to on a portable mp3 player or on smartphones.

Tools like Robobraille and VozMe are examples of this conversion process. RoboBraille is an email-based service which automates translation of text documents into Braille and speech. Users submit documents (e.g., text files, Word documents, HTML pages) as email attachments. The translated results are returned to the user via email [11]. VozMe (http://vozme.com/) is an easy-to-use online service for creating mp3s from text. It only requires typing in or pasting the text and pressing a button. However, the audio content created is not structured (is a single file) and so is not very suitable for studying.

Another TTS software program, Natural Reader [8] (which converts text into speech or mp3/wav files), provides a Web interface that allows one to jump to (listen to) the previous or next section of the uploaded document. Unfortunately, due to licensing restrictions, Natural Reader users cannot redistribute generated sound files unless additional licenses are bought from the voice providers.

3. THE SYSTEM

Via the system's web interface, the user uploads a document which is then transformed into structured audio files (mp3 encoding) that can be loaded by the user onto mp3 players. Specifically, blind users can take advantage of mp3 players equipped with Rockbox firmware (http://www.rockbox.org), which offers a voice-driven user-interface. Supporting special talk files, Rockbox allows users to navigate files and directories, without having to interact with the mp3 device screen.

The submitted document is saved on a server and an algorithm parses the file, attempting to identify the document section titles, to split the document into logical parts. After that, a Text-To-Speech module converts the text of each detected section (including its title at the beginning) into an audio stream; all the streams are sequentially saved on the server disk in a (non-compressed) wave file format. Generated podcasts are organized in a folder having the document's title as name. To decrease audio file sizes, next the wave files are encoded in mp3 format. Last, for Rockbox-mp3-player users, talk files are generated that vocally reproduce file and folder names.

Figure 1 shows the system's architecture. A teacher who wants to provide educational material uploads a document by means of the system Web interface. A student can download one, several or all generated mp3 files (along with associated talk files, if Rockbox-firmaware is used) and load them on his/her player.

4. PARSING ALGORITHM

Since the main goal of our system is to "segment" the document content in order to generate several podcasts, we mainly use an approach based on detection of section titles. To this end, we proposed a set of possible heuristics to apply in order to split the content into several sections.

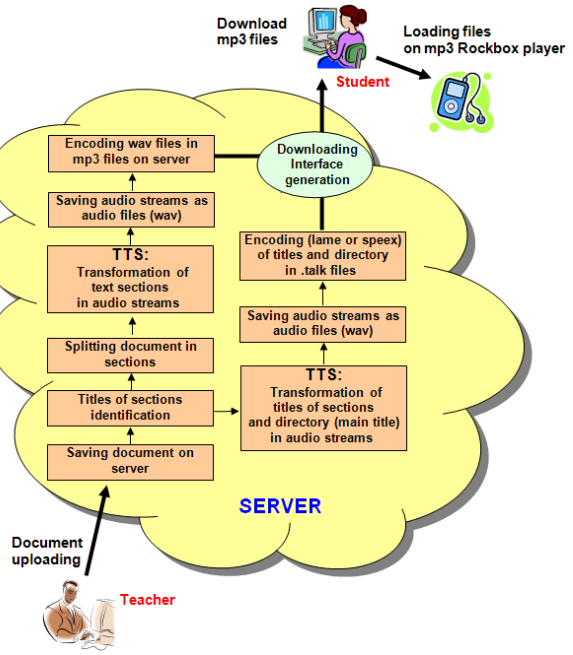

Fig. 1. System architecture

The algorithm analyzes document content in order to identify potential section titles. At the moment the algorithm is limited to MSWord, rtf and plain text document formats. MS Word/rtf documents can contain useful information characterizing the content, such as formatting styles, which can be exploited to detect the potential titles. This cannot be applied for text files, which only need a character-based analysis. In this perspective, when possible, the heuristics are applied in sequence: for MSWord/rtf documents MSWord-heuristics are first carried out and then the more general text-heuristics are considered.

In short, in order to split the content, the heuristics algorithm executes a sequence of steps. The rules are applied in a hierarchical process for refinements: at each step, the algorithm generates a list of potential title candidates that are filtered by the following steps. The final section titles extracted by the algorithm are used to split the document into sections. The main Text-heuristics algorithm can be summarized as follows:

Step 1: Tables of contents. The heuristic attempts to identify in the document the presence of a table of contents (toc). It searches for specific words such as "table of contents", "summary", etc., or for consecutive lines ending with numbers (could be page numbers). If the table of contents is found, its items are searched in the document and inserted in the candidate list.

Step 2: Line length. If a toc is not present, a candidate title is detected by counting the number of words contained in a line with specific properties: (1) the line is separated from the rest of text, and (2) the line starts with a number or with a capitalized letter or is composed of only capitalized letters, or there is a word such as 'Section', 'Chapter', possibly followed by a number, etc. A threshold of a maximum number of words is defined (i.e. 8). The next action consists of a syntactic analysis to check that the proposed line does not start, contain or end with certain characters (e.g. ":" ",", etc.).

Step 3: Numbered lists. The heuristic (through regular expressions) aims to discard elements of numbered lists from the section title list.

Step 4: Bulleted lists. This step is similar to the previous one, except that the first character should be a bullet (or characters used as bullet) rather than a number.

Step 5: First podcast. This step tries to extract the header of the document to generate the first podcast. Usually this part contains the title, authors, and so on.

Step 6: Consecutive titles. If two consecutive non-numbered candidate titles (no empty consecutive lines) are found (i.e. no text is between them) they are considered as content belonging to the same podcast. Instead, numbered sections/subsections are maintained to preserve the logical order of the document section for the user.

Step 7: Podcast size. Since consecutive short sentences, textual tables or non-bulleted/numbered lists may create false positives that would fragment the document into too many podcasts, small files are merged. It is preferable to lose some sections (which is not critical because all text is meaningful), than to split it up into a lot of blocks that do not correspond to logical sections, reducing document/podcast usability.

For MSWord/rtf documents, Word and text heuristics are combined. Additional Word-heuristics are:

Step 1: Headings and Table of Content. This heuristic is aimed at searching for Heading1, Heading2, and so on, TOC, Word styles. If they are detected, the document is probably well-structured. The headings extracted are considered to be candidate titles.

Step 2: Formatted lines. If step 1 does not identify any heading, bold and font size features are used to detect potential lines associated with titles. In this case the entire line included must have the checked property (bold, capitalized letters, etc.).

Step 3. If the two previous steps do not succeed, text-heuristics are carried out.

If the algorithm is unable to find any sections, it generates a single podcast for the whole document.

An additional feature for Word-heuristic is the extraction of long tables from a document, in order to generate separated podcasts. This can be useful for the users when studying or reviewing a section/chapter. In fact, if a table is stored in a single podcast, the user can easily skip it (if too long or uninteresting) or find it very quickly.

Another interesting feature regards podcast titles. First, the section title is used as podcast filename. Secondly, when generating the audio version, each podcast starts with the title. This helps the users when skipping forwards and backwards among the podcast tracks: if the title is read, it is easy to understand the content of the podcast in order to consider whether to listen or skip it. This function is particularly useful for a document split into many podcasts or when one needs to go over a document again. Furthermore, with a rapid scan of the first seconds of each podcast (with prev/next mp3 player functions) the user can have an overview of the document contents.

Last, another important system feature is implemented to deliver relevant information. Highlighted text indicates relevant content. The system detects bold or highlighted phrases or key words (included in sections of a submitted doc/rtf file) and delivers this information to the users by playing two tones that surround the highlighted text, the first tone ascending and the second descending. This is analogous to browsing a new book/document to acquire key words and elements to improve future comprehension.

5. INTERACTING WITH THE SYSTEM

The web interface (Fig. 2) enables the user to upload a document from his/her computer on the server and to set up his/her preferences such as the type of voices and execution speed rate of podcasts. The user selects a Speech Application Programming Interface (SAPI) voice synthesizer for podcasts and talk file announcements, from a set of voices installed on the server. A better voice quality improves user experience (perception and comprehension). In fact, several features impact on speech quality including phoneme quality, pausing, rhythm, intonation, etc. Poor pronunciation and other forms of degradation significantly demand more user effort for assimilating speech and increase the likelihood of possible misunderstandings [10].

The execution rate of podcasts can be regulated (preserving pitch) by a slider bar on the upload web interface. Executing an audio podcast at a slower speed could be useful for users with learning disabilities, while faster speeds may be preferred by users familiar with a more rapid learning process. Speed reading techniques find a trade-off between reading speed and user comprehension (http://en.wikipedia.org/wiki/Speed_reading). Depending on user ability, listening to podcasts at a faster rate might provide more rapid acquisition of information. Usually humans are able to understand speech played at a higher speed than normal, with speedup rates between 1.2 and 1.8 times faster (considering 1 the normal rate) [5].

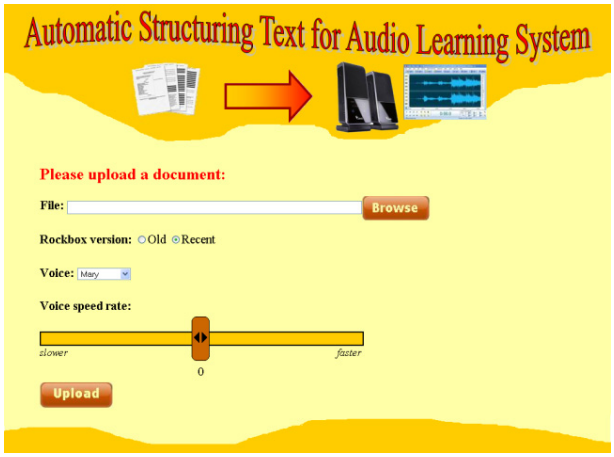

Fig. 2. Web system interface: file upload

Once the document is uploaded, parsed, processed and converted into audio podcasts (each one corresponding to a document section), it can be also downloaded link by link through the hierarchical tree (exploiting the structure of the document) on the download web page. Alternatively, the user can download all audio podcasts in two zip packages: one containing only all mp3 files and another one that also includes the talk files, ready to be loaded on a Rockbox player (Fig. 3). The original source document is downloadable via the link labeled with its title.

The Podcast list (Fig. 3) corresponds to a content table of the submitted document. This list allows the user to quickly understand the structure and topics of the document without having already read (or listened to) it. In our system, screen reader users can easily explore a podcast list on the web interface by jumping on the corresponding WAI-ARIA region [12]. In order to improve usability via screen reader (limiting the number of links), if the algorithm generates a high number of podcasts, only one link pointing to the whole list is inserted in the page.

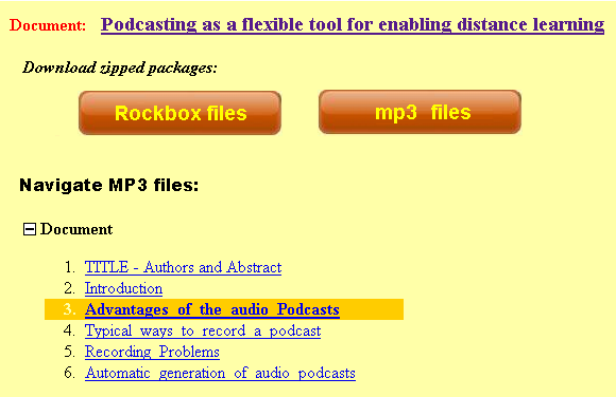

Figure 3. Web system interface: podcast download

Some complex widgets such as slider bars or tree views are usually not accessible by web browser or screen reader. In order to make the user interface more accessible, WAI-ARIA [12] suite has been applied to the interface development.

6. SYSTEM TESTING

Generated audio files (mp3 and talk type) were tested on different official Rockbox mp3 player emulators and real players (such as an Iriver H10 6Gb or an Ipod Nano 1st generation).

In the phase of system implementation, to verify and improve heuristics for document parsing, a training set of 200 documents was used, composed of 100 doc/rtf and 100 txt files. The 100 doc/rtf documents of the training set was composed thus: 50 well structured (well definite separation of sections using headings, hierarchical font size and styles for titles, etc), 40 partially structured and 10 poorly structured. The mixed composition of the training set aims to design more affordable heuristics when facing different document types. Percentage of correctly detected titles was 90.57% (for doc/rtf) and 88.60% (for txt). Next, we used another 210 documents as a test set (scientific papers, technical manuals, e-books, etc.) randomly picked by Google. This new set was composed of 110 doc/rtf and 100 txt files. The efficacy of detecting titles exactly was 83.83% (for the 110 doc/rtf documents set) and 78.44% (for the 100 txt set).

Analyzing the set of documents where section identification (partially) failed we observed that although well-structured documents (heading levels or table of contents) were correctly identified, several doc and rtf documents were partially or incompletely well-structured (no headings, no bigger font size, etc.) and practically, were analyzed by the text heuristics, which are less precise than structured documents (especially text documents that present contiguous short lines, fragmented parts, or some sort of ambiguous tables, or separation). Of 110 doc/rtf parsed documents in the test set, only 66 were well-structured, 30 were partially structured and 14 were very poorly structured.

7. DISCUSSION

Structured audio podcasts offer many advantages for all users and are greatly appreciated by the visually impaired. Short podcasts – for listening anytime, anywhere, and configurable at the user's own rhythm -- can greatly facilitate learning. Our system focuses on this research topic, providing automatic structured audio podcasts starting from text-based documents. The system's Web interface is fully accessible via screen reader as are the generated podcasts (including .talk files).

Future work includes extending the system's conversion ability to also convert other document formats (x/html, odt and pdf). We also plan user tests with blind subjects.

8. REFERENCES

[1] Aldrich, D., Bell, B., and Batzel, T. Automated Podcasting Solution Expands the Boundaries of the Classroom. in Proceedings of the 34th annual ACM SIGUCCS conference on User services, 1-4, 2006.

[2] Allen, J., Sharon Hunnicutt, M., and Klatt, D. From Text to Speech: The MITalk system. Cambridge University Press: 1987. ISBN 0521306418.

[3] Cebeci, Z., and Tekdal, M. Using Podcast as Audio Learning Objects. Interd. Journal of Knowledge and Learning Objects. 2, 2006.

[4] Deibel, K. Course experiences of computing students with disabilities: four case studies. in Proceedings of the 39th SIGCSE technical symposium on Computer science education, 454-458, 2008.

[5] Lauer, T., and Wolfgang, H. Audio-based Methods for Navigating and Browsing Educational Multimedia Documents. in Proceedings of the inter. workshop on Ed. Mult. and mult. ed. Poster and demo session, 123-124, 2007. DOI= http://portal.acm.org/citation.cfm?id=1290166.

[6] Leporini B., Buzzi M. C., Buzzi M., and Mori G. Automatically Structuring Text for Audio Learning. in the Proc. of HCI International 2009. in Springer LNCS. Vol. 5616/2009 Universal Access in Human-Computer Interaction. Applications and Services, 73-82, 2009.

[7] Mermelstein, B., and Tal E.: Using Cellular Phones in Higher Education, in Wireless and Mobile Technologies in Education (WMTE 2005), 2005.

[8] NaturalSoft Text-to-Speech, http://www.naturalreaders.com/index.htm

[9] Ormond, P. R. Podcasting enhances learning. Journal of Computing Science in Learning 24, 1 (2008), pp. 232-238.

[10] Pitt, I., and Edwards, A. Design of Speech Based Devices. A Practical Guide. Springer, London, UK, 2003.

[11] The RoboBraille Consortium. The RoboBraille email service, http://www.robobraille.org

[12] W3C. WAI-ARIA Overview, http://www.w3.org/WAI/intro/aria.php