BEP: a practical bilingual speech synthesis system for Japanese

T. Watanabe[a], M. Sakamoto, and M. Kiriake

[a] Department of Information Science, Shonan Institute of Technology,
1-1-25 Tsujido-NishiKaigan, Fujisawa, Kanagawa, 251-8511, JAPAN
<takayuki@la.shonan-it.ac.jp>, Tel +81-466-30-0217, Fax +81-466-34-1096

All authors are members of ARGV (Accessibility Research Group for the Visually-Impaired)

The Bilingual Emacspeak Platform (BEP) is a bilingual text-to-speech system for Japanese visually impaired Emacs users and runs under both Windows and Linux. It is implemented as an extension of Raman's Emacspeak, a self-voicing GNU Emacs platform with dedicated auditory user interface. Based on the experiences of existing Japanese screen-readers, BEP is equipped with functions necessary for Japanese speech system. BEP for Windows uses various types of Japanese and English voices to represent bilingual information naturally and unambiguously. In this paper, the authors firstly review the features of Emacspeak and BEP and then show the requirements of a bilingual speech synthesis system for efficient and practical use.

Keywords: bilingual speech system, human-computer interaction, auditory user interface, text-to-speech, visually impaired, assistive technology, Japanese

1. Introduction

Personal computers enabled people to use a computer for their personal use. Personal computers also changed the lives of people with disabilities because this new technology was used as an assistive technology. With the aid of a personal computer, people with disabilities can work, study, live, or play independently. For example, visually impaired (low-vision and blind) people can use a computer to read information with a screen-reader, which is software that reads text on a display through a speech synthesizer.

As technology grew more sophisticated, new technologies that made it easier for non-disabled people to use computers often created barriers for people with disabilities. For example, Windows OS popularized personal computers to general people; however, visually impaired people cannot make full use of its GUI (Graphical User Interface) because it is difficult for a screen-reader to transfer rich contents of 2-dimentionally arranged information, e.g. icons and menus, on the graphic display to simple 1-dimentional speech output. Thus, we must develop a new effective mechanism and user interface to output a lot of information in modern computers using synthesized voice. Such speech output also enhances mobile access of personal network appliances for general users and enables multi-modal user interface.

2. Emacspeak

In order to overcome the mismatching of Windows' GUI and screen-readers, T.V. Raman developed a new speech system called Emacspeak[1]. Emacspeak is an Emacs subsystem that provides complete speech access to GNU Emacs, a structure-

[1] Emacspeak is distributed as open-source at http://emacspeak.sourceforge.net/

sensitive, fully customizable editor. Emacspeak knows context-specific information of what it is speaking and can efficiently transfer the information to the sense of hearing with the following techniques:

1) Auditory User Interface (Raman, 1997): Raman claims that to adjust to various requirements of various circumstances of users, user interface must be separated from the system itself. In this manner, Raman added AUI, which works independently of GUI, to Emacspeak. For example, it reads a calendar with additional information such as "Monday, April 1st, 2001". Without AUI, 2-dimentional information of a calendar is lost and it is spoken as "one", which makes no sense to the visually impaired users.
2) Voice fonts: With the aid of functions of Emacs, Emacspeak can assign different voices to different information. For example, when reading a source code file, it assigns different voices to comment strings, key words, and quoted strings. Emacspeak reads a Web page with Auditory Cascading Style Sheets[2]: it uses different pitches of the same voice for reading different heading level elements. DECtalk, a high-quality text-to-speech hardware synthesizer, enabled Emacspeak to use various kinds of voices with various characteristics (e.g., pitch, stress, intonation).
3) Auditory icons: Auditory icons are short sound cues that alert the user to significant events. A user can do another task while, for example, compiling a source program until an auditory icon informs her/him the end of compilation.

[2] Auditory Cascading Style Sheets is defined in the CSS2 (http://www.w3.org/TR/REC-CSS2).

3. Bilingual Emacspeak Platform

3.1 Overview and structure

In 1999, we, the sighted and the visually impaired who love Emacs, launched the Bilingual Emacspeak Project (Watanabe, 2001) that develops Bilingual Emacspeak Platform (BEP) to make Emacspeak bilingual (Japanese and English) and to make it run under Windows as well as Linux. The current system is bilingual because Japanese users need to read English as well as Japanese at workplace and in the Internet. It runs under Windows because there are many users who use Windows. It also runs under Linux because the original Emacspeak works only under UNIX and UNIX is expected to be an alternative accessible OS for the handicapped (Watanabe, 2000b).

As shown in Figure 1 of the next page, the current system consists of two parts, an Emacs Lisp program package that acts as AUI and a software speech server. Emacs Lisp program represents the bilingual version of Emacspeak. The speech server controls English and Japanese text-to-speech engines. It accepts commands in the DECtalk format and text in the Japanese Shift_JIS character-encoding scheme. Windows' speech server, which is a modification of G. Bishop's one[3], uses Microsoft's American English text-to-speech engine and a commercial Japanese text-to-speech engine that are conformable to Microsoft Speech API 4 (Watanabe, 2000a). Linux's speech server uses a Japanese text-to-speech engine developed by us based on a commercial software development kit (Watanabe, 2000b) and is not bilingual. It has not had bilingual capability yet but it speaks English with a Japanese text-to-speech engine using the Japanese-English dictionary made by J. Ishikawa (Ishikawa) with Japanese accent.

[3] This program was written by G. Bishop for personal use and given to us as open source.

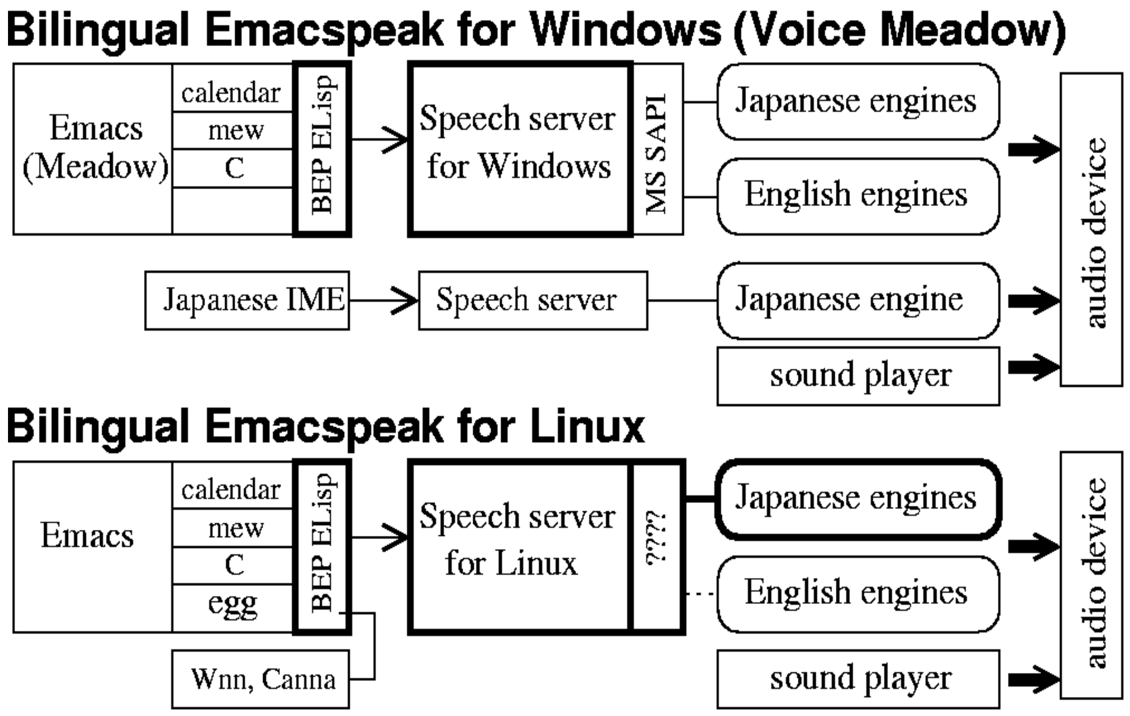

Figure 1. The structure of Bilingual Emacspeak. Bold boxes are modules developed by the current project. Meadow is a Windows' porting of Emacs 20.x, Mew is a Japanese Email package, and Wnn and Canna are Kana-Kanji translation servers.

3.2 Language identification and assignment of engine's language

The current version of BEP does not care about the language of the text in the Emacs Lisp level. It identifies the language in the speech server. BEP for Windows uses a Japanese text-to-speech engine if the text to be read, which is transmitted from Emacs, is encoded in multi-byte codes. As discussed later, language identification in a speech server is not good for effective bilingual speech system; therefore, this language identification will be carried out in the Emacs Lisp level in the near future to make robust and flexible discrimination of the language in multilingual information.

3.3 Auditory representations of Japanese

Usually, pronunciation of each Japanese word is determined by the Japanese text-to-speech engine. In some cases, however, pronunciation should be specified by the speech system to reflect user's demands or for syntactic and contextual reasons. Emacspeak has punctuation modes that determine how to pronounce some characters such as punctuation marks. For example, every mark should be read when writing a source code, whereas punctuation marks should not be read when reading a letter. As a text-speech-engine does not know what it is speaking, i.e. contextual information, a speech server must switch the punctuation mode according to contextual information or user's demands.

Japanese language is written in Kanji (Chinese characters), Hiragana, Katakana, and other multi-byte characters. Multiple Kanji characters have the same pronunciation. Hiragana and Katakana correspond one-to-one and each pair has the same pronunciation. As a user cannot identify these characters only by hearing the pronunciation, he/she needs additional information to distinguish them. Japanese screen-readers have developed many methods to unambiguously represent

Japanese to the ears (Saito, 1995). BEP uses two mechanisms to give this additional information to the user. One is audio formatting, or use of various voice-fonts[4]. For example, Hiragana and Katakana will be read with different voices or different pitch of the same voice. The other is an explanatory reading of Kanji, which is the same function as ordinary Japanese screen-readers have. The current system has 3 levels of explanation for each Kanji: 1) a very brief explanation used for reading a character according to a cursor movement, 2) a short explanation for selecting the correct Kanji in a Japanese input method, and 3) a long explanation used for other cases. The dictionary of these explanatory readings is based on the one used in J. Ishikawa's screen-reader. The first-level explanation is used when a user want to quickly examine the information character by character. In this case, pronunciation of each character is defined in a dictionary, i.e. raw and simple pronunciation is used to quickly identify each character. The second level is used when a user want to examine a character in detail. In this case, the character is explained with some additional information to clearly distinguish it from other characters having similar pronunciation. The third level will be used to explain the detailed information of that character.

3.4 BEP as a practical and efficient speech synthesis system

BEP is still under development but the alpha version of executables and source codes have been distributed on the Web (http://www.argv.org/bep/). Support is carried out using a mailing list and the web site. A small number of users are using BEP in their everyday life. The number of current users is not large because to use BEP the user must be familiar with Emacs and it takes some time to learn how to use it and also because the current version is not robust. However, BEP has many functions that are not offered by the existing Japanese screen-readers (Watanabe, 2001).

[4] This function has not been implemented yet.

4. Requirements of a speech synthesis system for practical and bilingual use

4.1 The characteristics of speech output

At first, let us summarize the characteristics of speech output:

1) Speech output lacks birds-eye view. As a speech output is 1-dimensional sequential audio stream, a user cannot grasp the whole structure with speech output. A user cannot obtain the necessary information until the audio stream reaches the point.
2) A user cannot easily hear again the missed information because it is difficult to rewind the speech stream.
3) It takes longer time to hear information by speech than to read by eyes.

4.2 Requirements of an efficient speech synthesis system

Visually impaired users have proved that a human can use a computer only with speech output. They use speech output because they cannot use sight. Sighted users can use sight, and speech output is used as multi-modal and alternative interface. In this sense sighted users are not well accustomed to speech output and they feel difficulties in using a speech system.

Characteristics of speech output and experiences in developing and supporting BEP clearly show the requirements of a practical speech system: the most important requirements are not to speak text naturally or expressively but to speak essential information as fast as possible in a manner best suited to the sense of hearing. In other words,

1) To speak information twice or three times faster than the normal speed. Speaking text in double speed shortens

the time to hear the information in half.

2) To speak the necessary information only. Otherwise it takes long time before obtaining the necessary information.
3) To use AUI instead of obtaining information displayed on a screen.
4) To use light text-to-speech engines with clear voices. Details of the requirements are discussed later in this section.
5) To use capable speech server. Details of the requirements are discussed later in this section.

4.3 Requirements of text-to-speech engines

Requirements of text-to-speech engines, which are basic components for a speech synthesis system, are as follows:

1) A quick engine: Good response is indispensable for a practical speech system.
2) A clear voice that acts as a code: The most important feature required to the voice is to correctly give the information to the user at a high speed. Thus a text-to-speech engine must have a voice that can clearly distinguish each character. Natural voices tend to lose this kind of clarity at faster speech rate.
3) Multilingual (American, European, Asian, etc) engines: Te xt-to-speech engines for many languages are available under Windows; however, there is no practical Japanese engine under Linux.
4) Unified interface: Windows has an established speech interface; however, there is no such interface for Linux. The lack of standard interface in Linux prevents us from simultaneously handling engines of different languages. A unified interface that considers characteristics of various languages is needed.
5) One letter speech synthesis: The current text-to-speech engines are good at speaking strings; however, they are not good at pronouncing a single letter because of their synthesis mechanism. One letter pronunciation is needed when a user want to examine what is written.
6) Phonemic speaking: In case a user want to read a character one by one, the fixed pronunciation, which should be predefined in a dictionary[5] of a speech system, can be assigned to every character. In this case, phonemic speaking, i.e. pronunciation represented in a string of phonemes is directly given to a speech server, reduces the analyzing time of a text-to-speech engine especially in Japanese. Phonemic speaking reduces the task of a text-to-speech engine because it can create PCM data directly from phonemic data.
7) Open source engines: Text-to-speech engines are basic components of AUI as fonts are basic ones of GUI. Thus open-source, or at least free, text-to-speech engines are needed.
8) Highly capable engine: A text-to-speech engine must have the ability to read at least as three times fast as the ordinary speech rate. It must have the ability to change characteristics such as speed, pitch, stress, and intonation. To use voice fonts, several kinds of voices must be available for each sex of each language.

4.4 Requirements of a bilingual speech system

Experiences of BEP show that a bilingual speech system should use the user's native language (for us, Japanese) in most cases because users are not bilingual, i.e. they cannot hear the second language (English) well. At the same time, users need the native English text-to-speech engine when

[5] This pronunciation is determined in the first level of our dictionary described in the section 3.3.

they want to hear information as English. Generally speaking, system messages, directory and file names, and an isolated English word in a Japanese context should be read in Japanese pronunciation even if they are English. Thus, a bilingual speech system should have the following functions:

1) Context-based language selection: The selection of language used to read information depends on the context and user's demands; therefore, language selection should be carried out by Emacs, i.e. Emacs Lisp level, and not by the speech server. As the current implementation of BEP automatically switches Japanese and English engines according to the language of the text in a speech server, it should be modified in the near future.
2) Native pronunciation and Japanese-English pronunciation: The ordinary Japanese are accustomed to hear Japanese-English pronunciation, or so called Katakana English, than native pronunciation. For them, English should be pronounced in this Japanese-English. As the current Japanese text-to-speech engines are not good at pronouncing native English[6], a bilingual speech system should have an `English to Japanese-English' dictionary, which gives Japanese-English pronunciation consisting of Japanese phonemes, to clearly present English to Japanese users. Actually, J. Ishikawa's Japanese screen-reader for DOS (Ishikawa) has developed such dictionary and reads English in Japanese-English well. BEP for Linux uses this dictionary. It should be noted that Japanese-English is not appropriate for all cases. Users want to hear Japanese-English when they think the information is a part of Japanese. Even if such information is written according to English grammar and written in ISO-8859-1, it should be treated as a part of Japanese and pronounced in Japanese-English. However, when users want to hear English information, it should be pronounced in native English or users cannot recognize the meaning well.
3) Variable ratio of speech rate: The speech rate for Japanese and English should be changed separately because hearing English is a difficult task for ordinary Japanese users.

[6] In general, Japanese text-to-speech engines are good at pronouncing English neither in native English nor in Japanese-English. These engines cannot correctly pronounce Japanese written in Roman letters. It seems that they do not have English pronunciation dictionaries and proper algorithms to determine appropriate pronunciation in these cases.

4.5 Importance of meta-data

Emacs, an intelligent text editor, can determine the characteristics and semantics of information to some degree; however, without showing meta-data by some other methods, it is difficult to represent information suitably to various users. For example, sometimes it is impossible to determine the language and the character-encoding scheme only by analyzing the character codes of the text. Emacspeak can use HTML and Auditory Cascading Style Sheets to expressively represent data. The combination of XML and style sheets, which act as some of AUI functions, would be the best way to represent data in a way that matches the user's demands.

6. Concluding Remarks

We have developed a bilingual speech system based on Emacspeak and Japanese screen-readers. The alpha version of BEP is used by Japanese visually impaired users who want to use Emacs. In developing and supporting the system, we found important requirements of a practical speech synthesis system that have not been discussed well. They are

(1) An auditory user interface that presents the information effectively to the sense of hearing using various voice fonts and auditory icons.

(2) Highly capable text-to-speech engines that produce easy-to-hear voice at double or triple speech rate.

(3) Free and open-source text-to-speech engines of many languages for many operating systems.

(4) A standard interface for international text-to-speech engines.

(5) A mechanism to select language and pronunciation according to the user's demands.

(6) Meta-data.

Emacs itself can handle multilingual text; therefore, it would be nice to extend BEP to a multilingual speech system to meet the requirements of worldwide users, especially of Asian users[7].

The current system for Windows requires large CPU power. Pentium II or higher and enough memory is required[8]. The required machine specification of Linux version is not so demanding. It can run on a MMX Pentium computer. As the current system is still under development, both versions are not stable enough to be used to practical use. We are now struggling to improve the system, especially the speech servers. Recently, some PDAs that use Linux are on the market. When running BEP on such a PDA that does not have enough machine power, hardware speech synthesizer would be a better choice to ensure good response.

[7] In fact, there are few useful screen-readers for Windows in Korea and China.

[8] BEP for Windows can somehow run on a computer with 200MHz MMX Pentium and 64MB memory but it becomes more stable with more CPU power and more memory.

Acknowledgements

The authors are grateful to all members, especially Mr. K. Inoue, of Bilingual Emacspeak Project for their contribution to the current system[9]. This work was supported in part by Research for the Future Program of JSPS under the Project ``Integrated Network Architecture for Advanced Multimedia Application Systems" (JSPS-RFTF97R16301).

References

Raman, T.V. (1997). Auditory user interfaces -- toward the speaking computer --. Kluwer Academic Publishers.

Watanabe, T., Inoue, K., Sakamoto, M., Kiriake, H., Honda, H., Nishimoto, T., & Kamae. T. (2001). Bilingual Emacspeak Platform -- A universal speech interface with GNU Emacs --. Proceedings of the 1st International Conference in Universal Access in Human-Computer Interaction 2001.

Watanabe, T., Inoue, K., Sakamoto, M., Honda, H., & Kamae, T. (2000a). Bilingual Emacspeak for Windows -- a self-speaking bilingual Emacs --. Tech. Report of IEICE SP2000, 100(256), 29-36, (in Japanese).

Watanabe, T., Inoue, K., Sakamoto, M., Kiriake, M., Shirafuji, H., Honda, H., Nishimoto, T., & Kamae. T. (2000b). Bilingual Emacspeak and accessibility of Linux for the visually impaired. Proceedings of Linux Conference 2000 Fall, 246-255, (in Japanese).

Ishikawa, J. Details of his speech system are described in http:// fuji.u-shizuoka-ken.ac.jp/~ishikawa/

[9] The authors think the most important point of BEP is not technical features but that the current system is developed by the cooperation of sighted and visually impaired Emacs users to make a new speech system that does not suffer from the mismatching of Windows' GUI.

devpgm.htm (in Japanese). Some dictionaries used in BEP are taken from his speech system.

Saito, M. (1995). Software Production for the Visually Impaired. Journal of IPSJ, 36(12), 1116-1121, (in Japanese).