A Flexible VXML Interpreter for Non-Visual Web Access

Yevgen Borodin
Department of Computer Science
Stony Brook University
Stony Brook, NY 11794, USA
borodin@cs.sunysb.edu

ABSTRACT

VoiceXML (VXML) is a W3C's standard for specifying interactive dialogs. It finds multiple uses in various Web applications. VXML can also be used in non-visual Web browsing. There is no suitable, complete, open-source, flexible VXML interpreter to process VXML dialogs. My project is focusing on developing a VXML interpreter, VXMLSurf, that will be fully compliant with VXML 2.0 specifications and geared toward accessing Web content. VXMLSurf implements a number of extended features that provide blind users with more control over interactive browsing dialogs. VXMLSurf is a part of the HearSay project for developing a non-visual Web browser. The goal of the project is to make the Web more accessible for blind people.

Categories and Subject Descriptors

D.3.4 [**Programming Languages**]: Processors—*Interpreters*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*natural language, Voice I/O*

General Terms

Design, Languages, Human Factors, Standardization

Keywords

Web Navigation, VoiceXML, VXML, VXMLSurf, Interpreter, Screen-Reader, Voice Browsing, User Interface, non-Visual

1. INTRODUCTION

As the Web matures, more and more services are provided online. Web standards emerge to facilitate the development of new tools and services, VoiceXML 2.0 [14] being one of such standards. VoiceXML (VXML) is a W3C's [15] standard XML format for specifying interactive voice dialogues between a human and a computer. VXML employs speech recognition grammars, embedded JavaScript, event handlers, etc. It is a powerful language that can describe complex audio dialogs with multiple applications.

VXML is widely used to describe menus in telephone systems. It can be used to disseminate information to public through phones or computer terminals. For example, [7] describes interactive bus-information dialog system in Kyoto City, Japan. GEMINI project uses VoiceXML dialogs in a multilingual interactive natural language interface [3]. VXML can also be used in computer games [6].

Another application of VXML is in non-visual voice browsing, where Web page content is converted to VoiceXML dialogs to help blind people browse the Web. Examples of VoiceXML dialogs for Web browsing are given in [9, 11]. Popular screen-readers, such as JAWS [4] and IBM's Home Page Reader [13], speak out the content of Web pages in a straightforward manner and do not require complicated dialog management systems. However, they provide very little interactivity or control, and their entire dialog management system has to be modified to accommodate any changes.

On the other hand, systems using VoceXML dialogs are more flexible, because every VXML dialog is a dialog manager in itself. HearSay [11] is one such system that employs the flexibility of VoiceXML dialogs. The system endeavors to dynamically create multiple layers of dialogs, which, in time, will allow mixed-initiative dialog-based interaction between a user and a Web browser. VoiceXML allows programming interactivity within dialogs.

A VXML interpreter is required to process VoiceXML dialogs. Unfortunately, there are no completed, open-source VXML interpreters. The existing open-source interpreters are supporting only a small subset of VXML, or have other drawbacks. Commercial interpreters are not extensible, even though some of them allow free evaluations for research purposes, like OptimTalk [10]. Besides, most VoiceXML interpreters are geared toward telephony applications [1].

The goal of my project [1] is to develop a open-source, modular, multi-platform, extensible VXML interpreter that will fully comply with VoiceXML 2.0 specifications. The interpreter is geared toward non-visual Web browsing to provide blind users with more control over dialog flow as they access the Web. The additional features will go beyond current VXML specifications. The results obtained in the course of the project will be submitted to W3C's for review to potentially contribute to future VoiceXML 3.0 recommendations.

2. THE INTERPRETER

The HearSay project is developed in collaboration with Helen Keller School for the Blind (HKSB) at Hempstead, NY. The design of the system is guided by people with visual disabilities who are teachers at HKSB. The ideas have been obtained and clarified in meetings with instructors and students of the school.

ASSETS'06, October 22–25, 2006, Portland, Oregon, USA.
ACM 1-59593-290-9/06/0010.

[1] This project is a part of the HearSay project supported by NSF grants CCR-0311512 and IIS-0534419.

My VoiceXML interpreter, VXMLSurfer, is already used as part of HearSay system. It is supporting a large subset of VXML tags, including a full implementation of event and variable spaces. However, the interpreter is still in the development stage. For complete listing of the features that have to be implemented please refer to VoiceXML 2.0 specifications [14]. In this section, I will concentrate on the extended capabilities and the architecture of the interpreter.

VXMLSurfer has been extended with additional features that control voice properties, flow of dialogs, and advanced event-handling. The ideas of key controls were borrowed from the state-of-the-art screen reader, JAWS [4]. Additional shortcuts were or will be implemented per request of end-users. Advanced features can be turned on/off for complete backward compatibility with VXML 2.0 specifications.

Extended Features. To improve user experience, the interpreter implements advanced voice controls that allow the use of shortcuts, such as pause, resume, restart the utterance, as well as change the pitch, voice, rate, and intensity of the speech. Additional shortcuts allow skipping the content at various scopes, e.g. sentence, paragraph. Key strokes are treated as events, and have predefined event handlers. Event handling is implemented in a way that will allow to define new events and override default events, making the interpreter very flexible.

VoiceXML can be used to specify both the dialog and the dialog manager and is powerful enough to implement most of the navigational controls. However, while that works well for smaller dialogs, it was experimentally determined that it often leads to significant increase of the VoiceXML dialog size. Implementing some of the navigational controls within the interpreter will reduce the complexity of creating VoiceXML dialogs and programming the dialog manager.

The features that have yet to be implemented include searching, moving in any direction in the dialog, and switching between different views of the same dialog. This will require saving various states of the interpreter and will be similar to dialog debugging.

Implementation. To be platform independent, VXMLSurfer is written entirely in Java. It has three separate threads: for input, output, and VoiceXML processing. The interpreter is designed to be modular and to be able to use various voice-recognition and text-to-speech engines through Java Speech API (JSAPI) [5]. Text-to-speech conversion is done via FreeTTS v1.2.1 [2], a freely available speech engine. In its current configuration, the interpreter allows only keyboard input. However, CMU Sphinx [12] voice recognition engine will be integrated in the interpreter as soon as speech-grammar support is added. VXMLSurfer will provide its users with more control through both shortcut keys and voice commands.

3. APPLICATION

VXMLSurfer is a part of the HearSay system. The HearSay infrastructure has superior browsing facilities and can perform simple structural and semantic analysis of Web page content. The system uses Mozilla Web browser [8] to obtain a frame tree representation of Web pages, which are then further analyzed, grouped, and partitioned.

Subsequently, a dialog generator uses the information of the frame tree and a number of VoiceXML templates to generate multiple layers of dialogs, which are then processed by the interpreter. The three layers of Web browsing dialogs include basic screen-reading, BFS/DFS navigation, and domain-specific dialogs. The generator is logically separated from the interpreter, so that they are easier to maintain and, if necessary, can be replaced individually. The exhaustive description of the Dialog Generator and other HearSay modules is not in scope of this paper. For more information refer to [11].

VXMLSurfer already supports the first layer of HearSay dialog interaction, supporting basic screen-reading and extended speech controls. Processing of the second and the third dialog layers will require implementation of more advanced dialog navigation controls. And, finally, VXMLSurfer has to be extended to support adaptive dialogs. A series of progressive evaluations of the HearSay system will be performed by the students of the Helen Keller School for the Blind as the system takes shape.

4. CONCLUSION

In this paper I presented a project directed toward the development of a flexible multi-platform VoiceXML interpreter, VXMLSurfer. The interpreter is a part of the ongoing HearSay project [11] for non-visual Web browsing. As the design of the Hearsay project is guided by blind users, the VoiceXML interpreter will contribute toward making the Web more friendly and accessible for blind people. The project will also provide the community of researchers with an extensible open-source tool for VoiceXML dialog exploration geared toward non-visual Web browsing. This work has the potential to have a positive effect on the W3C's VoiceXML 3.0 recommendations.

5. REFERENCES

[1] `http://cafe.bevocal.com`.

[2] `http://freetts.sourceforge.net`.

[3] S. Hamerich, V. Schubert, V. Schless, R. Crdoba, J. Pardo, L. d'Haro, B. Kladis, O. Kocsis, and S. Igel. Semi-automatic generation of dialogue applications in the gemini project, 2004.

[4] `http://www.freedomscientific.com`.

[5] `http://java.sun.com/products/java-media/speech`.

[6] `http://www.jsmart.com`.

[7] K. Komatani, F. Adachi, S. Ueno, T. Kawahara, and H. G. Okuno. Flexible spoken dialogue system based on user models and dynamic generation of voicexml scripts.

[8] `http://www.mozilla.com/firefox`.

[9] `http://www.internetspeech.com`.

[10] `http://www.optimtalk.cz`.

[11] I. Ramakrishnan, A. Stent, and G. Yang. Hearsay: Enabling audio browsing on hypertext content. In *Intl. World Wide Web Conf. (WWW)*, 2004.

[12] `http://cmusphinx.sourceforge.net`.

[13] H. Takagi, C. Asakawa, K. Fukuda, and J. Maeda. Site-wide annotation: Reconstructing existing pages to be accessible. In *ACM Intl. Conf. on Assistive Technologies (ASSETS)*, 2002.

[14] `http://www.voicexml.org`.

[15] `http://www.w3.org`.