VOICEXML INTERPRETERS

Ing. Stanislav Ondáš
supervisor: doc. Ing. Jozef Juhár, CSc.

Department of Electronics and Multimedia Communications,
Technical University of Košice, Park Komenského 13, 041 20 Košice,
Slovak Republic.
E-mail: Stanislav.Ondas@tuke.sk

ABSTRACT

The article deals with development and implementation of VoiceXML interpreter. Our goal is to develop an interpreter fully supporting VoiceXML v2.0. We have started from the scratch and in actual state it performs all fundamental algorithms of VoiceXML language and service functions for all VoiceXML commands, i.e. Form Interpretation Algorithm, Event Handling, Grammar Activation and Resource Fetching algorithms. It does not support full range of VoiceXML yet but it is enought for building basic voice dialogues. This interpreter is tested in two development dialogue platforms today.

1 INTRODUCTION

VoiceXML is a markup language derived from XML (eXtensible Markup Language). It is designed for creating audio dialogs that feature synthesized speech, recognition of spoken and DTMF key input, telephony and others technologies. Its major goal is to bring the advantages of Web-based development and content delivery to interactive voice response applications. It makes possible effective to build dialogues applications and services for IVR systems. For example weather and traffic information or financial services.

VoiceXML language as any XML languages needs an interpreter that interprets VoiceXML commands. This interpreter is now the ground of dialog management in interactive voice response systems. The main advantage of interpreted programming language is that its code need not be compiled. This is especially useful by dynamic generation of VoiceXML pages, when we work with often changing content.

There are several free (open source) and commercial VoiceXML interpreters. Free interpreters in most cases do not support all VoiceXML commands. On the other side commercial interpreters are often included in a complex solution of interactive voice response system. From this point of view are not useful for next development (for example for connect with speech recognizer).

2 IMPLEMENTATION OF VOICEXML INTERPRETER

VoiceXML interpreters interpret VoiceXML commands. These are not interpreted sequentially from above downward, but their executing is exactly described by VocieXML algorithms (Document loop, Form Interpretation Algotithm, ...). More of today′s VoiceXML interpreters interpret version 2.0 of VoiceXML language. Beside VoiceXML interpreter also other components are necessary to create dialog between computer and user (caller). These are for example speech recognizer and text-to-speech synthesis unit a.o.. Interpreter generates states of dialog and communicates with other components of dialog manager system.

2.1 Typical Scheme of VoiceXML applications

The structure of VoiceXML application is very similar to HTML document. The ground of VoiceXML documents are dialogs. Each dialog is delimited by the pair of form or menu elements. Each element of VoiceXML language must have beginning and termination markup. (for example <form> and </form> markups).

Within dialogs can we find input fields. They are represented by <field> element. Input fields collect information from user. After that is played the promt to the user by <prompt> element and is set up some speech grammar by <grammar> element an interpreter await spoken input from user. Semantic interpretation of this input fills variables of matching input field.

2.2 Algorithms of VoiceXML Interpreter

The figure 1 shows basic algoritms and implementation task for building VoiceXML interpreters. The next fundamental component of dialog management systems based on the VoiceXML interpreter is XML Parser. It creates hierarchical image of VoiceXML application from the file with VoiceXML code. This image representing the tree of the dialog.

All proceeding run within the document loop. The document loop define order of execution several dialogues and store information about variables and settings on the document level.

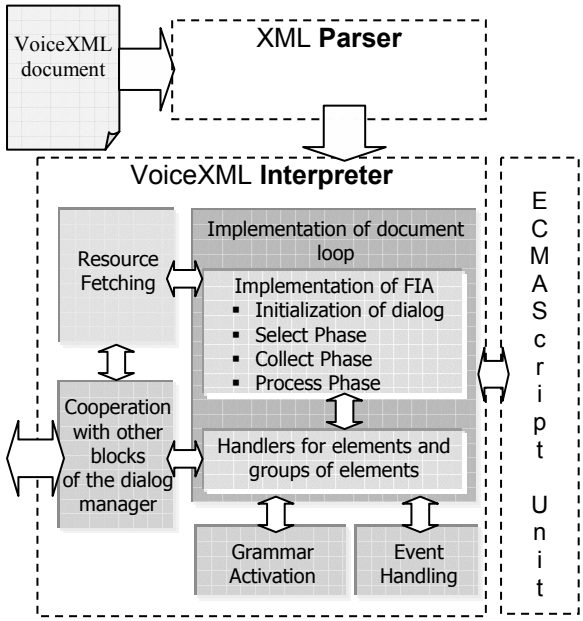

Fig.1 Algorithms of VoiceXML interpreter

Handling of elements within dialogs is managed by Form Interpretation Algorithm (FIA) that has four phases: initialization of dialog, select phase, collect phase and process phase. Last three phases make main loop. In the phase of initialization must variables on the dialog level be created and set up prompt and event counters. In the select phase one element of dialog is chosen that will be executed. After that is this executed in the collect phase. The purpose of the process phase is to process the input or event collected during the collect phase.

It is clear that Form Interpretation Algorithms determine the order of form elements execution only. Each VoiceXML element or group of elements must have their service function. For example service function of <prompt> element generates request for playing a prompt to the user.

During the collection of user′s spoken input more than one grammar can be in active state. The Grammar Activation Algorithm takes care of activation and deactivation of speech grammars. The VoiceXML works with static grammars. It does not define what kind of grammar format has to be used. In the VoiceXML 2.0 specification the XML or BNF grammar formats are recommended as specified in the Speech recognition grammar specifiacation (SRGS) released by the W3C. The choise of supported formats depends on concrete implementation.

Event Handling Algorithm is responsible for catching and handling events that occur during interaction between system and user (silence when the user has speak, unrecognized spoken input, hang up from the user side).

Resource Fetching Algorithm manages fetching of content from a URI occurs in VoiceXML applications. URI can refer to another VoiceXML document, audio files, grammars, objects and scripts.

VoiceXML supports the ECMAScript language. VoiceXML interpreter, XML Parser and ECMAScript unit create ground of the dialog management unit.

3 CONCLUSION

Our VoiceXML interpreter today does not support all commnads (elements) of VoiceXML 1.0 language. Form Interpretation Algorithm and ground of the others mentioned algoritms are fully implemented.

The interpreter works within dialog manager implemented in the simple voice platform VoiceON and FSC platform that is development in the research project Intelligent Speech Communication Interface. The Weather information service is running on these platforms. Interpreter is created in C++ programming language.

We were choosen version 1.0 even trough that today's interpreters support version 2.0. Building of the interpreter for version 1.0 present good ground for its update to version 2.0. Together VoiceXML 1.0 is useful tool for creating standart dialogues. This versions do not critical differ from yourself.

ACKNOWLEDGEMENTS

The work presented in this paper was supported by Grant Agency VEGA under Grant No. 1/1057/04 and Ministry of Education of Slovak Republic in the research project 2003 SP 20 028 01 03.

REFERENCES

[1] Juhár,J.-Čižmár,A.-Hintoš,Ľ: Speech-enabled information services. International Carpathian Control Conference ICCC2004, Zakopane, Poland, May,25-28,2004, pp. 837-842, 2004. (ISBN 83-89772-00-0)

[2] Lihan,S.-Juhár,J.: Hlasom ovládané telekomunikačné služby. Medzinárodná vedecká konferencia TELEKOMUNIKÁCIE 2003, pp.61-64, Bratislava, 2003. (ISBN 80-967019-4-0)

[3] Boyer, L. et al.: Voice eXtensible markup Language 1.0. W3C NOTE, Máj 2000, http://www.w3.org

[4] Beasley, R. et al.: Voice Application Development with VoiceXML. USA: Sams Publishing, August 2001. (ISBN 0-672-32138-6)