Telephone-Based Spoken Dialog System Using HTK-based Speech Recognizer and VoiceXML

Kinfe Tadesse Mengistu, Andreas Wendemuth
Cognitive Systems Group, Otto-von-Guericke Universität, 39106 Magdeburg, Germany
Email: {Kinfe.Tadesse,Andreas.Wendemuth}@E-Technik.Uni-Magdeburg.de

Introduction

Given the ubiquity of speech and the availability of the telephone coupled with the current state of speech technology, it is realistic to claim unconditional access to information. Telephone-based spoken dialog systems can take us closer to this claim of accessing information anytime, from anywhere, using any telephone.

A telephone-based spoken dialog system comprises of a telephone network interface to deliver calls into the system, a scheme for dialog authoring, a speech recognizer to accept requests from users, a method for comprehending requests, a means for response generation, a TTS engine for playing prompts and responses to the caller and a dialog manager to control the various components.

In order to build a practical dialog system, it is essential to define the application domain of the system so that optimal recognition performance can be obtained by using proper acoustic, language and semantic models. The DARPA Communicator Systems, for instance, are constructed to play the role of a travel agent speaking by phone with a customer in a specific domain of Airline Travel Planning [1].

To use a spoken dialog system framework for various task domains at different times, it is necessary to design a plug-and-play framework that can be used for multiple tasks and possibly in different languages. In this paper we describe such a framework.

Architecture

In this framework, the speech recognizer is built using ATK - an API designed to facilitate building real-time speech recognizers that use HTK-derived recognition resources [2]. Among other features, ATK allows a flexible use of grammars during the recognition process. Most notably, it uses a global configuration file where HTK compatible HMM models for a given task domain, and other recognition resources such as grammar, HMM list, and pronunciation lexicon are specified. Only the grammar for the initial dialog should be specified in the configuration file as the grammar for the rest of the dialog are written along with the dialog scripts.

To play prompts and responses to the user Microsoft's text-to-speech (TTS) synthesizer contained in Microsoft's Speech API 5.1 is used.

The interface to the ISDN telephone network, is implemented using CAPI (Common ISDN API) - the Application Programming Interface standard to access ISDN services [3]. For dialog scripting, the World Wide Web Consortium's (W3C) standard for human-computer dialog; namely, the Voice eXtensible Markup Language (VoiceXML) [4] and for telephony control functions, the Call Control eXtensible Markup Language (CCXML) of the same standard body (W3C) [5] are used.

At the core of the system is the VoiceXML platform OptimTalk. OptimTalk is preferred to other voiceXML interpreters such as OpenVXI [6] because it consists of not only a VoiceXML Interpreter but also a CCXML interpreter, and other abstract interfaces for the integration of ASR engine, TTS engine and telephony interface [7].

The high-level architecture of the system is depicted below (Fig.1).

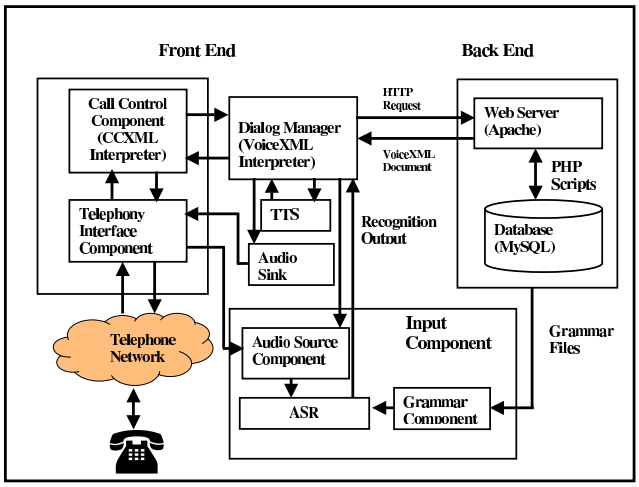

Figure 1: High-level Block Diagram of the System.

The Dialog Manager

A VoiceXML Interpreter is a piece of software that reads and processes VoiceXML pages as described by the VoiceXML language standard. The VoiceXML interpreter in OptimTalk is more than an interpreter. It loads the relevant dialog document from the Web Server that hosts VoiceXML applications, and executes the dialog by calling appropriate methods of the various components so as to play audio, accept user input via telephone, and pass them on to a speech recognition engine. It then determines what to do next according to the instructions in the active VoiceXML script. It is responsible for all communications with Web Servers using HTTP protocol. In short, it serves as the dialog manager of the system.

The Telephony Subsystem

The telephony subsystem is the interface between the telephone network and the application. It consists of a telephony interface component and a call control component (CCXML interpreter).

The rules for accepting and processing incoming calls are described in a CCXML document. The CCXML interpreter executes the commands in the CCXML document by calling the relevant methods of the telephony interface component. The methods for detecting an incoming call, establishing a connection, answering the call, streaming audio to the telephone, and capturing audio from the user via telephone are implemented in the telephony interface component using CAPI.

The URI to the initial dialog script of the application is also specified in the CCXML document and the CCXML interpreter instructs the dialog manager to fetch the document if the call is accepted. For every call control operation, the telephony interface component sends the corresponding event to the call control component.

Input Component

The input component wraps the ATK-based speech recognizer that is integrated with a separate Audio Source component, and a Grammar component. The methods to control the speech recognition process and to feed the recognizer with audio and grammar data are implemented in this component.

As the dialog manager directly instructs the audio source component in OptimTalk to capture audio from the user when spoken input is expected [7], OptimTalk's audio source component is used to capture audio instead of ATK's ASource component. The captured audio stream is then organized into wave packets and the packets are streamed to the buffer which connects the audio source to the coder component. The coder then gathers the incoming wave packets and passes them to the HTK module HParm for feature extraction [2]. The extracted features are then sent to the recognizer component that performs the decoding using the required recognition resources; namely, the pronunciation lexicon, the grammar, the HMM list and the HMM models that are prepared offline and defined in the configuration file.

The grammar component wraps the finite state grammars that appear in each VoiceXML document for each dialog and feeds the ASR engine with the grammar data in the required format. As HTK-based speech recognizers require grammar in HTK's Standard Lattice Format, a separate grammar component supporting this grammar format is developed. Furthermore, we exploited ATK's dynamic grammar enabling feature to enable dialog specific grammar at runtime. As the dialog progresses, the corresponding grammar file is loaded at runtime and is fed to the speech recognizer as appropriate. ATK also allows to incorporate the semantic content of the user's input into the grammar network.

The Database Interface

In a spoken dialog system of this architecture, most of the application logic resides on a Web Server and data on a Database Server. Apache web server is used to process HTTP requests from the client and serve VoiceXML documents and grammar files while MySQL database Server is used to implement the database(s). PHP scripting language is used to access the database and generate VoiceXML pages dynamically containing the requested information. Apache, PHP and MySQL are all open source and form a decent combination for creating data-driven applications.

Summary and Conclusions

We described a telephone-based spoken dialog systems framework that combines the power and flexibility of ATK-based speech recognizer and the convenience of VoiceXML. The framework can be used for various languages and application domains once the appropriate recognition resources are prepared for a known task and specified in the configuration file of the recognizer. Of course, the VoiceXML dialogs along with the grammar and the back-end should be properly set up.

We plan to use this framework for developing an Airline Travel Information System for English and Train Information System for German. We focus on building robust acousic models to be used in a real-time conversational system.

References

[1] A Aaron et al.: Speech Recognition for DARPA Communicator, in Proc. IEEE conf. Acoust. Speech Sig. Proc. (2001)

[2] Young, Steve. ATK: An Application Toolkit for HTK Version 1.4.1. Cambridge University, URL: `http://mi.eng.cam.ac.uk/~sjy/ATK_Manual.pdf`

[3] CAPI Association e.V. Common-ISDN-API. Version 2.0., URL: `http://www.capi.org/pages/downloads.php`

[4] W3C. Voice eXtensible Markup Language (VoiceXML) 2.0 Specification. Working Draft, URL: `http://www.w3.org/TR/voicexml20/`

[5] W3C. Voice Browser Call Control: CCXML Version 1.0. Working Draft, URL: `http://www.w3.org/TR/ccxml/`

[6] SpeechWorks OpenVXI 2.0, URL: `http://fife.speech.cs.cmu.edu/openvxi`

[7] OptimSys, s.r.o., URL: `http://www.optimsys.cz/developers/developers-guide/OptimTalk/?`