Vestec Automatic Speech Recognition Engine
Standard Edition
Version 1.1.1

Introduction

Vestec Automatic Speech Recognition Engine
Standard Edition
Version 1.1.1
Introduction

Copyright© 2009 Voice Enabling Systems Technology, Inc. All rights reserved.

145 Columbia Street West, Suite 1, Waterloo, Ontario, Canada N2L 3L2

Information in this document is subject to change without notice and does not represent a commitment on the part of VESTEC, Inc. The software described in this document is provided under a license agreement or nondisclosure agreement. You may not copy, use, modify or distribute the software without the express written permission of Vestec, Inc.

Table of Contents

About This Document

Vestec's Automatic Speech Recognition Engine (VASRE) is a speaker-independent speech recognition engine that supports a distributed architecture of servers and clients. VASRE works for Windows, GNU/Linux, and Open Solaris platforms to process an audio file or stream from external sources, such as telephone systems. The grammar can be built simply, with a list of keywords or pronunciations to be recognized, or with a more sophisticated industry standard.

This document introduces the main components and distributed architecture of VASRE.

Audience

This guide is intended for general users of VASRE who are interested in the main components and distributed architecture of VASRE.

Organization

This guide is organized as follows:

- Section 1 presents the main program components of VASRE.
- Section 2 outlines the distributed architecture of VASRE.

Conventions

Guides for VASRE use the following conventions:

- **Bold Arial** represents user utterances, recognized strings, and semantic results.
- `Courier New` represents file names, directory names, command line strings, and file contents.
- *Italic text* represents types, tokens, keywords, variables, and functions.
- Underlined text represents menu strings or texts in graphical user interface.
- *`Italic Courier New`* represents values replaced by you. For example, *`YYYY-MM-DD`* may represent a date in the year-month-day format.
- A paragraph starting with **N.B.** represents critical information or warning.
- For abbreviated terms, both singular and plural are spelled the same. For example, RM represents both resource manager and resource managers.

1 Main Components of VASRE

This section presents the four main program units of VASRE Standard Edition: recognition servers, clients, Resource Managers (RM), and grammar compilers.

1.1 Recognition Servers

The recognition server of VASRE supports multilingual, speaker-independent, large vocabulary, and continuous speech recognition. The acoustic models of the server were trained based on continuous hidden Markov modelling. The front-end of VASRE is equipped with noise reduction and voice detection algorithms to ensure smooth data input. To make its integration with application programs easier and more reliable, the output of the engine contains such information as the raw recognized text, confidence scores, and logical parsing for generating semantic results. For a speech application with a few hundred words or phrases to be recognized, the server processing time will be as short as one-fifth of the audio length. The recognition accuracy of VASRE is comparable to state-of-the-art systems in large vocabulary tests with up to several thousand words in the grammar. The server in the current version of VASRE processes 8 kHz linear PCM audio only.

1.2 Server Controller

The server controller is a program unit that reads the server configuration and starts one or more servers accordingly. Since VASRE allows multiple server processes running on a single machine, the server controller plays an essential role in implementing the VASRE distributed architecture.

1.3 Recognition Clients

The recognition client of VASRE acts as a bridge between the recognition server and external applications such as Private Branch Exchange (PBX) or Interactive Voice Response (IVR) systems. The C Application Programming Interface (API) is supported for effortless development of the recognition client. VASRE also provides a ready-to-go shared object file for easy integration with Asterisk, a widely-used software PBX. The recognition client accepts audio in both batched and streamed formats.

1.4 Resource Manager

The RM of VASRE manages speech recognition sessions by coordinating communications between recognition servers and clients. At system start-up, the RM reads the number of servers and the grammar size specified in the license file. Based on this license information, the RM monitors the status of the machines in the local network and recognition servers running thereon. The RM performs load balancing by assigning a given client request to an idle server on the least busy machine.

1.5 Grammar Compiler

The VASRE operation involves converting an easy-to-read text grammar into the binary format, which is directly loadable from the recognition server. This task is performed by the grammar compiler. As the format of the text grammar, VASRE uses the W3C standards called Speech Recognition Grammar Specification (SRSG) version 1.0 and Semantic Interpretation for Speech Recognition (SISR) version 1.0. The grammar compiler works with a dictionary to generate the pronunciations of words used in the

text grammar. The dictionary covers most common English terms. For words beyond the scope of the dictionary, the author of the grammar may rely on an auto pronunciation engine, which guesses the pronunciations of given words, or specify the pronunciations manually using predefined phonetic symbols. For more details, see Section 7 of the Grammar Developer's Guide.

2 Distributed Architecture

The VASRE Standard Edition supports a distributed architecture for reliable and scalable operation of recognition servers. VASRE's distributed architecture comprises three components: the recognition servers, clients, and RM. The servers and RM are the permanent components of the distributed architecture, while the clients can be implemented either as permanent or temporary components, according to your design.

The RM is the control tower of the distributed architecture. It manages one or more recognition servers and coordinates communication sessions between servers and clients. The recognition servers need not run on the same machine of the RM; they can even be distributed over multiple machines. The recognition servers have two states: busy or idle. A server is busy if its port is connected to a recognition client. A server is idle if it is not busy. Under the idle state, the recognition servers periodically communicate with the RM to report their status.

If your system is supposed to work with many servers and clients, it is a good strategy to distribute the servers over multiple machines to avoid performance degradation. For each machine, the memory consumption and CPU-seconds required for speech recognition will be roughly proportional to the number of busy servers running on that machine.

To illustrate this, we performed a resource requirement test using a PC with AMD Athlon 64-X2 Dual Core Processor5600+ 2.9 GHz and 3GB RAM. With a grammar containing 500 keywords, the server ran on the PC, processing an audio input 1.6 seconds in length, issuing from clients at a rate of one every fourteen seconds. The relationship between the number of ports and the corresponding memory usage and CPU usage is listed in the following table:

# Ports	Memory Usage (MB)	CPU Usage (%)
0	1	0
1	55	1.74
2	101	3.45
4	193	6.45
8	378	13.11

The RM balances server loads over different machines. For example, if two servers are busy on Machine A and one server is busy on Machine B, the RM will guide the next client request to Machine B, even though Machine A still has idle servers.

The recognition client initiates a speech recognition session by asking the RM which recognition server the client should connect to. A server can serve only one client at a time; hence, you must start as many servers as there are clients to be run at the same time.