HEISON CHAK

Asterisk and LumenVox ASR

Heison Chak is a system and network administrator at SOMA Networks. He focuses on network management and performance analysis of data and voice networks. Heison has been an active member of the Asterisk community since 2003.

heison@chak.ca

IN THIS ARTICLE I INTRODUCE THE LumenVox Speech Recognition software and how it can be tied into a VoIP platform. Asterisk as an interactive voice response (IVR) system can utilize text-to-speech (TTS) and automatic speech recognition (ASR) technologies to improve system usability and encourage customer interaction. I will use the LumenVox ASR engine as an example.

Text-to-Speech

Text-to-speech technologies allow computer systems to convert text into spoken languages. TTS engines are capable of synthesizing male or female voices in various languages. Quality and accuracy are dictated by the affordability of the products; more expensive products tend to produce easier-to-understand phrases than their less expensive counterparts.

An open-source TTS engine, such as Festival, is suitable for starting a TTS application, but one may soon realize that such development efforts deserve more pleasant and accurate voices. Cepstral (~$30) offers a better synthesizer and is often involved in proof-of-concept of application developments. As we move into more sophisticated products (with a $300 to $3000 price tag), the synthesizers tend to become more accurate—instead of typing the text in a special phonetic way, the effort can be spent on achieving business requirements.

For example, if you want output of a TTS to sound like "Hello, how are you?" you may need to input the following text:

"Hah loow, how arh u" - Festival, open source
"Ha lo, how are you?" - Cepstral, $30 products
"Hello, how are you?" - better commercial products, >$3000

To present to a caller of Asterisk the synthesized audio, the Asterisk Gateway Interface (AGI) is often used. AGI allows external applications (shell, Perl, PHP, C, etc.) to have read and write access to the voice channel. Read access can allow these AGI scripts access to DTMF (or touch-tone) inputs, whereas write access enables the scripts to play out the TTS synthesized audio to the caller, in response to the caller's DTMF input or selection.

Automatic Speech Recognition

We often see the terms *speech recognition* and *voice recognition* used interchangeably. In academic con-

texts, voice recognition identifies the speaker, whereas speech recognition distinguishes the words or phrases spoken by the speaker (i.e., contents of the speech). Speech recognition converts spoken language into readable text, and there are two types of speech recognition systems—speaker-dependent and speaker-independent.

Speaker-dependent systems require prescribed text and phrases to be read by the speaker prior to conducting recognition; these materials are used to train the engine's vocabulary. A typical use for user-dependent ASR is speech transcription—using speech to replace nonspeech input. A properly trained speaker-dependent ASR system can transcribe speech at an average rate and accuracy of 17 words per minute and 84.9%, respectively.

Speaker-independent systems, often seen in telephony applications (e.g., auto attendant and interactive voice response systems), can be classified as ASR, computer-driven transcription of spoken language into readable text in real time; it handles pronunciation discrepancies in words or phrases among users. ASR has been viewed as the bedrock for building next-generation telephony applications. Not only is it a more natural input method—allowing a higher level of engagement from callers—but it also enables menu options to go beyond the limitation of a touch-tone keypad.

For example, imagine a caller who wants to speak with John Smith. On a traditional IVR/auto attendant, the caller may be entering "76484" to spell "Smith" and the system may respond with:

"Press 1 for Alan Smith, 2 for Bob Smith, 3 for Jan Smith, ..., 5 for John Smith."

If there is more than one match, the caller may have to wait for all of the choices before making a selection. With an ASR-capable IVR system, the caller can simply say "John Smith" and conversation can be established much more quickly. In addition to traversing the employee directory more easily by flattening the menus, ASR can also allow more free-form input, such as for country and city names, which are traditionally difficult to implement.

ASR Software for Asterisk

Sphinx is open-source ASR software that runs under Linux. However, Asterisk's handling of audio in 8 kHz samples is not readily compatible with the 16 kHz that Sphinx expects. Up-converting is an option but the end result is far from ideal. Sphinx has been reported to do a fine job on a native 16 kHz sample, just not on up-converted samples.

LumenVox (www.lumenvox.com) addressed the need for an affordable speech solution for the Asterisk community by integrating its speech recognition engine into Asterisk 1.4. Users can choose to purchase a supported platform with packages tailored toward deployment needs (e.g., number of concurrent recognitions and size of the vocabulary). LumenVox supports RedHat Enterprise and Fedora Core; Debian support has been added to the mix recently.

Customers can choose to download Speech Recognition Engine, License Server for their favorite platform, as well as an Asterisk Connector for the specific version of Asterisk they are running. There is also a Wintel-based Tuner, which can be used to fine-tune the speech engine.

In Debian, there are a number of required packages to run the Speech Recognition Engine:

```
# apt-get install libboost-filesystem1.33.1 libboost-program-options1.33.1
```

```
libboost-regex1.33.1 libboost-thread1.33.1
libboost-date-time1.33.1 libxul0d libc6-i686
iceweasel libnspr4-0d
```

After apt-get, the downloaded Debian LumenVox packages can be installed without a hitch:

```
# dpkg -i lumenvoxsre_8.0-106_i386.deb   # Speech Recognition Engine
# dpkg -i lumenvoxmrcpserver_8.0-106_i386.deb # License Server
```

The Asterisk connector contains binary distribution of a module, res_speech_lumenvox.so, that works with Asterisk's Generic Speech Recognition API (res_speech.so). With the API loaded, the LumenVox module can be loaded into Asterisk and is ready to test:

```
[lumenvox-test]
exten => s,1,Answer
exten => s,n,Wait(1)
exten => s,n,SpeechCreate
exten => s,n,SpeechLoadGrammar(yesno|${LUMENVOX_PATH}/ABNFBoolean.gram)
exten => s,n,SpeechActivateGrammar(yesno)
exten => s,n,SpeechBackground(beep)
exten => s,n,Verbose(1,Result was ${SPEECH_TEXT(0)})
exten => s,n,Verbose(1,Confidence was ${SPEECH_SCORE(0)})
```

The sample dial plan creates a speech object and allows a caller to say "Yes" or "No" at the beep. The recognition will be in the form of a Boolean value "true" or "false." The results can be viewed from the Asterisk console with verbosity set to at least 1.

Tuning

With ASR, tuning becomes essential to providing an intuitive tool for callers to engage in, rather than having to press "0" all the time to reach an operator. Unlike traditional nonspeech input methods (e.g., touch tone), what a caller may say is unpredictable. Tuning is the key to ensure reliability and improve accuracy on an ASR system:

- Dial-plan Tuning—The IVR prompt may have instructed the user to say "Main Menu" to return to the main menu. Yet if the caller says "Go Back," the dial plan may need to incorporate that as a valid command and take appropriate action (e.g., "Go one level up, except if already in Main Menu").
- Speech Database Tuning—Using the "Go Back" example, it is possible for the speech engine to fail in correctly identifying the phrase. In that case, the Windows-based Tuner can be used to correct the misinterpreted text. Using the Tuner, corrections can be made to the engine's database so that the spoken language will be interpreted as "Go Back" rather than "Go Bat." This will require that the original audio be recorded and stored for analysis—a tunable feature in the engine itself.

Some of the worse ASR-capable systems out there are the ones that have both touch-tone and ASR, especially the ones with long explanations about how to use the system followed by a large number of touch-tone options. It makes a caller yearn to press "0" and get a real person on the phone, defeating the purpose of an ASR system.