By Richard A. Frost

CALL FOR A PUBLIC-DOMAIN SPEECHWEB

All that's missing is innovative use of existing technology and the encouragement of public participation.

Imagine using your cell phone to navigate through a network of hyperlinked speech applications as rich and diverse as the visual applications on the conventional Web. On holiday at the beach, you could use it to navigate through a vendor's site for information on the new camera you're using to take snapshots of your friends. In the evening, by the fire, you could verbally browse a library of electronic stories, picking the one most appropriate for the moment. Imagine, too, being able to build speech applications and add them to a public-domain SpeechWeb as easily as you build and add pages to the conventional visual Web. You could have a speech-accessible resumé. Information (such as directions on how to get to your home) could be made available for your guests to access through hands-free speech in their vehicles. Companies (with the expertise) could create natural-language speech interfaces to their products and services, making them available to anyone, anywhere using a handheld SpeechWeb browser. Moreover, a public-domain SpeechWeb would allow blind users to participate more fully in our knowledge-based society.

A Kelsey Group study estimated that speech-enabled services could generate more than $4.6 billion in revenue for North American wireless carriers and $25 billion worldwide by 2006 [6]. However, says Dale Hartzell of Sandcherry, Inc., a software company specializing in speech products, this world of speech-enabled services will be a reality only if speech solutions are able to deliver flexibility, scalability, and economy, and speech applications can be created and deployed as easily as text- and picture-based Web services are today [6]. This has not been possible due to the fact that existing architectures for distributing speech applications are not conducive to public involvement in the construction of SpeechWebs.

Here, I present a new architecture—called Local Recognition Remote Processing (LRRP) I've developed at the University of Windsor—for distributed speech processing, showing how it can be implemented through existing software and Web protocols and that it is conducive to the development of a public-domain SpeechWeb (see the figure). My goal is to encourage participation in the construction of a public-domain SpeechWeb, showing the ease speech browsers and applications can be built using existing software and the ease hyperlinked speech applications can be built on conventional Web servers.

Existing Architectures

Speech applications can be distributed via three main architectures: speech interfaces to HTML pages; networks of hyperlinked VXML pages; and call centers containing speech applications. Early versions of the first (in the early 1990s) allowed users to scan HTML pages using a standard set of voice commands. More recent versions translate Web pages into speech dialogues. Recognition grammars, or sets of rules defining user input languages, are derived from the Web pages and used to direct speech-recognition software, thereby improving speech-recognition accuracy. The Speech-Aware Multimedia (SAM) system, introduced in 1995 by Texas Instruments, [7] is an example that also allowed grammars (defined by application developers) to be associated with so-called "smart pages." When an end user downloads a smart page, the associated grammar tailors the speech interface to recognize utterances that are interpreted locally by the speech browser.

The use of speech interfaces to HTML pages goes some way toward creating a public-domain SpeechWeb. They are easy to install by nonexperts and provide speech access to knowledge already available on the Web; local speech recognition allows users to train the interfaces for their voices, further improving recognition accuracy. However, this architecture is somewhat limited for three main reasons:

HTML pages are typically not designed for speech browsing. Icons and other visual effects are used to convey information about links, and pages are usually structured to take advantage of the users' ability to scan the page in two dimensions;

Recognition grammar can't be derived. In many applications, the browser is unable to derive the recognition grammar directly from the HTML page; and

Speech can't always be recognized. Keyword and phrase-matching techniques are not well suited to speech access, owing to the large number of words and phrases that might be input by users, resulting in poor speech-recognition accuracy.

More information on tools that enable speech access to Web pages is in the Eurescom P923-PF project's documentation (www.eurescom.de/).

In the second architecture, VXML, which is like HTML, is used to create hyperlinked speech applications [9]. VXML pages, which are executed on VXML browsers, include commands for prompting user speech input, invoking recognition grammars, outputting synthesized voice, iterating through blocks of code, calling local Java scripts, and hyperlinking to other remote VXML pages downloaded in a manner that's like the linking of HTML pages in the conventional Web. Though this architecture is likely to be a major aspect of the development of a public-domain SpeechWeb, it is also limited by two factors:

- Many speech applications require natural-language access to large knowledge sources and are best executed on powerful remote servers rather than on VXML browsers on end-user devices; and
- Developers should be able to create their speech applications in whatever language they prefer and not have to embed them in VXML pages.

In the third architecture, which involves telephone access to remote speech applications, call centers allow users to phone in and speak to their applications. Speech recognition and application processing are carried out at the call center. A fairly recent innovation is the integration of VXML pages stored on call-center Web servers with speech dialogues accessed through remote client telephones. (For example, the city of Montréal implemented this approach in 2003 to accommodate the hundreds of thousands of calls it receives annually.) The resulting architecture, which can be viewed as a variant of the second architecture, supports sophisticated applications requiring powerful processing. It also allows applications to reside on conventional Web servers, reducing administrative overhead. However, using the call-center architecture in a public-domain SpeechWeb also involves three main limitations:

Need to store voice profiles. In order to obtain adequate speech-recognition accuracy for advanced applications, user voice profiles must be stored at each call center or at the user site and transferred to the call center each time it is accessed, neither of which is practical;

Need for expensive software. Application providers must employ specialized, typically expensive, software to allow their applications to be accessed from remote telephones; and

Need to support hundreds of concurrent users. Call centers must be able to provide speech-recognition capabilities for hundreds of concurrent users.

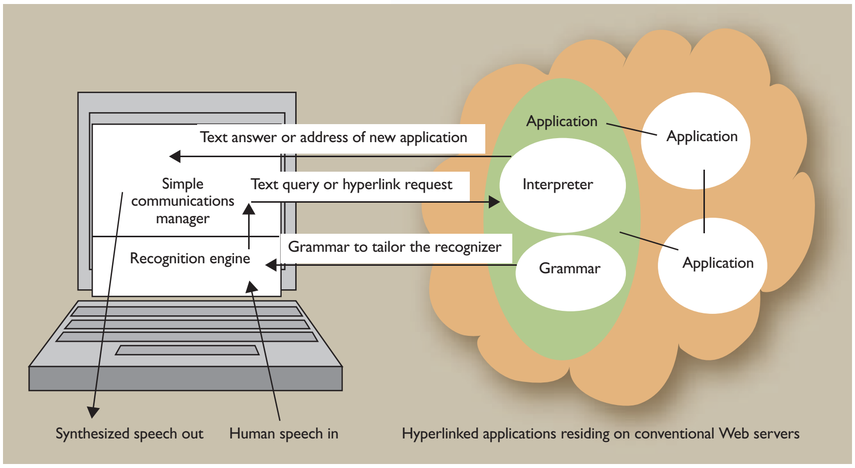
SALT vs. VXML

The protocol called Speech Application Language Tags (SALT) developed by Microsoft, Intel, and other sources [11] includes tags (fewer than VXML) that are intended to be added to HTML pages so they can be browsed through speech. An advantage of this approach is that existing Web browsers are more easily extended to accommodate SALT tags; SALT relies on the execution model of HTML. On the other hand, VXML is an independent language that can be used to create and hyperlink speech applications. VXML and SALT might converge over the next five years into a single voice-application markup language.

The Microsoft Speech Server (MSS), which supports SALT, is intended to simplify the integration of an enterprise's Web services, speech processing, and telephony. MSS can help implement all three of the architectures described earlier for distributed speech applications. However, the role MSS might play in the creation of a public-domain SpeechWeb is less clear, owing to its high cost and complexity for nonexpert SpeechWeb-content developers.

A New Architecture

Statistical language models, according to [8], were popular in 1995, but by 2001 grammar-based models had claimed the preeminent position in commercial speech-recognition products. My proposed LRRP architecture takes advantage of this fact, making use of the grammar-based speech-recognition software readily available today. In the LRRP architecture, speech applications and their associated recognition grammars are stored on regular Web servers, and LRRP speech browsers reside on end-user devices. When users access remote applications, grammars are downloaded to their browsers where they are used to tailor the speech recognizer for the application. When a user utterance is subsequently recognized, it is sent to the remote application for processing. If the utterance is a request for information, the result is returned in the form of synthesized voice. Speech recognition is local, and application processing is remote. Alternatively, users might ask to be connected to other speech applications by uttering a speech hyperlink (such as "Can I talk to the chemistry expert?"). The Web address of the new application is then returned to the browser, which downloads a new recognition grammar from that address and subsequently sends all recognized utterances to the new application.


Text answer or address of new application
Simple communications manager
Text query or hyperlink request
Recognition engine
Grammar to tailor the recognizer
Application
Interpreter
Grammar
Application
Application
Synthesized speech out
Human speech in
Hyperlinked applications residing on conventional Web servers

The local recognition and remote processing SpeechWeb architecture.

The LRRP architecture has something in common with each of the three architectures discussed earlier. Recognition is local, as it is with speech interfaces to HTML pages. The downloading of recognition grammars from remote applications is similar to SAM's smart pages. Like VXML, LRRP is intended to support access to applications designed for speech (rather than multimodal access to HTML pages). And finally, it involves remote execution of applications, like the call-center architecture. In terms of support for a pub-

My goal is to demonstrate the ease speech browsers and applications can be built using existing software and the ease with which hyperlinked speech applications can be built and deployed on conventional Web servers.

lic-domain SpeechWeb, LRRP reflects a combination of the following advantages:

Applications on conventional Web servers. Applications that reside on conventional Web servers can be written in any language, provided that input and output conform to the Web communication protocol being used;
Available to nonexperts. Nonexperts can create simple applications as scripts that return canned answers to user queries, while advanced developers can create complex applications that use natural-language processors and databases residing on powerful server-side machines;
Improved client-side speech-recognition accuracy. Recognition accuracy is improved through the downloading of application-specific grammars and locally maintained user voice profiles; and
Communication through text. Using conventional Web protocols, clients and servers communicate through text, which is efficient when users access the SpeechWeb through commercial wide-area wireless networks.

Prototype Implementation

Aiming to demonstrate the viability of the LRRP architecture, my research group and I have experimented with a number of technologies for building browsers and deploying speech applications. We built early versions of the browser (1998) in Java using IBM's proprietary speech APIs. I demonstrated them at the 1999 Conference of the Pacific Association of Computational Linguistics [5] and at the 2002 Conference of the American Association for Artificial Intelligence [4].

Last spring, one of my students and I rebuilt the browser as a single VXML page, integrating it with two simple Java objects [12]. The single VXML page remains on the local device and accesses remote hyperlinked non-VXML speech applications. In order to run this upgraded browser, users install the Java runtime environment and a VXML interpreter, both readily available as free downloads on the Web. In order to add a speech application to the SpeechWeb, developers need access to a Web server and software to create their text-in/text-out applications. We tested the browser on PCs and laptops using both wired and wireless connections to local and wide-area networks, then demonstrated it in 2004 at the second annual conference on Communications Networks and Services Research in Fredericton, N.B. [3].

My somewhat unorthodox use of VXML (to implement the LRRP browser) provides several notable advantages: the browser can be distributed freely; VXML interpreters are available for a range of devices, including desktop PCs, laptops, and pocket PCs; and the browser inherits improvements in speech-recognition technology as they're integrated into new versions of VXML interpreters. Finally, this method of constructing the browser facilitates integration of the LRRP architecture with the conventional "networks of VXML pages" architecture; the result is a richer environment for SpeechWeb application development. I also expect that the browser can be implemented as a single HTML page containing appropriate SALT tags, with comparable advantages.

Technological Challenges

The functionality of a public-domain SpeechWeb will be limited by three main issues:

- Lack of readily available techniques and tools to create sophisticated natural-language speech applications;
- Inability of speech-recognition technology to support free-flowing speech; and
- Limited availability of speech-recognizers that execute locally on lightweight devices (such as cell phones and handhelds).

Addressing the first involves development of special tools and techniques. Tools are needed to facilitate construction of natural-language applications by nonexperts. Microsoft, IBM, and other sources have developed software for constructing natural-language interfaces to proprietary database systems. In addition, advanced developers can construct natural-language processors using Yet Another Compiler

Compiler, or YACC, as well as the Cup parser generator for Java and the parser combinators in logic and functional programming. Compositional semantic theories (such as those devised by Richard Montague at the University of California, Los Angeles [2]) can help develop the semantics for the subset of natural language that are appropriate for a given application. Even with these tools and theories, development of natural-language interfaces remains a difficult technical task for all but the experts.

Guidelines are needed to help developers create recognition grammars that provide the proper mix of accuracy and robustness for a given application. Although a number of tutorial-style documents on recognition-grammar design are available on the Web, they must be augmented with more comprehensive material based on empirical results obtained through grammar-based recognition engines in a range of applications.

Design rules are needed to construct sets of hyperlinked speech objects and decompose large speech applications into sets of hyperlinked units, balancing recognition accuracy, robustness, and ease of navigation; these tasks are similar to dialogue design [10].

The second limitation—lack of support for free-flowing speech in current technology—was discussed in [1], which identified the steps researchers must take to improve accuracy and robustness, thereby facilitating the mainstream adoption of speech technology. The recommendations of these researchers, also discussed in [1], apply to the development and acceptance of a public-domain SpeechWeb.

Fortunately, the third limitation—the need for speech recognizers on lightweight end-user devices—has been overcome to some extent by the recent availability (June 2005) of freely downloadable browsers supporting the multimodal markup language XHTML+Voice, also known as X+V. This language was developed by Access Systems, IBM, Motorola, and Opera Software and has been submitted to the World-Wide Web Consortium for approval. Trial versions of X+V multimodal browsers for PCs, pocket PCs, and cell phones are freely downloadable from www-306.ibm.com/software/pervasive/multimodal/. They can be used to execute a single X+V page constituting a SpeechWeb interface and provide local state-of-the-art speech-recognition and text-to-speech synthesis.

Conclusion

The conventional Web has grown dramatically in terms of numbers of users and Web sites, e-commerce activity, collaborative applications, information accessibility, and social networking over the past decade. The availability of free browsers and the ease of developing HTML pages represent important factors. My aim here is to motivate development of a public-domain SpeechWeb with which speech browsers and applications can be built and deployed using readily available software and conventional Web protocols. C

References

1. Deng, L. and Huang, X. Challenges in adopting speech recognition. *Commun. ACM 47,* 1 (Jan. 2004), 69–75.
2. Dowty, D., Wall, R., and Peters, S. *Introduction to Montague Semantics.* D. Reidel Publishing Company, Dordrecht, Boston, Lancaster, Tokyo, 1981.
3. Frost, R., Abdullah, N., Bhatia, K., Chitte, S., Hanna, F., Roy, M., Shi, Y., and Su, L. LRRP SpeechWebs. In *Proceedings of the Second Annual Conference on Communication Networks and Services Research* (Fredericton, N.B., Canada, May 19–21). IEEE Computer Society, 2004, 91–98.
4. Frost, R. SpeechWeb: A Web of natural language speech applications. In *Proceedings of AAAI Intelligent Systems Demonstrations* (University of Alberta, Edmonton, July 28–Aug. 1). AAAI Press/MIT Press, 2002, 998–999.
5. Frost, R. and Chitte, S. A new approach for providing natural-language speech access to large knowledge bases. In *Proceedings of the Conference of the Pacific Association for Computational Linguistics* (University of Waterloo, 1999), 82–90.
6. Hartzell, D. Simplifying Speech-Enabled Solutions or Deploying Speech-Enabled Services Should Be as Easy as Deploying Web Services. Invited talk at the Conference on Voice Enabled Services, London, Jan. 2003.
7. Hemphill, C. and Thrift, P. Surfing the Web by voice. In *Proceedings of the Third ACM International Multimedia Conference* (San Francisco). Addison-Wesley, Reading, MA, 1995, 215–222.
8. Knight, S., Gorrell, G., Rayner, M., Milward, D., Koeling, R., and Lewin, I. Comparing grammar-based and robust approaches to speech understanding: A case study. In *Proceedings of Eurospeech 2001, the Seventh European Conference of Speech Communication and Technology* (Aalborg Denmark, Sept. 3–7, 2001), 1779–1782.
9. Lucas, B. VoiceXML for Web-based distributed conversational applications. *Commun. ACM 43,* 9 (Sept. 2000), 53–57.
10. McTear, M. Spoken dialogue technology: Enabling the conversational user interface. *ACM Computing Surveys 34,* 1 (2002), 90–169.
11. SALTforum Speech Application Language Tags (SALT) 1.0 Specification. SALTforum, 2002; www.saltforum.org/.
12. Su, L. and Frost, R. A novel use of VXML to construct a speech browser for a public-domain SpeechWeb. In *Proceedings of the 18th Conference of the Canadian Society for Computational Studies of Intelligence* (Victoria, B.C., 2005), 401–405.

Richard A. Frost (rfrost@cogeco.ca) is a professor in the School of Computer Science at the University of Windsor, Windsor, ON, Canada.

This research is supported by the Natural Science and Engineering Council of Canada. Sanjay Chitte, Kunal Bhatia, and (Jack) Li Su (all graduates of the University of Windsor) helped implement versions of the SpeechWeb browser.