Deploying Speech Applications over the Web

David Goddeau, William Goldenthal, and Chris Weikart

Digital Equipment Corporation
Cambridge Research Laboratory
1 Kendall Square, Bldg 700
Cambridge, MA 02139
`http://www.research.digital.com/CRL/`
email: dg@crl.dec.com, thal@crl.dec.com, weikart@crl.dec.com

ABSTRACT

At Digital Equipment Corporation's Cambridge Research Lab (CRL), the Speech Interaction Group has been focusing on building speech applications for deployment over the World-Wide Web. Web-based speech applications require the browser to capture and transmit speech to remote servers for back-end processing, maintain application state, and present multi-media responses. This paper describes the group's strategy for delivering speech applications built around a mechanism, the DIGITAL Voice Plugin, for capturing and transmitting audio from a browser. It describes a conversational application implemented within this framework and discusses the problems of delivering these systems on the Web. In addition, we briefly touch upon some other Web-based speech applications that have been developed at CRL.

1. Introduction

The emergence and exponential growth of the World-Wide Web offers opportunities for new modes of human-computer interaction. At Digital's Cambridge Research Lab (CRL), the Speech Interaction Group has been focusing on building speech-based applications for deployment over the Web. We believe it is interesting to target speech applications to the Web because it provides an ideal way to reach a wide audience with a rich, multi-modal user interface. Applications accessible through the Web are naturally mobile, unlike stand-alone programs.

Conversational spoken language is a natural and powerful mode of human computer interaction. It allows a user to easily specify their needs and constraints in a given domain, rather than forcing them to select from a limited list of options. Conversational systems, operating in limited domains of expertise, have improved in performance to the point where deploying these systems is becoming practical. When deployed over the Web, they can provide a useful supplement to the browsing-mode interaction currently supported.

Despite its advantages, the Web in its current form poses several challenges for the delivery of speech-based interaction. Applications which would be straightforward to implement in a conventional client-server architecture require considerable cleverness to integrate into the restricted environment of the browser and the limitations of the current Web protocols. Among the challenges relevant to delivering conversational systems are:

1. How to capture audio samples from the user and transmit them to the processing servers,
2. How to synthesize and present spoken responses back to the user, and
3. How to maintain the dialog state associated with each current user within the state-free protocols of the Web.

In addition, we wish to solve these problems while imposing the minimum possible requirements on the client machine, considering not only computational requirements (CPU cycles and memory) but software requirements (client-resident code) as well.

The limitations of the Web are widely recognized and many alternatives are being put forward to extend the fundamental architecture. As of this writing, no single alternative has emerged as dominant and the situation remains fluid. In light of this, we have confined our current approach to those technologies which are fairly stable and widely supported.

This paper presents an overview of our approach to deploying speech-based applications over the Web. It describes a conversational system implemented within this approach and discusses solutions to problems listed above. In addition, we briefly touch upon some other Web-based speech applications that have been developed at CRL.

2. Previous Work

Previous work in this area falls into two main categories: distributed conversational systems, and systems which extend Web browsers with speech recognition capabilities. A review of architectural tradeoffs and alternatives was presented in [5]. Conversational systems which operate in a distributed Internet environment include systems developed at MIT and SRI. The MIT GALAXY system [3, 8] is a multi-domain system which distributes speech recognition and language understanding among remote servers and uses an HTML enabled client to communicate with the user. It is also capable of running remote clients which capture user voice input via telephone and display the response via the X Windows protocol (or by driving a Web browser). The SRI system [4] uses a Java applet to manage client/server communication and display. As in the remote GALAXY system, voice capture is via a telephone call into the server.

The second class of related systems are speech-enabled Web browsers [1, 2]. These typically use client-side speech recognition as a control mode for a Web browser, supplementing or replacing the mouse. Functionality includes basic navigation controls and anchor selection. The TI system [1] allows Web sites to specify a grammar within an HTML document and associate Web links with the sentences or the corresponding language. When such a page is loaded into the browser and one of the specified sentences is recognized, the associated URL is loaded as if typed by the user.

The conversational system presented here differs from speech-enabled Web browsers in that it supports full multi-turn interaction, requiring language generation (spoken responses), discourse resolution, and dialog planning. This introduces several complexities not present in the speech-based browsing paradigm. In addition, the architecture allows for off-loading the recognition process from the client machine. This work differs from other distributed conversational systems such as the MIT GALAXY system in that the audio capture and transport, is kept within the Web browser paradigm rather than requiring a separate channel. This is particularly important for at-home delivery where only a single communication line is available.

3. Audio Capture and Transmission: The DIGITAL Voice Plugin

Our goal at CRL is the deployment of applications over the Web in a direct and intuitive manner. To achieve this, we have developed a technology, the DIGITAL Voice Plugin, which embeds a microphone user interface within a Web page. The plugin is available from `http://interface.digital.com/voice/` for free download. Since its release in early March, over 15000 copies have been downloaded.

3.1. Technical Description

The DIGITAL Voice Plugin is a Netscape plugin for Win32 platforms, which captures and transmits speech from a microphone user interface embedded within a Web page to a remote location. The plugin GUI is simple and takes up little space. The GUI provides a push-to-talk/push-to-stop microphone icon and displays a volume level meter during speech input. After recording, the GUI allows the user to playback what was said and optionally send or throw it away. The captured audio is posted to a URL specified by the user or page in which the plugin is embedded.

Figure 1: DIGITAL Voice Plugin with Light-Bulb Activated

In addition, the plugin communicates with our Web server whenever it loads into a page. This allows us to activate two "hooks": a pop-up dialog which notifies the user when a new version of the plugin is available, and a "light-bulb" button which appears when we have something to announce. When the light-bulb is pressed, the plugin loads a designated Web page.

3.2. Plugin Applications and Deployment

We have developed a number of speech-based applications using the Voice plugin, aside from conversational systems. Some example applications are described below.

Voice Email: To transmit captured audio, a MIME-encoded message is posted to a target URL. By default, the `mailto:` protocol is used with an email address. This, in conjunction with an SMTP server, allows the plugin to send audio as an attachment to an email message. The recipient can listen to such a message with any MIME-compatible email client and any sound player that supports the Microsoft `.wav` format.

Speech Data Collection: In order to develop speech applications for the Web, we need a training corpus which accurately reflects the acoustic and channel conditions expected by the target application. For this reason, we have a built a data-collection application which embeds the Voice Plugin and steps the user through a series of utterances. The captured audio is posted to a cgi-bin program on our Web server via an http: URL.

The plan for deploying this application is to recruit from the installed base of Voice Plugin users through the light-bulb mechanism. This mechanism can exploit information about the user's Internet domain to perform data-collection according to language or geographic domain. This capability offers us an opportunity for targeted speech corpus development on a world-wide basis.

Speaker Identification/Verification: Another use for the Voice Plugin technology is a speaker verification system for Web page access restriction. This application is described in detail elsewhere in these proceedings [6]. Basicly, a user attempting to access the restricted site is asked to speak into the Web microphone. The speech is compared against stored models on a server and the user is admitted only if a sufficiently good match is found.

Animated Avatars: Finally, a novel browser-based application couples the Voice Plugin with client-side phonetic recognition and transmits the audio waveform annotated with phoneme alignments to a facial animation display. This application, also described in these proceedings [7], allows people to communicate through lip-synched synthetic avatars.

The above illustrates the variety of applications that can be delivered over the Web using the DIGITAL Voice Plugin. The remainder of this paper will discuss another application area, conversational systems, in detail.

4. Web-Based Delivery of Conversational Systems

The conversational systems work at CRL was initially inspired by research in the Spoken Language Systems group at MIT's Laboratory for Computer Science. Our goal is to extend this work concentrating on those problems most important for commercialization. In developing the component technologies for conversational systems, our research strategy focuses on three main issues: portability, robustness, and stability.

We have chosen to concentrate on delivering conversational systems over the World-Wide Web rather than through the telephone network, and to develop a server-based architecture to minimize both the computation and software required on the client machine. Figure 2 shows the architecture of our current system. The recognition server and application server operate on the content-provider's Web site, and the client program is a Netscape browser augmented with a plugin.

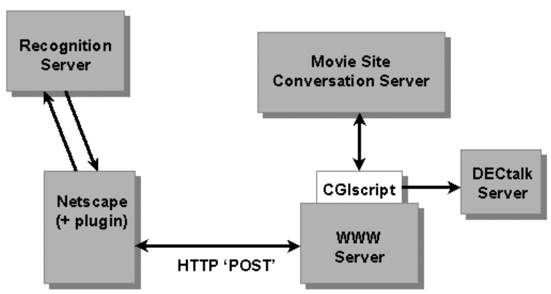

Figure 2: Web-Based Architecture for Conversational Systems.

4.1. Audio Capture and Utterance Processing

The user's speech is captured by a modified version of the DIGITAL Voice Plugin described above. The version streams the input waveform to its destination, a recognition engine. This allows recognition to begin as soon as the user starts speaking, thus reducing the overall system latency. Currently, the audio waveform is transmitted uncompressed, which is acceptable for LAN operation but less desirable over slow links. We are investigating compression schemes and their effects on recognition accuracy. The signal is transmitted via a TCP socket connection to a recognition server, which returns its transcription to the plugin on the same connection. The plugin packages the transcribed utterance into an HTTP POST request to the URL of the application server (specified in the embed tag for the plugin). The response to the POST request is an HTML document and is displayed by the browser in the usual fashion. Figure 3 shows the interface and display for the system.

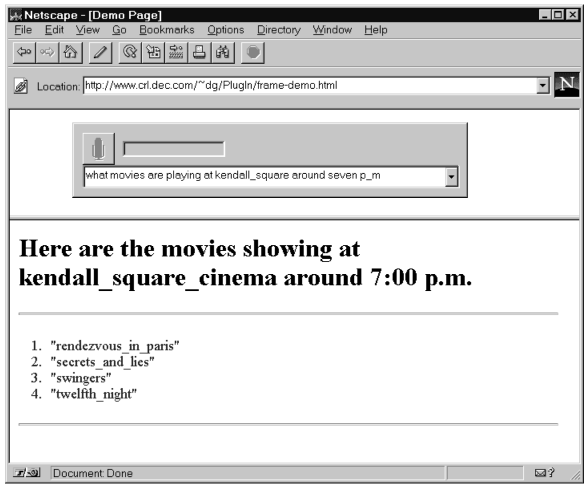

Figure 3: Browser Interface to Movie Information Conversational System.

4.2. Conversation State Maintainance

One of the differences between a conversational system and traditional Web document browsing is that a conversation maintains a state which is referenced and updated by every utterance. A client-server implementation of these systems raises the issue of where to maintain the dialog state. In previous work [8], the state was maintained on the client and transmitted to the server when needed. This is a natural approach in a multi-client environment as it frees the server from managing and garbage collecting the memory associated with multiple conversations. This approach maps nicely onto the state-free nature of the Web. Each request to the server is now a self-contained operation on an utterance-state closure. The disadvantage of the client-side state maintainance is the increased time required to transmit the dialog state. For a short conversation in a constrained domain, this is probably not significant, though it could become so in more complex situations. It also requires a more sophisticated client plugin and is further removed from the standard Web model.

In our current system, we have been investigating keeping the state on the server-side, despite the added inconvenience. The motivation assumes that browsers will eventually support audio input and output natively, and that we will be able to deliver conversational systems without requiring the download of a plugin.

4.3. Audio Response Delivery

In order to make conversational systems naturally interactive, it is desirable to have a spoken reply as well as a graphical display. Ideally, the spoken reply occurs simultaneous with the display update and without any need for user action (such as clicking a link). In our system, the spoken response is generated on the server side by a TTS system.[1] The resulting audio waveform file then needs to be downloaded and played in concert with the response page. There are several approaches to accomplishing this. We have chosen to embed a Java applet in the HTML of the response page which, in its initialization phase, downloads and plays an audio file (specified by a URL in its arguments). Since the lifetime of these audio files need only be a few minutes, the garbage collection overhead on the server is not significant.

4.4. Current Status

The development domain we have been using for this work deals with queries about movies currently playing in the Boston area. Example queries include:

- "What movies are playing at Kendall Square around 7 pm?"
- "What are they rated?"
- "Who is in Mission Impossible?"
- "When is The English Patient showing?"

This domain was chosen because it is of reasonable size and immediately intuitive to the average person. It is also easy to obtain and update the underlying movie database.

Initial versions of all of the conversational system components have been created and targeted to this domain. The demonstration system is deployed on servers on our network and provides real-time responses via a Web-browser based client interface.[2] In addition, we have collected a set of utterances from CRL employees and visitors for use in language model training and CSR evaluation.

5. Summary and Future Directions

The Web is evolving rapidly. The approaches and solutions described here are directed to delivering speech-based applications within the current constraints to Web technology. As the technology infrastructure evolves, our system will naturally follow and some of the approaches presented here may become obsolete.

One expected development is the definition of a standard interface for interacting with audio I/O from the browser environment. This may involve Java support for audio or native support within the browser.

Another development likely to affect our architecture is the integration of binary component technology and mobile code into the Web, creating the so-called Object Web. There are many entrants in this area, including DCOM and ActiveX, JavaBeans, and CORBA. This distributed/mobile object environment will undoubtedly enable new means of delivering speech-based applications.

6. REFERENCES

1. Hemphill, C. and Baker, C., "Accessing Information by Voice on the World Wide Web," *Proceedings of Avios'96*, San Jose, CA, September, 10-12, 1996.
2. House, D., "Spoken-Language Access to Multimedia (SLAM)", Masters Thesis, Oregon Graduate Institute, 1995.
3. Hurley, E., Polifroni, J. and Glass, J., "Telephone Data Collection Using the World Wide Web", *Proceedings ICSLP*, pp. 1898-1901, Philadelphia, PA, October, 1996.
4. Julia, L., Cheyer, A., Neumeyer, L., Dowding J., and Charafeddine, M., "http://www.speech.sri.com/demos/atis.html", *Proceedings of the AAAI'97*, Stanford, CA, March 1997.
5. Bayer, S., "Embedding Speech in Web Interfaces," *Proceedings of ICSLP*, pp. 1684-1687, Philadelphia, PA, October, 1996
6. Sokolov, M., "Speaker Verification on the World Wide Web," Submitted to *Proceedings Eurospeech '97*, Rhodes, Greece, 1997.
7. Goldenthal, W., Waters, K., and Van Thong, J-M, "Driving Synthetic Mouth Gestures: Phonetic Recognition for FaceMe!," *Proceedings Eurospeech '97*, Rhodes, Greece, 1997.
8. Goddeau, D., Brill, E., Glass, J., Pao, C., Phillips, M., Polifroni, J., Seneff, S., and Zue, V., "GALAXY: A Human-Language Interface to On-line Travel Information", *Proceedings ICSLP*, pp 707-710, Yokohama, Japan, 1994.

[1] Although client-side synthesis is certainly possible, we have chosen to minimize the requirements on the client. When TTS is bundled with every browser or becomes universally distributed on client machines, we may switch to client-side synthesis.

[2] Currently, the system is not accessible outside of CRL.