Voice-Enabling Your Web Sites
Kimberlee Kemble

This article introduces VoiceXML language, discusses how it fits into the current Web environment, and describes how you can combine VoiceXML with speech recognition and speech synthesis technologies to voice-enable your Web site.

Introduction

Here's your chance to learn about VoiceXML. This article will introduce you to the VoiceXML language, discuss how it fits into the current Web environment, and describe how you can combine VoiceXML with the technologies of speech recognition and speech synthesis to voice-enable your Web site.

If you've developed any kind of Web application in the last several years, you're already familiar with HTML. You probably know about XML. But do you know what VoiceXML is? Hopefully, you've heard of it, and have been wondering what it is and how you might use it.

What is VoiceXML?

VoiceXML stands for Voice eXtensible Markup Language, and it is an XML-based markup language for creating distributed voice applications, much as HTML is a markup language for creating distributed visual applications.

The VoiceXML language was originated by the VoiceXML Forum, which is an industry organization founded by AT&T, IBM, Lucent, and Motorola (www.voicexml.org). The Forum is chartered with establishing and promoting VoiceXML as a standard for making Internet content and information accessible via voice and telephone.

"Okay," you say, "Now what does that really mean?" Well, it means that VoiceXML allows you to develop Web-based applications that users can access by telephone -- using their voice. Instead of having to be connected to the Internet and tied to a computer and a mouse, a user can call into your Web site and access the very same information and services, simply by talking to a VoiceXML application over the phone.

Now you're probably thinking, "Okay, so why do I want to provide voice access to my Web site?" The answer to that question is quite simple. By providing voice access to your Web site over the phone, you can provide your customers with virtually anytime, anywhere access to your site. Think about it. The Web has raised customer expectations that business information and services
are available 24 hours a day, seven days a week, 365 days a year. But, you have to be tied to a computer and connected to the Web to access these services. Not everyone has a computer, and even those who have a computer aren't always connected. However, almost everyone has a phone (or two or three...). And in the ever mobile world that we live in, most people don't leave home without their phones, so they've essentially got their Web access devices with them at all times. Talk about convenience and availability!

**The Extended Web World**

VoiceXML is designed to extend the existing Web environment by providing another way of accessing Web information and services. With VoiceXML, you use your voice and a telephone to access the Web instead of a computer and a mouse.

There are many similarities between the visual (HTML) Web world and the audio (VoiceXML) Web world. For example, in the visual world, you use a Web browser to access the Web; in the VoiceXML world, you use a VoiceXML Browser. Web browsers present information to the user through HTML; Voice Browsers present information to the user through VoiceXML.

Every programming language has a Hello World! example. VoiceXML is no different. Here is what it might look like in VoiceXML:

```xml
<?xml version="1.0"?>
<vxml version="1.0">
<form>
  <block>
    Hello World!
  </block>
</form>
</vxml>
```

The first statement `<xml>` is probably very familiar to you. It must be the first statement in every VoiceXML document. The next element is `<vxml>` which tells the XML interpreter that this is a VoiceXML document. A VoiceXML document is essentially a container for **dialogs**. There are two types of dialogs: **forms** and **menus**. Forms present information and gather input; menus offer choices of what to do next. The "Hello World!" example uses a single `<form>` element, which contains a `<block>` that synthesizes the text "Hello World!" and presents it -- that is, reads it -- to the user. At this point, the dialog ends.

And these similarities are by design. The primary goal of VoiceXML is to bring the power of Web development and content delivery to voice applications. It was designed to provide a way for Web developers to use a familiar markup style and existing Web server-side logic to deliver voice content to the Internet. If you know HTML or WML, VoiceXML is going to look very familiar to you.

**The VoiceXML Browser**

In the world of VoiceXML, you interact with a Web site over the phone using a VoiceXML Browser. The VoiceXML Browser is analogous to a graphical Web browser, such as Netscape® Communicator and Microsoft® Internet Explorer. It is the way you interact with a Web server using
your voice and a telephone. Instead of rendering and interpreting HTML (like a graphical browser), the VoiceXML Browser renders and interprets VoiceXML. Rather than clicking a mouse and using your keyboard, you use your voice and a telephone (and even the phone keypad) to access Web information and services.

One of the primary functions of the VoiceXML Browser is to fetch VoiceXML documents from the Web server, just like a graphical Web browser fetches HTML documents. The request to fetch a document can be generated either by the interpretation of a VoiceXML document, or in response to an external event. The VoiceXML Browser uses HTTP over a LAN or the Internet to fetch the documents (the very same HTTP requests that are used by the graphical Web browser).

The VoiceXML Browser interprets and renders the VoiceXML document. It manages the dialog between the application and the user by playing audio prompts, accepting user input, and acting on that input. The action might involve jumping to a new dialog, fetching a new document, or submitting user input to the Web server for processing.

**Architecture**

Let’s take a look at how VoiceXML and the VoiceXML Browser fit into the current Web environment. We’re all very familiar with the Web as it works today. You use a graphical Web browser (such as Netscape Communicator or Internet Explorer), which renders and interprets HTML to present information to the user (text, graphics, audio, hyperlinks, etc.). When the user makes a selection (for example, a click on a hyperlink), the graphical Web browser sends an HTTP request to the Web server (in this case, to retrieve another page). The Web server responds by locating the new page and returns HTML to the browser to present the new page to the user. The Web server may also have to interact with a back-end infrastructure (database, servlets, etc.) to obtain and return the requested information.

The VoiceXML Browser extends this paradigm. You’ll notice a telephone and a **Voice Server** have been added to the Web environment. For the purposes of this article, a Voice Server is an abstraction. It is an entity that contains the VoiceXML Browser, the speech recognition software, and the text-to-speech software.
To be able to "talk" to a Web site and have a Web site "talk" to you requires some underlying voice technologies, not just a programming language (VoiceXML). These technologies are **speech recognition** and **speech synthesis**.

Speech recognition is a software component that translates spoken input into text. An application can then do something with that text. For example, if a caller said "checking account," the application could retrieve the caller's current checking account balance and tell her what it is.

Text-to-speech, on the other hand, is a voice technology that converts text into spoken output. In the previous example, the application could "read" the checking account balance to the caller using text-to-speech.

VoiceXML introduces a new way of presenting the very same Web information. Now, instead of presenting the information visually (through HTML, graphics, and text), the VoiceXML Browser presents the information to the caller using VoiceXML. When the caller says something (which is the voice equivalent of clicking on something to make a selection), the VoiceXML Browser sends an HTTP request to the Web server, which may access the VERY SAME back-end infrastructure, to return information -- this time in VoiceXML to the user.

When the VoiceXML Browser is started, it sends an HTTP request over the LAN or Internet to request an initial VoiceXML document from the Web server. The requested VoiceXML document can contain static information, or it can be generated dynamically from data stored in an enterprise database using the same type of server-side logic (CGI scripts, Java Beans, ASPs, JSPs, Java servlets, etc.) that you use to generate dynamic HTML documents.

The VoiceXML Browser interprets and renders the document. Based on the user's input, the VoiceXML Browser may request a new VoiceXML document from the Web server, or may send data back to the Web server to update information in the back-end database. **The important**
thing is that the mechanism for accessing your back-end enterprise data does not need to change: your VoiceXML applications can access the same information from your enterprise servers that your HTML applications do.

Voice-Enabling the Web

There are two very important concepts to keep in mind:

• Voice-enabling the Web doesn't mean throwing away the graphics from a traditional graphical Web page and reading the rest of the information aloud. That probably would not be very useful. What it does mean, though, is providing a different way of accessing the same information and services. Even though you are providing the same information and services as you would with a graphical browser, you probably need to change the way you present this information. For example, you may be able to show a list box with 30 items in it using a graphical browser, but you probably don't want to read 30 items to the caller over the phone. The key point is, you are changing the presentation of the information, not the information or how it's generated (by the Web server and the back-end).

• Voice isn't always the best user interface for an application. There are some applications that are just more suited for a visual medium. And that's OK. For example, it may be quite acceptable to purchase a music CD or a book over the phone, but you might not want to purchase a $300 cashmere sweater without being able to see it and feel it first. However, a customer might find it very useful to check the status of their order over the phone.

With that said, most Web applications can be voice enabled. What you have to do as an application designer is to decide what kind of information to provide, how much information to provide, and how and when to present it to the user.

Conclusion

When you add VoiceXML, speech recognition, and text-to-speech to the Web environment, you can easily provide voice access to Web content. VoiceXML fits into the existing Web paradigm and allows you to leverage your substantial Web developer skills base by adding a new dimension to your Web site.