Building Voice Applications from Web Content

César González-Ferreras[1] and Valentín Cardeñoso-Payo[1]

Universidad de Valladolid, Valladolid 47011, Spain
`{cesargf,valen}@infor.uva.es`,
WWW home page: `http://www.infor.uva.es/~cesargf/`

Abstract. Using voice to access on-line information from the web would be really useful, because of the proliferation of mobile devices which allow Internet access anytime and anywhere. However, vocal interface is sequential and not persistent, and thus, we have to restructure the information in order to achieve an efficient and natural way of interaction. Our proposal is based on converting original web contents into VoiceXML dialogs, using VoiceXML templates and extraction rules written in XSLT. Our system has two main components: a development tool to build voice applications and a transcoding server to access them. We have identified five typical HTML patterns and designed a way to browse them using voice.

1 Introduction

Internet is an information repository whose size and popularity increases everyday, and where there is a great diversity of contents. All that information is created to be accessed using a web browser (visual interaction), and thus, the focus is on visual appearance. However, enabling other modes of interaction could enrich the user experience and could be more suitable for some kind of users (blinds) and some environments (eyes-busy). In fact, vocal interaction is more natural to most of the people.

Nowadays, using voice as a means of communication with a computer is achieving a high maturity, because of the development of spoken dialog systems which allow a friendly interaction with the user in natural language [1, 2]. However, traditional spoken dialog systems are designed to access database information, and they need to be adapted to access textual information.

Browsing Internet contents using voice is becoming more and more important, mainly because the spread of mobile devices which allow web access anytime and anywhere. However, the task is not easy, because a restructuration of the information is required to adapt it to the new modality, very different from the visual one: vocal interaction is sequential and not persistent. We can not present all the information at once, like in a traditional web browser, we have to dialog with the user in order to give him only the information he wants. The lack of metainformation describing web contents makes more difficult the conversion, because content authors emphasize visual appearance instead of structuring the contents properly.

In this paper, we present a system which allows browsing web content using voice. The system reuses HTML information, converting it into VoiceXML pages that can be accessed using a VoiceXML browser. First of all, a voice application has to be created, describing how the conversion has to be done for each HTML page. To this end, we use VoiceXML templates and extraction rules written in XSLT language. We have created a development tool which helps in building voice applications. Five typical HTML patterns have been identified and we provide for each of them a way of accessing the information using voice. Once the application is built, we use a transcoding server to access the information using voice.

The structure of the paper is as follows. Section 2 presents some related work. Section 3 explains five frequent HTML patterns and their voice counterpart. Section 4 describes our system. Section 5 shows a study case and section 6 some conclusions.

2 Related Work

There are several approaches to access web content using voice. The first one creates a different version of the contents using VoiceXML [3], a language designed to bring the advantages of web-based development and content delivery to interactive voice response applications. The main drawback is the maintenance of several versions of the same contents.

Other approaches add a vocal interface to an existing web browser, using voice commands instead of mouse and keyboard, [4, 5]. In order to have a voice output, a screen reader can be used to present the contents of the web page to the user [6].

Another option is to convert the original HTML contents into VoiceXML, like in [7, 8]. Both systems try to extract the structure of the web documents and use it to dialog with the user. However, given the inherent characteristics of the voice channel, transcoding simplified versions of HTML pages could be a better solution, [9]. A similar approach is given in [10], where well structured XML documents are converted into VoiceXML.

Finally, the solution could be restricted to a limited domain, like in [11, 12], where the dialog system works for selected on-line resources. This allows the designers to develop ad hoc dialogs in order to achieve a more efficient and user-friendly interaction. Moreover, the information could be stored in a database, which allows to use traditional spoken dialog systems.

3 Patterns

Although there is a huge diversity and variability in web content over the Internet, some patterns are frequently used to structure information. In our work, we have identified five typical HTML patterns. For each of those patterns we have designed a way of browsing their contents from a vocal application. The use of those patterns helps us to automate the development of voice applications to access HTML on-line information.

The selected patterns can be seen in table 1. It shows the name of each pattern, its characteristics and the way we propose to access its contents using voice. We selected those five patterns because they are the most frequent ones found in HTML pages.

Table 1. Most frequent HTML patterns and their associated voice interaction

Pattern	**Characteristics**	**Voice Interaction**
Text	Textual information divided in sections. Each section has a title and a body.	Prompt a message describing the content of all sections. The titles of all sections are prompted and the user selects one. Then, the body of that section is prompted to the user.
Table	Information structured in rows and columns. Typically used to describe objects: each object is a row in the table, and the columns are its properties.	Prompt a message describing the content of the table. The user selects one object of the table (one row) using one of the properties (column). Prompt all the information associated with that object.
Form	Several fields of information to be filled by the user. Each field has a text which describes that field. There is a submit button, which sends all the information to a CGI program.	Prompt a message describing the content of the form. The user is required to fill each of the fields. When the form is completed, all the information is submitted to the CGI program.
List	A set of text elements, each presented in a different line and starting with a symbol or a number. There is the possibility of nested lists.	Prompt a message describing the content of the list. Prompt each element of the list, in the same order of the original list. If there is a nested element, the user can select to browse it or to continue with the next element of the same level.
Navigation Bar	A set of links grouped like a menu. Usually found in the left side or in the top of the page. Each link has a text describing the content of the target page.	Prompt a message describing the navigation bar. All the elements of the navigation bar are prompted and the user selects one of them. The interaction continues in the URL of the selected item.

In the following section we explain in detail how our system makes the conversion from HTML to VoiceXML.

3.1 Conversion of Contents

The core of our system is the conversion of HTML pages into VoiceXML ones. The aim of the conversion is twofold: to select the elements of the original HTML page to be used in the vocal application and to describe how to transform the HTML code into VoiceXML.

The conversion is carried out in several steps, as can be seen in figure 1. We use a semi-automatic conversion scheme, in which the developer has to specify how the conversion must be done, providing two XML files:

- **VoiceXML template**: contains the structure of the resulting VoiceXML page. It has references to extraction rules, which will provide the final content of the page.
- **Extraction rules**: these rules are used to select elements from the HTML page and to transform them into VoiceXML. These rules are written using XPath and XSLT.

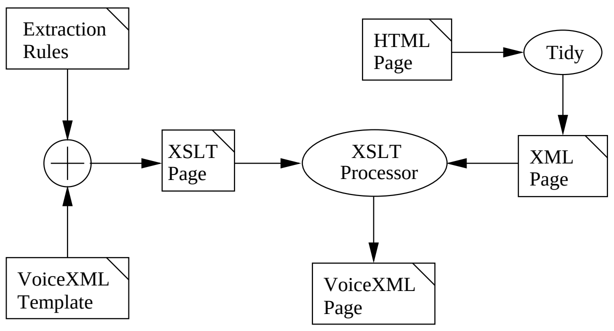


Fig. 1. Transformation of a HTML page into a VoiceXML one

First, the HTML page is transformed into a XML page, using Tidy [13]. Next, the VoiceXML template and the extraction rules are joined, generating a XSLT page. Finally, the XML page with the original information is transformed, according to the XSLT page, using a XSLT processor to produce the resulting VoiceXML page.

4 System Overview

Our main objective was to reuse existing HTML information, allowing to access web contents using voice, instead of visual interaction. We selected VoiceXML as a language to describe dialogs, so the system converts HTML code into VoiceXML, allowing to access the information using any off-the-shelf VoiceXML browser.

The system is composed of two main components: development tool and transcoding server, as can be seen in figure 2. The system uses a semi-automatic approach to do the conversion: first, a developer has to create a voice application, using the *Development Tool*, specifying how the conversion has to be done for each HTML page. Next, that application is deployed on the *Transcoding Server*, where the application can be accessed by the users.

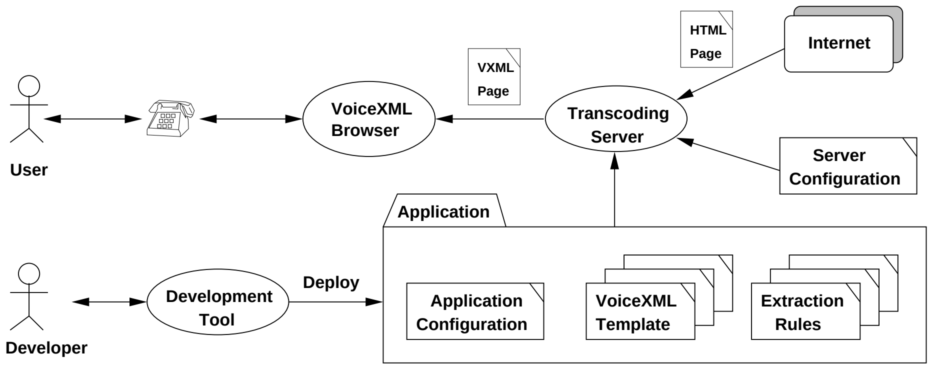


Fig. 2. System architecture

It is not possible to use all the information of the original web content, because of the limitations of the voice channel. The developer has to first select which elements to use and later he has to design a way to convert them into VoiceXML. Once the application is built, it can be used several times, even if the information changes, because the conversion of contents is done on the fly (provided that there is no change in the structure of the page).

A voice application specifies how each HTML page is converted into VoiceXML. It consists of three main components: a set of VoiceXML template files, which contain the skeleton of the resulting VoiceXML files; a set of Extraction rules files, which describe how to extract and transform the information from HTML pages; and an application configuration file, which defines for each URL which template and rules files have to be used.

In the next section we describe the development tool. Next, we describe the transcoding server, and finally, the VoiceXML browser.

4.1 Development Tool

The development tool allows developers to build voice applications from web content and deploy them into the transcoding server.

First we have to select the pages to process from Internet. For each new URL added to the application, a VoiceXML template and extraction rules have to be created. This can be done using a wizard, if the HTML content matches one of the predefined patterns (see section 3), or manually in any other case.

Our system helps developers to create VoiceXML templates, just by selecting options from the menu. When creating extraction rules, XPath expressions have to be built to select elements from the original web page. Using our tool developers can build such expressions pointing with the mouse in the desired element of the page. Once the template and the rules are created, the system applies them to the original web page, allowing developers to preview the result, helping them to test and debug applications. Finally, the application can be deployed in the transcoding server.

4.2 Transcoding Server

The transcoding server converts HTML pages into VoiceXML ones. The conversion is done on the fly, upon request from the VoiceXML browser. When the server receives a request, it also receives the URL to convert as a parameter. The application configuration file specifies which template and rules have to be used to convert this URL. Using them, the original web page is converted into VoiceXML, as described in section 3.1.

We implemented the Transcoding Server as a Java Servlet, in order to communicate it with the VoiceXML browser in a standard way, i.e., using the HTTP protocol. We used Tomcat as servlet container. The server uses a configuration file to describe in which directory applications are deployed.

4.3 VoiceXML Browser

A VoiceXML browser allows users to browse information using voice. It parses VoiceXML pages and dialogs with the user using speech synthesis and speech recognition over the telephone line. The main advantage of using VoiceXML is that it is a standard, so the voice applications can be accessed using off-the-shelf technology.

In our system, any VoiceXML browser can be used to access the information. We have tested the system using our VoiceXML platform, which is composed of: our own VoiceXML interpreter; speech synthesis and speech recognition engines developed at *Universidad Politécnica de Cataluña*; and a *Dialogic* telephone card.

5 Case of Study

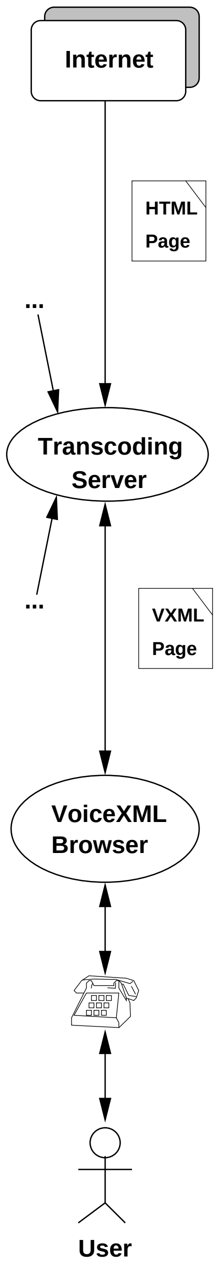
In order to show how the system works for a given HTML page, we include an example. We used a web page from *Yahoo!* which gives information about the Dow Jones Industrial Average Index.[1] In figure 3 we show the original HTML page, the VoiceXML page generated by our system and a sample interaction with a user.

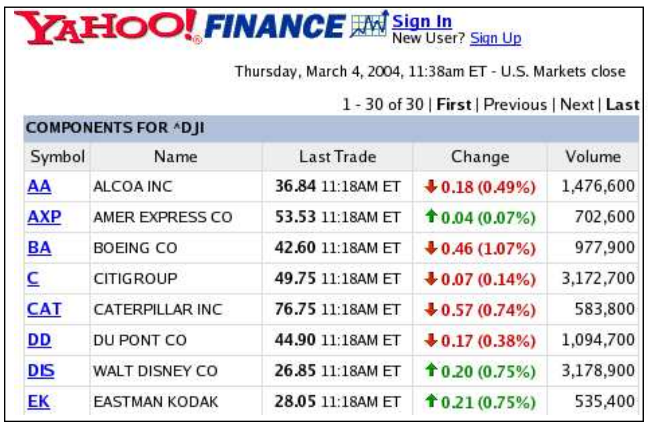
6 Conclusions

Voice browsing of Internet contents is really useful, mainly because of the proliferation of mobile devices which allow access to the web anytime and anywhere. However, a restructuration of the original contents must be done to adapt it to the new modality, very different from the visual one.

In this paper we have presented a system which allows voice browsing of web contents using VoiceXML. The system is based on a transcoding server, which converts HTML pages into VoiceXML ones, reusing existing information found

[1] http://finance.yahoo.com/q/cp?s=^DJI




YAHOO! FINANCE Sign In
New User? Sign Up

Thursday, March 4, 2004, 11:38am ET - U.S. Markets close

1 - 30 of 30 | **First** | Previous | Next | **Last**

COMPONENTS FOR ^DJI

Symbol	Name	Last Trade	Change	Volume
AA	ALCOA INC	**36.84** 11:18AM ET	↓0.18 (0.49%)	1,476,600
AXP	AMER EXPRESS CO	**53.53** 11:18AM ET	↑0.04 (0.07%)	702,600
BA	BOEING CO	**42.60** 11:18AM ET	↓0.46 (1.07%)	977,900
C	CITIGROUP	**49.75** 11:18AM ET	↓0.07 (0.14%)	3,172,700
CAT	CATERPILLAR INC	**76.75** 11:18AM ET	↓0.57 (0.74%)	583,800
DD	DU PONT CO	**44.90** 11:18AM ET	↓0.17 (0.38%)	1,094,700
DIS	WALT DISNEY CO	**26.85** 11:18AM ET	↑0.20 (0.75%)	3,178,900
EK	EASTMAN KODAK	**28.05** 11:18AM ET	↑0.21 (0.75%)	535,400

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0">
<form id="start">
  <block>Welcome to Dow Jones Industrial Average Index.
  <goto next="#main"/></block>
</form>
<menu id="main">
  <prompt>Please select a company.</prompt>
  <choice next="#ALCOA_INC">ALCOA INC</choice>
  <choice next="#AMER_EXPRESS_CO">AMER EXPRESS CO</choice>
  <choice next="#BOEING_CO">BOEING CO</choice>
  <choice next="#CITIGROUP">CITIGROUP</choice>
  ...
</menu>
<form id="ALCOA_INC">
  <block>Last Trade: 36.84 11:18AM ET<goto next="#main"/></block>
</form>
<form id="AMER_EXPRESS_CO">
  <block>Last Trade: 53.53 11:18AM ET<goto next="#main"/></block>
</form>
<form id="BOEING_CO">
  <block>Last Trade: 42.60 11:18AM ET<goto next="#main"/></block>
</form>
<form id="CITIGROUP">
  <block>Last Trade: 49.75 11:18AM ET<goto next="#main"/></block>
</form>
...
</vxml>
```

System: Welcome to Dow Jones Industrial Average Index. Please select a company.
User: CITIGROUP
System: Last Trade: 49.75 11:18AM ET. Please select a company.
...

Fig. 3. Conversion of Dow Jones Industrial Average Index page from Yahoo! and sample interaction

on Internet. A development tool is also provided to build voice applications, which describe how the conversion of contents has to be made. Finally, we have identified five typical HTML patterns and have proposed a way to access them using voice interaction. This helps developers on creating voice applications.

As future work, we plan to make an evaluation of the system performance and a usability study. This will allow us to validate the adequacy of the voice interaction proposed for each HTML pattern. We will study how users respond to the system and, as a result, we will have a better understanding of the Vocal User Interface.

7 Acknowledgments

The authors want to thank Jorge Olfos-Martínez for his collaboration in the development of some parts of the software used in this work. This work has been partially supported by *Consejería de Educación de la Junta de Castilla y León* under project number VA083/03 and by *Ministerio de Ciencia y Tencología* under project number TIC2003-08382-C05-03.

References

1. Lamel, L., Rosset, S., Gaubain, J., Bennacef, S.: The Limsi Arise System For Train Travel Information. In: ICASSP. (1999)
2. Zue, V., Seneff, S., Glass, J.R., Polifroni, J., Pao, C., Hazen, T.J., Hetherington, L.: JUPITER: A Telephone-Based Conversational Interface for Weather Information. IEEE Transactions on Speech and Audio Processing (2000)
3. W3C Voice Browser Working Group: Voice eXtensible Markup Language (VoiceXML) Version 2.0 (2004) <http://www.w3.org/Voice>.
4. Hemphill, C.T., Thrift, P.R.: Surfing the Web by Voice. In: ACM International Conference on Multimedia. (1995)
5. House, D.: Spoken-Language Access to Multimedia (SLAM): A Multimodal Interface to the World-Wide-Web. Master's thesis, OGI (1995)
6. Vesnicer, B., Zibert, J., Dobrisek, S., Pavesic, N., Mihelic, F.: A Voice-driven Web Browser for Blind People. In: Eurospeech. (2003)
7. Lamb, M., Horowitz, B.: Guidelines for a VoiceXML Solution using WebSphere Transcoding Publisher (2001) <ftp://ftp.software.ibm.com/software/wtp/info/VxmlTranscodingGuide.pdf>.
8. Goose, S., et al.: Enhancing Web Accessibility Via the Vox Portal and a Web Hosted Dynamic HTML & VoxML Converter. In: Int'l WWW Conf. (2000)
9. Freire, J., Kumar, B., Lieuwen, D.F.: WebViews: Accessing Personalized Web Content and Services. In: Int'l WWW Conf. (2001)
10. Araki, M., Ono, T., Ueda, K., Nishimoto, T., Nimi, Y.: An Automatic Dialogue System Generator from the Internet Information Contents. In: Eurospeech. (2001)
11. Lau, R., Flammia, G., Pao, C., Zue, V.: WebGalaxy - Integrating Spoken Language And Hypertext Navigation. In: Eurospeech. (1997)
12. Polifroni, J., Chung, G., Seneff, S.: Towards the Automatic Generation of Mixed-Initiative Dialogue Systems from Web Content. In: Eurospeech. (2003)
13. Raggett, D.: Clean up your web pages with HTML TIDY (1999) <http://www.w3.org/People/Raggett/tidy/>.