SCIENTIA PLENA **VOL. 3, NUM. 6** **2007**
www.scientiaplena.org.br

Challenges of voice-enabling Web content

Hendrik T. Macedo

Departamento de Ciência da Computação, Universidade Federal de Sergipe, 49100-000, São Cristóvão-SE, Brasil

hendrik@ufs.br

(Recebido em 26 de agosto de 2007; aceito em 23 de outubro de 2007)

Mainstream Internet usage is growing ripe into a pervasive set of services people use as common tools in everyday life, like cellular phones and a sort of different wearable gizmos. Voice-based access enhances Web accessibility by enabling the physically and visually impaired community. Unfortunately, ubiquitous browsing lacks a consensual delivering format which overcomes the physical constraints, memory and processing limitations of such wearable gizmos. In addition, delivering to such devices often occurs under time-pressure and content should be customized to user and context. For all these reasons, hypertext looses much of its benefits. In this paper we discuss the main differences between visual and voice-based access to Web content and the challenges of providing customized vocal interfaces to intrinsically visual oriented Web applications. This paper also provides some guidelines towards a solution to the problem. The key ideas concern the use of natural language generation, voicexml and Web services.

Palavras-chave: vocal interfaces, web content, natural language generation, voicexml.

1. INTRODUCTION

In the emerging ubiquitous computing scenario, increasing the practicality and profitability of Web content critically involves making it more personalized as well as available anytime and anywhere. This, in turn, requires content delivery that is intelligently customized to the user's profile and preferences inferred from her past interaction with the system, as well as to the particular interaction context of the current request. This context includes a variety of factors such as the device used to access the service, time constraints on its delivery and the purpose behind the user's information request. Definitely, it is no longer essential how the information is presented but how easily people can access it.

When Web content is delivered only through desktop computers, the unique presentation medium of HTML is generally sufficient. Thanks to its highly mnemonic visual clues and its three dimensional structure, hypertext drastically simplifies interaction with the user, content selection and content organization. The interaction with the user is menu, link and click-driven, avoiding natural language processing. Content selection and Content organization are essentially left to the user through navigation. So, a single document can satisfy different information needs, from different user classes in different contexts.

Ubiquitous delivery, however, can no longer rely on these simplifying assumptions. On devices such as PDAs, mobile phones and upcoming wearable gizmos, hypertext looses much of its benefits. In addition, delivery to such devices often occurs under time-pressure. Thus, from one medium to the next, content from a given provider must not only be presented in different formats, but also selected and organized in a different way for each different format, user and context. Users expect customized information to be delivered quickly and accurately. As a consequence, single content selection and single selected content presentation format for same request by different users at different times using different platforms is definitely no longer a feasible approach.

Providing vocal capabilities to dynamic Web content generation is mandatory. Voice-based access is not hindered by the small keypads and displays of cell phones and PDAs, allowing for hands and eyes free access to Web content, which is particularly important for navigating while executing a manual task. The visually or physically impaired community greatly benefits from such kind of Web access. The voice-enabled access to Web content brightens up the awkwardness of such physically impaired user to handle the mouse and keyboard. Likewise, a visual or physically impaired person benefits from synthesized speech to find out about on-line information [1].

However, generation of voice-based content requires a quite different approach from that traditionally used to generation of hypertext documents. One of the most high-end requirements for voice-enabled Web applications is that they should convey unambiguous, short and clean chunks of information to avoid user's misunderstanding.

In this paper, we make a judicious analysis of the key differences between voice and visual browsing interfaces. We also discuss the challenges of voice-enabling exceedingly HTML-encoded Web sites. Next, we explain why Natural Language Generation (NLG) technique [2] can be used to support scalable automated generation of voice-browsing dialogs and how NLG and modern Web engineering technologies could be harmoniously integrated in an abstract software architecture to enable such generation. At last, we present some concluding remarks.

2. VOICE-BASED BROWSING

A natural trend for accessing Web content is made by means of multimodal interfaces. Indeed, there are numerous ongoing efforts to define standard markup languages targeted at specific interface media: HTML for desktop browsers, WML for mobile device displays, and VoiceXML for speech interfaces. The W3C has already defined multimodal requirements for voice markup languages [3].

Although many attempts have been made, until present, a unique, uniform, expressive and optimized markup language has not yet emerged. Currently, there are two feasible alternative approaches to turn Web content accessible by multiple devices. Content providers may present services for different devices developing each application service for each set of presentation requirements at design time. In other words, if one wants its documents be accessed by common desktop users as well as by cell phone users or landline phone users, thus, it must provide different versions of the same content in each HTML, WML and VoiceXML format. Hence, each page contains a set of links to content in the appropriate form for the respective device. Although there are many editors to facilitate the implementation of such pages, the burden of creating, testing, and maintaining different versions of each application for different presentations can become rapidly impractical as the number of devices increases. This approach provides both quality of presentation, since each page is constructed and hand-tailored directly by human designer, and good performance, since all the work is done at design time which reduces run-time processing required to generate dynamically information in one format to another.

An alternative approach is to generate different versions of the same content, on the fly. The developer stores the data in a database and builds an application to dynamically produce HTML, WML and VoiceXML documents. In this case, data is modeled as XML trees and subsequently XSL transformations may be applied to produce the respective documents. The specific script used to fetch the corresponding transformations to be applied is chosen at runtime based on information in the HTTP headers of the incoming request (figure 1).

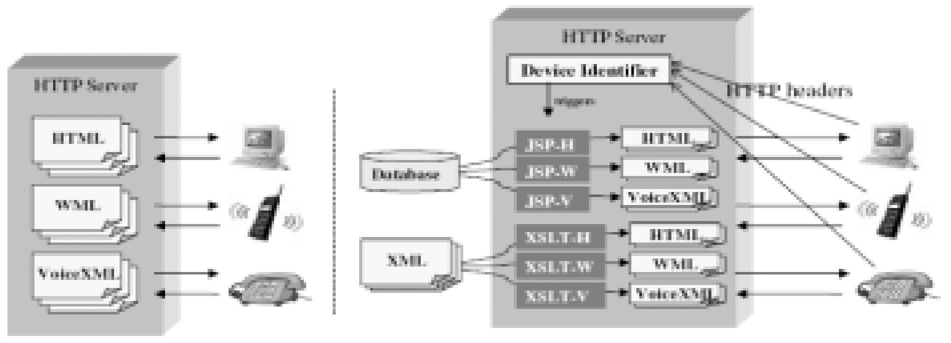


Figure 1: Two feasible approaches to turn Web content accessible by multiple devices: static and dynamic generation of documents.

2.1 From Visual to Voice-enabled

The approaches just described may be applied in the case of building voice-enabled (or even multimodal) application from scratch. VoiceXML may be used to build a voice-based Web site from scratch similarly to what is done with visual HTMLencoded web site; VoiceXML has the support of IDEs, script language, tutorials and books to application developers.

The question is: what about voice-enabling already existent HTML-encoded web sites?

Although the usefulness and business value gain that a voice interface could provide to Web systems and services, building a mechanism that enables such an automated translation of these mainstream HTML-encoded systems to a particular voice format, such as VoiceXML is definitely not a trivial task. In fact, several issues arise during the design of a mechanism for voice enabling hypertext-based applications: How to adequately determine the information we want to voice-enable, since a significant part of the content semantics is implicitly conveyed by the layout navigational structure and other features existent at the presentation layer? How to reorganize and adjust the information to vocal media in such a way to keep thematic coherence and similar browsability? In fact, on-the-fly content translation is a very challenging activity due to the existence of intrinsically visual features there are hard or even impossible to translate or to adequately represent in a voice format, such as tables and images, for instance.

A document generated for spoken presentation purposes may bear a different content and structure from one to visual presentation. Conciseness, precision and unambiguity are high-end requirements in speech generation in order to avoid overpowering the user's short-term memory [4]. Also, speech takes time to be uttered.

A judicious analysis of some web statistics may point out the most appropriate approach to voice-enabling a web-based site: what is the average time a Web page remains without any content change? What is the average time between succeeding Web page's modifications? What is the average amount of modifications in pages? How many web pages lays out of graphical content? In some cases, the modification rates of the application contents are so low that would be more suitable providing all appropriate formats at design time avoiding the hard of implementing a profitable translating tool. Likewise, performing content adaptation dynamically is inadequate for pages full of graphical content. On the other hand, if content updating is frequent, the burden of constantly hand-rewriting different documents formats may be untenable.

2.2 Differences between Vocal and Visual Browsing Interfaces

Voice interfaces imply a thoroughly different set of conventions and expectations for the end user than traditional visual interfaces. There are intrinsically visual elements that can hardly be conveyed in a voice application. Also, the elements lying beyond the document, such as advertisements and banners, get in the way of the user to get to the core content easily. Much less data can be represented in a vocal document than in a visual one. Furthermore, information in visual page are not time-dependent and are available as long as it is desired; the end user owns the control of the navigation: he/she may quickly move back through previous pages or start up two or more browser's windows pointing to different pages. In a visually presented web page, many different images, tables and the like can be presented on the screen at the same time, in a spatial format that is quickly and effectively processed by the human perceptual system.

In contrast, voice based information is transient just like any human-to-human spoken conversation. Once the user has access to a voice application and has heard the content, that content is gone and is hard to get back to it. In addition, it is usually more laborious to give heed to audio information than to visual ones. A study conducted by Karl et al. [5] has shown that subjects had more difficulty memorizing symbols when commands to manipulate those symbols where issued by voice than when they where issued by keyboard or mouse.

Thus, it is crucial for voice applications to limit the amount of information presented to the user and reduce the complexity of the data, avoiding overpowering the user's short-term memory. Another challenge is the provision of user's consistent navigation and error recovery mechanisms. It is often difficult for the end user to know where she is in a voice application and possibly, to lose focus of the subject matter. The content presentation must provide concise, yet descriptive and representative links for the current area of the application. Only closely related information should be conveyed. Unlike visual applications where error recovery is often simple to be provided, voice applications are more error prone. They must be designed in such a way to make it easy to retry an input, or limit the user's choices for input to minimize the risk of mismatch errors. A text input must be limited to a well-understood and bounded grammar, or specification of valid input. Error supporting mechanisms must be carefully worked out to enable a trustworthy navigation. The use of pre-recorded report messages such as: "I didn't understand what you have said. Say it again", "Would you like to listen to it again?", "For going back say …", "For getting to the main say…", "Continue?" provide secure aural navigation landmarks. Such mechanisms also avoid the time wasted in listening to everything in a page, for example, when one is not interested to do so.

Interestingly, content limitations of voice navigation may have positive side effects to visual web-pages design as well. In order to facilitate interface convergence (e.g, multimodal web), content providers will become more and more aware about the amount and complex of their data. This forces application design to become more centered on the content structure than on the presentation structure.

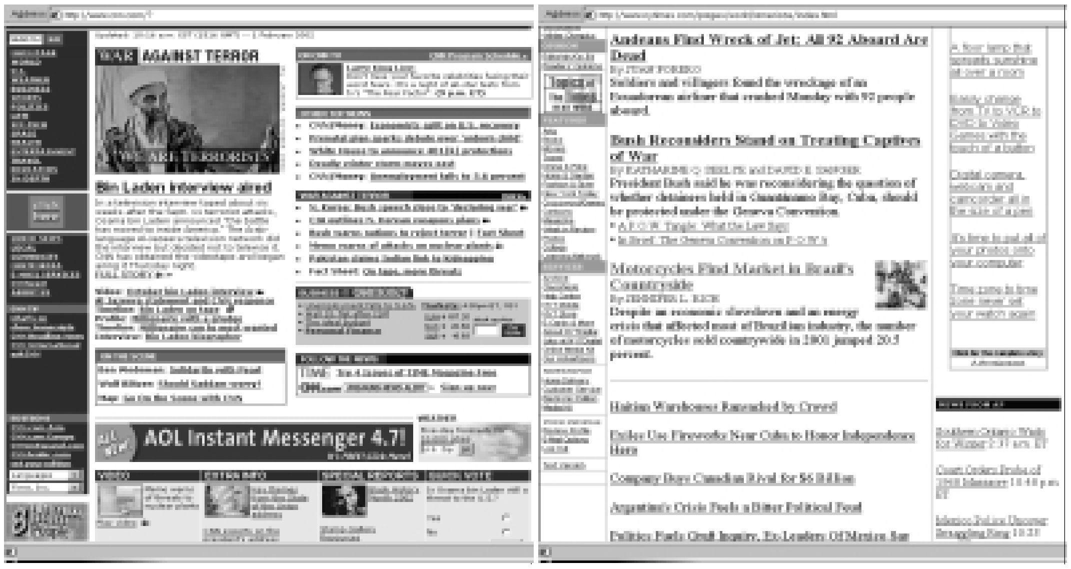
Christian et al. [6] have made an interesting study to discover whether navigating the Web by voice is a viable alternative to traditional mouse-based navigation. They have tested three browsing modes - by mouse, by voice with text links, and by voice with numbered links – for three common web navigation patterns: the hierarchical menu, the linear slide show, and a two dimensional panning map (no zooming). For the hierarchical-tree style menu, the task tested was that of finding specific information beginning the search at the root page. For the slide show, it was navigating through the slides (which had numbered links) and relaying to the administration the number (that had been randomly generated previously for each slide) on the target slide. Finally, for the panning map, it was moving the "frame of focus" around the landscape; for example, "Starting from Detroit, following the red line, what is the name of the destination city located at the end of the line?" The adopted metrics were the time to correct completion, error rate (missed commands and misinterpreted commands) and subjective satisfaction as overall impression, ease of navigation, ease of learning, etc. The volunteers were subjects with no previous voice browsing experience.

The results showed that voice control takes about 50% longer for navigation intensive tasks and fewer penalties for lighter navigation tasks. An interesting result was that the three browsing styles took similar completion time for the panning map navigation tasks. Numbering links adds cognitive overhead, thus, it is important to avoid numbered links. Moreover, designers must use short, easily spoken links that are distinct when spoken. The results also show that more robust techniques for form based data entry and URLs are needed.

2.3 Challenges of Voice-enabling HTML-encoded Web Sites

The considerations taken above illustrate the complexities involved in voice-enabling standard visual pages. Figure 2 shows ordinaries HTML documents.

The page on the left is quite complex. In spite of this complexity users can easily assimilate the document structure at a quick glance through the visual layout and landmarks that implicitly carry much of the context. Rendering such document by simply sending the raw text to a Text-To-Speech engine (TTS engine) most of the context would be lost. In addition, there is no way to voice-enable some of the elements listed above (images, banners, etc..). A dynamic translation framework must be capable of identifying the type of the page, drawing out the main subjects, presenting them in a reasonable order, and providing good navigational support for avoiding user's loose of track.



Bin Laden interview aired

Andeans Find Wreck of Jet; All 92 Aboard Are Dead

Soldiers and villagers found the wreckage of an Ecuadorean airliner that crashed Monday with 92 people aboard.

Bush Reconsiders Stand on Treating Captives of War

President Bush said he was reconsidering the question of whether detainees held in Guantánamo Bay, Cuba, should be protected under the Geneva Convention.

Motorcycles Find Market in Brazil's Countryside

Despite an economic slowdown and an energy crisis that affected most of Brazilian industry, the number of motorcycles sold countrywide in 2001 jumped 20.5 percent.

Haitian Warehouses Ransacked by Crowd

Exiles Use Fireworks Near Cuba to Honor Independence Hero

Company Buys Canadian Rival for $6 Billion

Argentina's Crisis Fuels a Bitter Political Feud

Figure 2: Online newspapers.

Identifying the type of the page means to find out if it is a search page, a news page, a personal page, a service page, and so on. A variety of techniques has been proposed or can be used for such task. For instance, to know whether a document is primarily an index or primarily information, we can measure the link density of the document. One way is to simply count the absolute number of links contained in a document, which is problematic because it may unfairly characterize long, mostly content documents. An alternative measure is to count the number of links per unit of text, or links per kilobyte of text, but again, this may mischaracterize documents which are mainly content, but which have been diligently hyperlinked to related pages.

Another way is to ask whether or not most of the information in the document consists of links to other documents by calculating the number of characters in the document that are contained within link anchors, as a percentage of the total number of characters. A final concern is that hyperlinks are sometimes anchored on bitmaps, rather than text, and thus are not counted towards the link density in the previous heuristic. Accounting for such links is done by counting them as χ characters, both toward the number of characters within anchors, and toward the total number of characters, where the value of χ can be determined empirically. Client-side image maps containing η links count as $\eta^* \chi$ characters.

Likewise, the media content may be analyzed to assist the page's classification: how many images, animations or audio clips are there in a document? Graphics intensive pages may indicate more commercial documents, on-line catalogs may be categorized by the presence of images or audio, and media content may indicate documents which may be more or less suitable for rendering in particular environments. A useful distinction is between images which actually convey information and images which are simply ornaments of the page structure, such as dividing bars, fancy list bullets, and simple link icons, based on the number of pixels.

The accuracy of such classification may be improved by learning from previous translation experience. For instance, instead of having to spell carefully the name of a desired URL, the user might say a keyword or a set of them and, based on an internal database, the system would find the related URL and prompt the desired page. Similarly, for a casual access, the engine could advertise the user with some page's characteristics.

A Web page may be reached fortuitously or intentionally. It is considered as intentional when the user has already previous knowledge of the page's subject, the type of the page. The other kind of access will be labeled as "fortuitous access". This may work upon the way end-users would like to listen to the content of such a page and, thus, on the way the translator engine will work. For instance, an assiduous user of the newspaper illustrated before could be interested only on “sports” or “weather” news, so it would be boredom for him/her to listen to the whole headlines at first. For a fresh user, however, the second approach is more appropriate.

Thus, when accessed for the first time, the translator could render a “welcome” voice document, in which would be listed some navigation’s patterns. The user, then, could choose listening to the whole document sequentially, listening the headlines, doing an in-depth navigation, or issuing a keyword. In order to organize a smooth speech presentation flow, such a translator’s algorithm should traverse the HTML document to identify paired tags (figure 3).

For instance, consider the page on the left in figure 2. It is relatively easy to remark that the HTML source document is structured by means of tables with embedded images, links, and raw text. Also, there are forms, banners and, likely, some script code.

In figure 3 (a) the algorithm must notice that there is a table with a list of links. Since the list is composed by small sized words, it is assumed to be a list of sections of the document outlined. In (b), the algorithm is supposed to identify that there is an linked image and, based on the attributes “height” and “width” of the tag <IMG>, that such image actually conveys information. There are three possible ways to render such information: (1) ignoring the image, (2) supposing it is related with the following text link (when there is one), and (3) trying to spell the phrase or words of the link. For newspaper’s pages at least, the second option is usually the most recommended. In (c), the algorithm would find another table with an embedded image and a list of links. The image has a very low dimension, so may be inferred that it is just an ornament of the page structure, and ignore it so. Afterwards, it finds another list of links, but unlike the previous list, the hyperlinked text is composed by many words: they are not sections, but links to more information, maybe. Finally, (d) shows a form that is preceded by a row text and a link. The algorithm may notice that after the text of the link there is a “:”, which means that the next close set of tags is related to it. Thus, the translator may render as so, prompt the form for the user and submit the response.

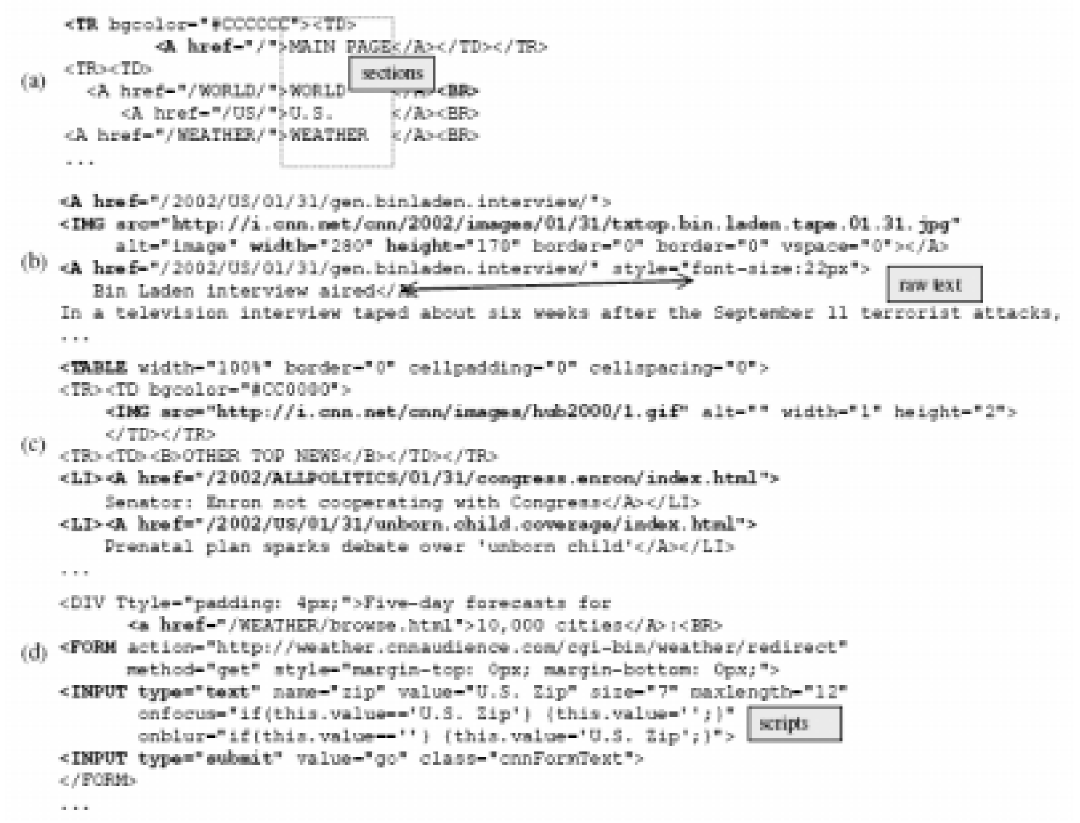
Depending on the kind of the page and the user’s choice, the algorithm must perform some little “adjustments” to enhance performance. When heading tags <h1>, <h2>, etc. are found, the algorithm may organize the document in sections and subsections based on these tags. However, there are a lot of authors that use a large font or underlined texts to customize the appearance of a section heading instead of using the standard HTML header construct. Thus, the algorithm must identify such behavior to determine the author’s actual intent (see figure 3 (b)).

When the user decides for listening to headlines, a suitable design pattern to voice enabling the content is to organize the whole document as a list of links and read it sequentially until the user choose one of the links or quit the application. For instance, the page on the right in figure 1 includes some hyperlinked texts followed by an excerpt of the subject it treats. In such case it is easier to organize the voice document. However, many times there are not hiperlinked words, as we have already commented. In that case, the algorithm may choose the first words of the texts and create a hyperlink for the whole text.

It is extremely important to provide commands for users do not loose track of the context. “Help”, “Quit”, “Back”, or “Main” commands should always be active. After each section or group of links, it is important to provide commands such as “Repeat”. Prerecorded audio for publicity, for example, may be skipped with a “Skip” or “Jump” command.

Unfortunately, designers of such a translator framework must consider several other issues as ill-formed HTML documents, end-users preferences and profiles, and technology limitations. They must also decide *about how general will be the translator? Will it embrace all kinds of pages? Only index pages? Only documents without multimedia elements? Only textual documents?*

In fact, for textual-only documents, mainstream screen reader systems, which employ speech synthesis technology, are very satisfactory. Moreover, some HTML elements are hard to voice-enable: since voice applications are one dimensional, there is no good concept of a table representation in voice. The algorithm may attempt to convert the table in a list, for instance. There are also elements that are difficult to speak like “acronyms” (for instance, WYSIWYG stands for “what you see is what you get”) and “spelled URL links” - not so hard, we find the text of an anchor tag to match its associated HREF URL link).



```
    <TR bgcolor="#CCCCCC"><TD>
            <A href="/">MAIN PAGE</A></TD></TR>
    <TR><TD>
(a)   <A href="/WORLD/">WORLD     </A><BR>
         <A href="/US/">U.S.      </A><BR>
    <A href="/WEATHER/">WEATHER   </A><BR>
    ...

    <A href="/2002/US/01/31/gen.binladen.interview/">
    <IMG src="http://i.cnn.net/cnn/2002/images/01/31/txtop.bin.laden.tape.01.31.jpg"
         alt="image" width="280" height="170" border="0" border="0" vspace="0"></A>
(b) <A href="/2002/US/01/31/gen.binladen.interview/" style="font-size:22px">
       Bin Laden interview aired</A>
    In a television interview taped about six weeks after the September 11 terrorist attacks,
    ...

    <TABLE width="100%" border="0" cellpadding="0" cellspacing="0">
    <TR><TD bgcolor="#CC0000">
        <IMG src="http://i.cnn.net/cnn/images/hub2000/1.gif" alt="" width="1" height="2">
        </TD></TR>
(c) <TR><TD><B>OTHER TOP NEWS</B></TD></TR>
    <LI><A href="/2002/ALLPOLITICS/01/31/congress.enron/index.html">
        Senator: Enron not cooperating with Congress</A></LI>
    <LI><A href="/2002/US/01/31/unborn.child.coverage/index.html">
        Prenatal plan sparks debate over 'unborn child'</A></LI>
    ...

    <DIV Ttyle="padding: 4px;">Five-day forecasts for
          <a href="/WEATHER/browse.html">10,000 cities</A>:<BR>
(d) <FORM action="http://weather.cnnaudience.com/cgi-bin/weather/redirect"
          method="get" style="margin-top: 0px; margin-bottom: 0px;">
    <INPUT type="text" name="zip" value="U.S. Zip" size="7" maxlength="12"
           onfocus="if(this.value=='U.S. Zip') {this.value='';}"
           onblur="if(this.value=='') {this.value='U.S. Zip';}">
    <INPUT type="submit" value="go" class="cnnFormText">
    </FORM>
    ...
```

Figure 3: Excerpts from the HTML source for the web page on the left in figure 2.

3. TOWARDS A SOLUTION

A great issue to consider when building such a translator is how to effectively customize at reasonable cost answers to the same query to different devices, different user profiles and different user purposes? The correct identification of the target audience, for instance, is an important prerequisite for the success of a vocal site. It is mandatory, for example, to investigate the types of callers. Each type of caller has some distinct characteristics that affect their assessment of an application. Although some of them may have prior experience with voice applications, most of them are beginners. As a consequence, dialogs should be clear and effective, and an exhaustive and effective help system should be provided as well. In contrast, experienced users usually expect a good dialog design with clear and to-the-point information in prompts.

Serving different age groups of callers is also another important factor, including some humorous dialogs to amuse kids, while avoiding over consuming short-term memory of adults is a good policy. Furthermore, voice personalities are more appropriate to some types of applications than to others. A financial application, for example, might sound authoritative and trustworthy, whereas an entertainment center application might sound pleasant and funny.

The solution to this issue must rely on three important aspects. The first aspect regards the separation of concerns. The information content, the queries to that content and the final presentation format should be treated separately in order to allow application maintenance independence and reuse. Actually, a change in the presentation format should not compel content modification.

A second aspect regards portability. Seamless port to different platforms is highly required. Definitely, the solution should not be tied to specific platform current technologies.

The third aspect regards the provision of a fine-grained content, service and presentation descriptions in order to allow a maximization of the reuse of common elements between the different answers to the issue. Higher granularity of the content description enables higher degree of flexibility in generating sentences, phrases and other textual structures, which enhances the possibilities of customization to different user profiles and purposes, for instance.

3.1 Key Ideas

We think that Natural Language Generation and Web services [7] satisfy such requirements.

Web Services are a versatile framework for building distributed information systems through the assembly of components that are independently deployed on the web. The Web services allow components to be published, discovered, and invoked over the network. This new services-based architecture hides the underlying technology of a component, allowing an application to use the service without having to understand the underlying technology. The calling application needs only to know what a service does, how to use it, and where to find it. This loose coupling of Web services allows applications that rely on that service to continue to run regardless of implementation changes.

Natural Language Generation enables the production of natural language text from computer-internal non-linguistic knowledge sources that represents the information about the domain. The output text may range from a single clause to an entire multi-sentential document. Based on a high-level communicative goal, a natural language generator decides which information from the underlying content source should be communicated in the output document. During the generation process, this high-level goal is refined into more concrete goals and some sorts of planning are employed to progressively convert them into the final linguistically well-formed text. NLG technology is particularly attractive to contemplate the third aspect.

Once we have identified the technologies that attend the prerequisites, we need to provide a way to integrate them in synergy. The key ideas of such integration is to provide an abstract mediator architecture that uses NLG technology to support scalable, automated generation of highly customized voice browsing dialogs and Web services to integrate speech gateways with the NLG component and Web applications.

3.2 Supporting Scalable Automated Generation of Voice-browsing Dialogs

VoiceXML allows fine tuning the level of user initiative in a dialog to the size of computational grammars and lexicons of non-prohibitive development cost. It also allows packaging all the related content for a variety of information needs into a single document, leaving the user selecting and ordering its exposure to the relevant units through voice navigation choices. Automating dialog generation using VoiceXML is thus far closer to the task of hypertext generation [8], than to that of classical natural language question-answering interfaces focused on sentence interpretation [9]. In fact, previous work in NLG [10] consider hypertext as a simplistic form of dialog. These works states that a hypertext-based content can actually be seen as an interactive system in which the user request some information, and then receive from the system the core of the requested information together with some anchors to complementary information. Hence, a hypertext document may be seen as a kind of dialog between the user and the information provider, where a hypertext click is a metaphor for a user request, to which the system responds.

Using such metaphor and considering VoiceXML as the potential target language format for providing a vocal interface to existing Web contents, an important generation issue is the establishment of a correspondence between specific interface markup elements (e.g., menus, links, and forms) of HTML and VoiceXML languages. Opportunely, VoiceXML provides the HTML key elements counterparts and supports the representation of multi-sentential dialog-based spoken-oriented documents. VoiceXML thus precisely fits the requirement to specify the voice-based dialogs that forms the basis of the interaction with such a user.

Due to its emphasis on generating a coherent, multi-paragraph discourse, hypertext generation and thus VoiceXML dialog generation must rely on the use of the complex but scalable NLG technology.

This NLG approach to mediate between a web content provider and a voice portal is entirely new. Previous work [11] proposed instead on a machine translation approach that reused a classical dynamic HTML generation infrastructure, together a generic HTML to VoiceXML translation system. We believe that even for mildly sophisticated information needs, voice-browsing differs too radically from visual browsing as a sensory and cognitive experience for such an approach to robustly produce voice-browsing dialogs that are effective enough for practical applications.

A potential solution to the problem of vocal-enabling Web contents is based on full-fledged generation of VoiceXML dialogs. This means that VoiceXML should not be used as a shallow interface to a NLG-based dialog system that generates prompts in turns according to user inputs [12]. The generated VoiceXML document must implement all possible dialog alternatives, avoiding time-consuming extra natural language generations.

3.3 Web Services to Integrate Speech Gateways with NLG and Web Applications

Mediators promote loose coupling by allowing self-governing components to interact. The mediation here consider three distinct components: (1) Web Content Provider (WCP) is a third-part Web application hosted in a web server which provides information to be consumed by another application, (2) Voice Service Provider (VSP) is a kind of speech gateway which encapsulates all the required hardware and software infrastructure to interpret VoiceXML documents, generate speech from text and automatically recognize speech, and (3) the NLG component working as the mediator itself, which is hosted in a specific Web server. It is in charge of providing a voice-driven interface to WCP and, thus enabling information to be consumed by ubiquitous users.

To integrate the NLG mediator with the two other components it connects in the most general and platform independent way, we can rely on Web services.

One advantage of the use of Web services is that Web services' SOAP messages communicate XML, which is semantically stronger than HTML-based documents. Some works have already made use of Web services together with VoiceXML. In [13], for instance, a spoken dialog architecture is proposed to control virtual electronic devices with the mediation of Web services. Each device provides the dialogue interface knowledge for each service; the dialog interface is realized by VoiceXML. The purpose of the project described in [14] is to provide ubiquitous access to a Web service, via telephone or cellphone. The author implements a Restaurant Finder Web Service which provides list of restaurants along with their addresses upon user's request. VoiceXML is used to achieve interaction between the Web service and the telephone. The methods of the WebService are invoked trough the VoiceXML document and the restaurants' results generated by the Web service is an XML file, which is parsed in VoiceXML using the Document Object Model (DOM). However, the work does not detail how this process is done.

A possible mediation scenario is to have the NLG component working as the provider of services for third part Web applications that wish for a voice-enabled interface.

As a service provider, a NLG component can publish methods for allowing Web applications to set their contents or processing results. Additionally, it publishes a method which returns the final dialog the NLG application produces from these Web application contents or processing results. This dialog is a VoiceXML-encoded document that can be accessed by a simple user application running on the VSP. Figure 4 illustrates the generic component-based architecture for this mediation scenario.

A VSP usually provides the required API to interface with SOAP oriented Web services. A simple VoiceXML application in the VSP makes use of this SOAP API to invoke the Web service. The BeVocal voice portal, for instance, use the SOAP API illustrated bellow to create a proxy object on the fly, which, in fact, invokes the methods of such a Web service.

```
Bevocal.soap.serviceFromWSDL (
String WSDLUrl, {URL of WSDL file describing the Web service}
String port, {port name of the service}
String svcNamespace, {service namespace}
String svcName, {service name}
String endpointURL {SOAP endpoint of the service}
   )
```

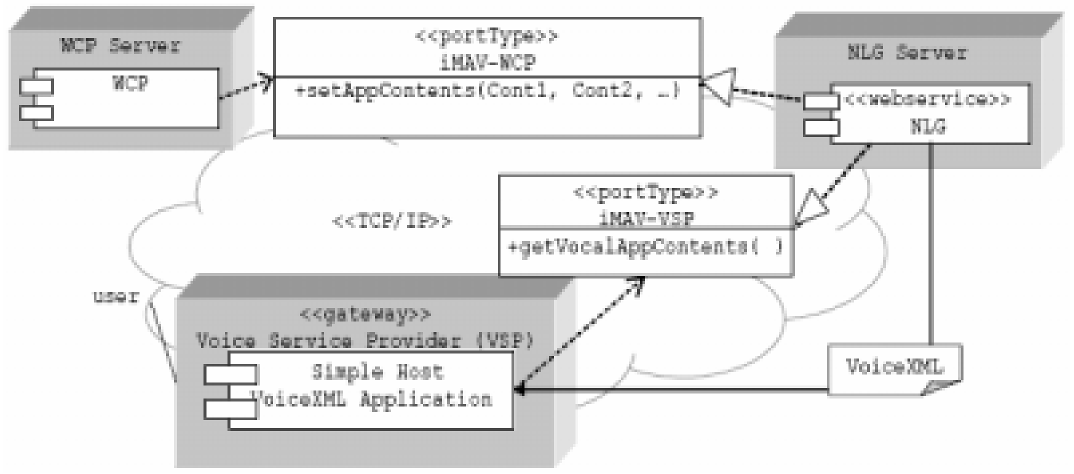


Figure 4. NLG component as a service provider for WCP

The code bellow illustrates how a NLG Web service called GenericNLG could be invoked using BeVocal as the VSP.

```
var service =
bevocal.soap.serviceFromWSDL("http://www.nlgserver.com
/genericnlg.jsp?WSDL", //WSDL file
"GenericNLGSoap", //port name
"http://www.nlgserver.com/", //service namespace
"GenericNLG" //service name
);
```

4. CONCLUSION

Wearable devices enhance Internet accessibility by enabling information delivery anytime and anywhere. It is however constrained by the little size of the screens and the keyboard to navigate the Web space. Voice-based access to Web content is a natural trend to overcome the limitation. The benefits of a voice-based access are extended to the traditional access from personal computers. Additionally, the voice technology can lead Internet to ordinary telephony. Hypertext and its visual clues simplify interaction with the user, content selection and content organization. Nevertheless, the mainstream Web content management approaches so long used to generate hypertext documents, do not deal properly with the complexities involving the provision of vocal utterances. Voice-based content generation for wearable devices must also consider, for instance, time-pressure constraint. In this case, not only the presentation format is important, but the content must be selected and organized in a different way for each type of user and context. In this paper we have discussed the key differences between visual and voice-based and the challenges involved in providing vocal interfaces to mainstream Web applications. The analysis has shown the hardness of automating a coherent delivering format translation. We thus outline a solution that is founded on three important aspects: the separation of information content from the final presentation format, the portability, and the provision of a fine-grained content, service and presentation descriptions. We show that these can be achieved by the synergistic usage of technologies such as Natural Language Generation and Web services.

1. RODRIGUES, A.: *Acessibilidade na Internet para Deficientes Visuais*. Dissertação de Mestrado, UFRN, Brasil (2000)
2. REITER, E. AND DALE, R.: *Building Natural Language Generation Systems*. Cambridge University Press (2000)
3. W3C.: EMMA: *Extensible MultiModal Annotation markup language, Working Draft*, http://www.w3.org/TR/2003/WD-emma-20031218/ (2003)
4. MCKEOWN, K., PAN, S., SHAW, J., JORDAN, D. AND ALLEN, B.: Language Generation for Multimedia Healthcare Briefings. *Proceedings of Applied Natural Language* (1997)
5. KARL, L., PETTEY, M. AND SHNEIDERMAN, B. Speech-Activated versus Mouse-Activated Commands for Word Processing Applications: An Empirical Evaluation, Intl. J. Man-Machine Studies (1993)
6. CHRISTIAN, K., KULES, B., SHNEIDERMAN, B. AND YOUSSEF, A. A Comparison of Voice Controlled and Mouse Controlled Web Browsing. *ASSETS 2000*, Arlington, VA (2000) 72-79
7. CERAMI, E. Web Services Essentials. O'Reilly and Associates (2002)
8. FAVERO, E. AND ROBIN, J. HYSSOP: Natural Language Generation Meets Knowledge Discovery in Databases. *IIWAS'2001*, Linz, Austria (2001)
9. DALE, R., MOISI, H. AND SOMERS, H. (Eds.). Handbook of Natural Language Processing. Marcel Dekker (2000)
10. LEVINE, J., CAWSEY, A., MELLISH, C., POYNTER, L., REITER, E., TYSON, P. AND WALKER, J.: IDAS: Combining Hypertext and Natural Language Generation. *Proceedings of the Third European Workshop on Natural Language Generation*, Innsbruck, Austria (1991) 55-62
11. ANNAMALAI, N.; GUPTA, G.; PRABHAKARAN, B. Accessing Documents via Audio: An Extensible Transcoder for HTML to VoiceXML Conversion. *Proceedings of ICCHP* (2004).
12. MITTENDORFER, M, NIKLFELD, G AND WINIWARTER, W.: *Evaluation of Intelligent Component Technologies for VoiceXML Applications*. Technical Report, Software Competence Center Hagenberg (SCCH) and The Telecommunications Research Center Vienna (FTW) (2001)
13. ARAKI, M. Spoken Dialogue Agent Architecture for Web Service Mediator. *Proceedings of First International Workshop on Language Understanding and Agents for Real World Interaction* (2003)
14. BOPARDIKAR, K. *Implementation of Voice, Web and Wireless Interfaces to a Web Service.* Research Project. Division of Computing Studies (DCST) of the College of Technology and Applied Sciences on Arizona State University's (ASU) Polytechnic campus. Mesa, Arizona.