A design of the transcoder to convert the VoiceXML documents into the XHTML+Voice documents

JIEUN KIM, JIEUN PARK, JUNSUK PARK, DONGWON HAN
Computer & Software Technology Lab, Electronics and Telecommunications Research Institute,
Kajung-Dong 161, Yusung-Gu, Daejon, KOREA 305-350

Abstract: - Whereas HTML is commonly used for creating graphical web applications, VoiceXML is used for voice-enabled web applications. But those applications that contain a big data or interaction a single-modality are not suited for small and mobile devices. The multimodal markup language like the XHTML plus Voice(X+V) is used for multimodal applications that are composed of VoiceXML-based voice applications and XHTML-based visual applications. It can overcome the limitations of current voice browser and mobile devices by the efficiency of visual display and ease of speech input.
And work to maintain different application documents that are formatted to specific devices on the same web service is not economic. Therefore effort to reuse exiting application documents is necessary.
In this paper, we propose a new transcoder with an efficient algorithm to convert exiting VoiceXML-based applications into corresponding multimodal applications. The transcoder consists of the voiceXML parser module for fetching VoiceXML documents, the VoiceXML-to-X+V converter module for transforming a VoiceXML tree into a X+V tree on various rules, and the X+V generator module for editing X+V documents about the X+V tree. As a result, all the people who uses telephone and the X+V Browser can access VoiceXML-based services. A service provider can reduce cost and time that are in production and maintenance.

Key-Words: - VoiceXML, Multi-modal interaction, XHTML+Voice, transcoder

1 Introduction

The emerging world without wires has fostered a growing number of small and mobile devices (everything from personal digital assistants to smart phones) capable of accessing data and running applications. Mobile information access and remote transactions (for example, top news stories, weather, sports, and stock quotes) are fast commonplace. However, as devices become smaller, modes of interaction other than keyboard and stylus are a necessity. For example, mobile phones have relatively a small visual display and a cumbersome keypad input. PDAs have better the visual display, but have the same input limitations [1, 2].
But the web application that was basically stored in the HTML (Hypertext Markup Language) format was originally designed for traditional desktop browsers. So, the HTML-based application is not suited for small and mobile devices. For example, to consider accessing the CNN's weather page through mobile devices, it is very difficult to display world maps and weather forecasts in the mobile device's small screen. It is even difficult and time consuming to enter texts through mobile device's key pad or stylus [3].
Also the web application that was stored in the VoiceXML (Voice eXtensible Markup Language) [4] format was originally designed for interactive voice browsers, particularly for the telephone. Voice is a good way that you can input quickly. But you must remember what kind of item there are, and confirm whether your speech input is valid or not repeatedly. As a result, this is inefficient for you using mobile devices that have a small screen.
In order to solve these problems, we consider the multimodal web application that was stored in the X+V (XHTML plus Voice) [5] for mobile devices. Like VoiceXML, X+V meets the user demand for a voice-based interaction in small and mobile devices. Unlike VoiceXML, X+V uses both a voice-based and a visual-based interaction. For example, in a web browser on a PDA, you can select items by tapping or by providing spoken input. Similarly, you can use voice or stylus to enter information into stylus. Information can be displayed and spoken by mobile device [2].
But it is not economic to prepare different application documents that were formatted to specific devices on

the same web service. Therefore effort to reuse exiting application document is necessary.
In this paper, we propose a new transcoder with an efficient algorithm to convert the exiting VoiceXML -based applications to the corresponding multimodal applications. Section 2 introduces a conceptual overview to VoiceXML and X+V. Section 3 illustrates an architectural view of our transcoder, a converter algorithm, and an example that demonstrates the utility of the transcoder.

2 Background

Today, most web application developers use some type of markup language to code an application's user interface. The markup language for the user interface defines how the user can interact with the application.

2.1 What is VoiceXML?

VoiceXML is a markup language for creating voice-base web applications, and is based on earlier technologies from Motorola and IBM, and provides a standard interface between voice and the Internet. It uses the speech recognition and the touchtone (DTMF keypad) for input, and the pre-recorded audio and the text-to-speech synthesis (TTS) for output.
Instead of using a device with a web browser, any telephone (or cellular phone) can access VoiceXML applications via a VoiceXML "Interpreter" (also known as a " VoiceXML Browser") running on a telephony server.
One popular type of voice-based applications is the voice portal, a telephone service where callers dial a phone number to retrieve information such as stock quotes, sports scores, and weather reports. By separating application logic (running on a standard Web server) from the voice dialogs (running on a telephony server), VoiceXML and the voice-enabled web allow for a new business model for telephony applications. Other type of voice-based applications such as speech-controlled home appliances are starting to be developed [6].

2.2 What is X+V?

HTML was once the ruling standard for coding the graphic-based web applications, but in recent years it has been supplanted by XHTML [7]. Building an XHTML-based user interface typically involves laying out graphics, input fields, text prompts, check boxes, and so on. More sophisticated user interfaces might also include some type of scripting, such as JavaScript, to enable input checking and other minor computation or user-interface tasks. XHTML also focus on mobile devices such as PDA, wireless cell phone as well as PC.
X+V (XHTML plus Voice) is a markup language for developing multimodal applications. X+V uses XHML for visual interaction, a subset of VoiceXML (basically the <form> element and everything it contains) for voice interaction, and XML Events to correlate the two. And X+V defines some extension modules that include the <sync> element, the <cancel> element, the *src* attribute of the VoiceXML <prompt> element, and the *id* attribute of the VoiceXML <field> element for synchronization between the XHTML element and the VoiceXML element.
For example, you can request Internet search information from a travel site, enter the parameters for your flight, book your flight and hotel, and even rental car simply by speaking--which can be a far easier solution than manually typing your information, especially on handheld computers or mobile devices [2, 5].

3 The proposed Transcoder

3.1 Architecture

Our transcoder is to convert a VoiceXML document format into an X+V document format. But a VoiceXML document cannot be converted into an X+V document on sentence by sentence or node by node basis. So, we proposed appropriate algorithm for converting. Fig.1 shows components of our transcoder, and their relationship to each other.

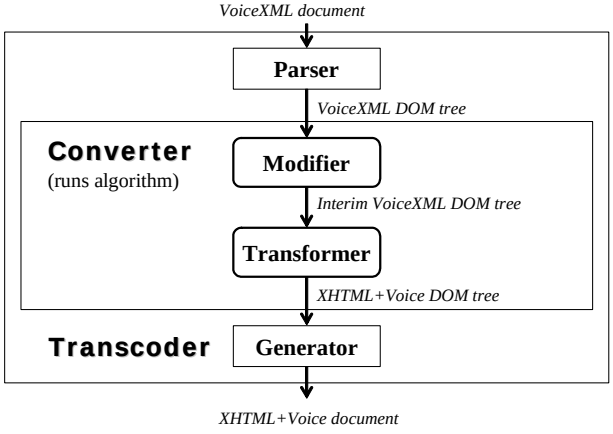

Fig.1 An architecture of the transcoder

The transcoder consists of:

- ✓ The VoiceXML Parser

- ✓ The VoiceXML-to-X+V Converter
  - ✧ The VoiceXML Tree Modifier
  - ✧ The VoiceXML-to-X+V Tree Transformer
- ✓ The X+V Generator

3.1.1 Transcoder Operation

The first thing that the transcoder does is to parse the VoiceXML document into a DOM (document object model) tree where each node in the tree is a node in the original VoiceXML. Then the VoiceXML-to-X+V converter operates. This phase converts the passed VoiceXML tree into a new X+V tree consisting of a XHTML tree and a set of VoiceXML tree. By the final step, the X+V generator produces an X+V document as output that corresponds with the X+V tree.

3.1.2 Converter Operation

Since VoiceXML is based on an XML document, the converter phase is basically a tree translation. The converter has two phases. The first is the VoiceXML tree modifier phase and the next is the VoiceXML-to-X+V tree transformer phase.

In the first phase, the VoiceXML tree is passed to the modifier as input. Using modifier algorithm, this phase remodels the VoiceXML tree into an interim VoiceXML tree.

In the second phase, the interim VoiceXML tree is passed to the transformer as input. Using transformer algorithm, this phase produces a new X+V tree consisting of a XHTML tree and a set of VoiceXML tree in the interim VoiceXML tree. The transformer extracts a node of a XHTML tree (for example, an <label> element) from a node of the VoiceXML tree (for example, a <prompt> element). And, it changes the interim VoiceXML tree to a transformed VoiceXML tree or trees. It links a node of the XHTML tree to a form of VoiceXML trees using XML Events and extension event modules.

3.2 Converter Logic

3.2.1 Modifier Algorithm

The modifier algorithm describes how to remodel a VoiceXML tree into an interim VoiceXML tree.

The modifier traces an original tree from top-level node to bottom-level node. A VoiceXML tree can contain one or more form nodes, and a form can contain various nodes.

A <prompt>, a <block>, a <field>, and a <grammar> element must belong to each one form. And the dependency of a variable must be removed or changed. Though a <menu> element is the same level with a <form> element, it must belong to one form, too.

3.2.2 Transformer Algorithm

The transcoder algorithm describes how to transform an interim VoiceXML tree into an X+V tree.

Fig. 6 is a block diagram showing an execution algorithm of the transformer roughly. Firstly, the transformer creates a new XHTML tree having sub-nodes that is composed of a <head> and a <body> element. Secondly, it appends a <form> element to a body node. Finally, it executes as following steps until all nodes in the interim VoiceXML tree are visited.

The prompt node that belongs to the form node is used to speak the content of a prompt node through the speech synthesis engine. Therefore the transformer appends a <p> element shown in a plain text to the XHTML tree. Both the block node that belongs to the form node and the prompt node that belongs to the menu node are equally run, too.

The submit node that belongs to the block node is used to deliver valuables to the linked application. Therefore the transformer appends a <input type="submit"> element shown in a push button to the XHTML tree. The submit node that belongs to the filled node is equally run, too.

The prompt node that belongs to the field node is used to speak a question through the speech synthesis engine, and to get a reply through the speech recognition engine. Therefore the transformer appends a <label> element shown in a plain text to the XHTML tree, and then appends a <input type="text"> element shown in an input field to the XHTML tree.

The choice nodes that belong to the menu node are used to select one among menu's items. Therefore, the transformer appends <a> elements shown in hypertext fields to the XHTML tree.

The one-of nodes that belong to the rule node are used to select one among grammar's items. Therefore, the transformer appends <input type="radio"> elements shown a bank of radio buttons to the XHTML tree.

In the <subdialog> node, the transformer jumps to the above block, and then a current-pointer node of a tree is changed to a top-level node of a subdialog tree.

In all case, the transformer has to define XML event modules of XHTML to invoke the form of VoiceXML documents, and extension event modules of X+V to synchronize both valuables of XHTML and valuables of VoiceXML. At the same time, some nodes of the interim VoiceXML tree are deleted or changed suitably.

3.3 Conversion Example

3.3.1 A basic VoiceXML document

To see how our transcoder works, Fig.2 shows a simple code of a Voice XML document.

If this speech dialog gets an input value, it executes the next file, "info.asp".

```
<vxml >
  <form id="choice">
  <field name="info">
    <grammar src="choice.grxml"
                  type="application/srgs+xml"/>
    <prompt> Choose from sports, weather,
               and news </prompt>
  </field>
  <block>
    <submit next="info.asp"/>
  </block>
 </form>
</vxml>
```

Fig.2 A code of VoiceXML document, " choice.vxml"

3.3.2 Parser and Modifier phase

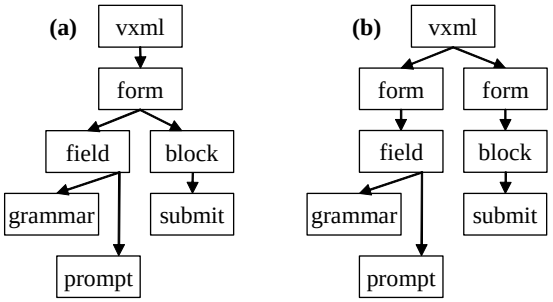

Fig.3 The VoiceXML tree: (a) an original tree, (b) an interim tree

Fig.3-(a) shows a VoiceXML tree about an original document. The top-level VoiceXML element can contain dialogs of ether <menu> or <form> elements. In this case, an original tree contains a form node. And the form contains a field node composing of a grammar node and a prompt node, and a block node including a submit node.

The modifier remodels an original tree into an interim tree with two forms such as Fig.3-(b) using the modifier algorithm.

3.3.3 Transformer phase

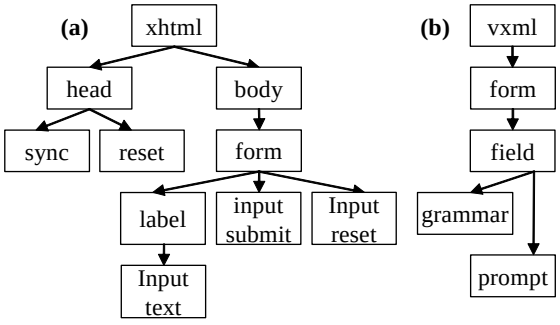

Fig.4 The X+V tree: (a) an XHTML tree, (b) a transformed VoiceXML tree

Fig.4 shows a XHTML tree that is created from an original tree, and a transformed VoiceXML tree that is modified from an original tree. The field node of VoiceXML is transformed into the label node and the <input type="text"> node of XHTML, and is linked to the form node of XHTML using XML events. The submit node of VoiceXML is transformed into the <input type="submit">, and is deleted with the above nodes at the last.

The sync node of X+V is used to synchronize both the field node of VoiceXML and the label node of XHTML. The reset node of X+V and the <input type="reset"> node of XHTML is used to cancel the voice-able mode.

3.3.4 Generator phase

Fig. 5 shows a display of X+V browser that executes a VoiceXML document, "choice.vxml" You can get a voice prompt as well as visual text.

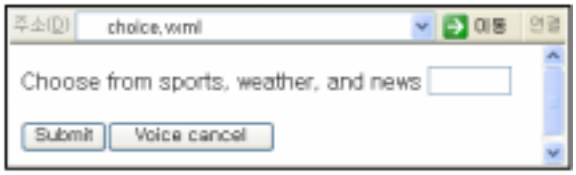

Fig.5 example execution in X+V Browser

The interaction would be as follow:

-You load the X+V browser in the mobile device and request "choice.vxml" via URL.

-The request is routed to the application web server.

-A VoiceXML application document (in this case choice.vxml) is delivered via HTTP at the transcoder.

-The transcoder transform a voiceXML document into an X+V document.

-The X+V browser interpreters the X+V document.

Our transcoder may be part of X+V Browser or be located in a network and running, for example, on another machine, proxy server, on a server of the content provider (server-side), or distributed on various machines in the network.

-The speech dialog is activated when you click an input field.

-The VoiceXML interpreter which belongs to the X+V browser instructs the speech synthesis engine to render a content of the prompt as audio. The speech output is the text, "Choose from sports, weather, and news"

-Then, it instructs the speech recognition engine to listen for the words, "sports, weather, or news"

-You have to tell your selection to system as voice.

-If you pushes a submit button, the X+V browser fetches the file, "info.asp" and the process continues.

4 Conclusions

In general, the exiting transcoders transformed HTML documents into VoiceXML documents [3, 8] and HTML documents into XHTML/WML documents for mobile handhelds devices. Though the multimodal interaction is necessity as devices become smaller, those still support a single-modality.

Multimodal applications that are generated by our transcoder, give you a choice of how they interaction with system, and reduce the overexertion that can result from single-modality interactions. And a service provider can reduce a development time and a maintenance cost via reuse of a code.

In this paper, we didn't explain how some elements convert. The call control elements like a <transfer> element can be replaced by some system calls or by outputs of another application. And ECMAscripts that are supported by VoiceXML can alternate with ECMA scripts that are support XHTML.

References:

[1] Harsha Srivatsa, *Multimodal applications:Another step in computer/human interaction*, Whitepaper of IBM develoerWorks, 2002.

[2] Les Wilson, *X+V is a markup language, not a Roman math expression*, Whitepaper of IBM developerWorks Whitepaper, 2003.

[3] Narayanan Annamalai, Thesis: An extensible transcoder for HTML to VoiceXML conversion, *The University of Texas at Dallas*, 2002.

[4] Scott McGlashan et al, *VoiceXML 2.0*, W3C CR available at: http://www.w3.org /tr/ voicexml20/, 2003

[5] Chirs Cross et al, *XHTML+Voice Profile 1.1*, W3C available at: http://www.ibm.com/software/ pervasive /multimodal/x+v/11/spec.htm, 2003.

[6] Kenneth G. Rehor, *What is VoiceXML?*, The VoiceXML Review, 2001.

[7] Altheim, Shane McCarron, *XHTML 1.1*, W3C available at: http://www.w3.org/TR/xhtml11/.

[8] Zhiyan Shao et al, Transcoding HTML to VoiceXML Using Annotation, *ICTAI*, 2003, pp.434 -452.

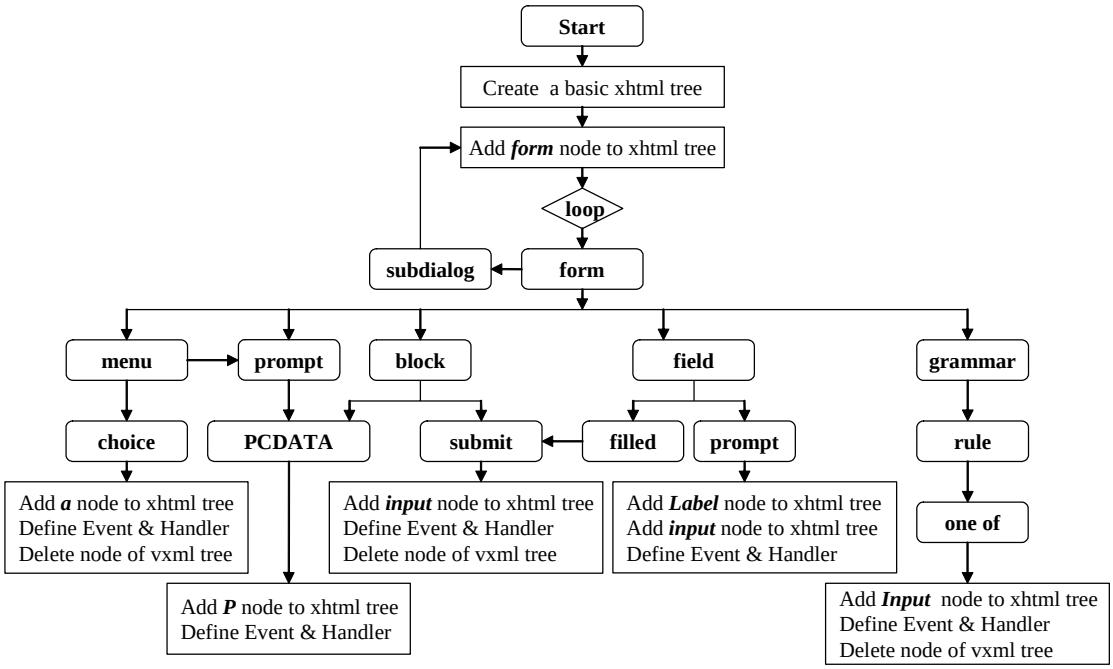

Fig.6 An execution algorithm of the transform in VoiceXML-to-X+V Converter