ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2005, Vol. 34, No. 2

SALT – MARKUP LANGUAGE FOR SPEECH-ENABLED WEB PAGES

Rytis Maskeliūnas, Kastytis Ratkevičius, Algimantas Rudžionis

Speech Research Laboratory, Institute of Information Technology Development,
Kaunas University of Technology
Studentų St.65-108, LT- 51369, Kaunas, Lithuania

Vytautas Rudžionis

Department of Informatics, Kaunas Faculty of Humanities, Vilnius University
Muitinės St. 8, LT- 44280, Kaunas, Lithuania

Abstract. Activities to develop Lithuanian Web pages for speech-enabled access from telephone are presented. After the integration of Lithuanian text-to-speech synthesizers "Aistis" and LtMBR to SAPI 5 (Speech Application Programming Interface), some Lithuanian speech-enabled Web pages were prepared (www.speech.itpi.ktu.lt). Two methods of SALT (Speech Application Language Tags) technology implementation to Web pages are presented. First attempts to master Microsoft Speech Server'2004 are discussed.

Keywords: SALT, text-to-speech synthesis, speech-enabled Web pages, speech recognition.

1. Indroduction

Over the past two decades, significant progress has been made advancing speech technology as a mainstream modality for human/machine interaction. It is obvious that speech recognition by humans will be more accuracy than recognition by machines for a long time. Although reducing the speech recognition error rate is an ongoing issue for speech researchers, the technology is already useful in addressing various interaction challenges. Telecommunications carriers already use speech recognition for directory assistance. Speech recognition is employed in many luxury cars to control non-critical functions. PC users can dictate directly into Microsoft Word XP. Text to speech (TTS) quality has also reached the level that is not only intelligible but also close to be natural. Unquestionably, speech technologies have had a positive impact on businesses and consumers. The analyst firm Datamonitor PLC estimates that the supply-side market for speech technologies and services will grow to more than $4 billion over the next three years [12]. With multiple locations and mobile employees, companies are demanding anytime, anywhere access to information. Enterprises can improve employee productivity and agility via a unified Web and voice portal that enables employees to access information through not only a PC, but also a wireless personal digital assistant (PDA) or telephone. This will be one of the key factors driving the growth of the speech industry, since speech is the only modality that can provide a consistent user interface across all devices.

In recent years two major standard proposals for voice based internet services have been prepared. One of them is called VoiceXML (Voice Extensible Markup Language) and is result of the initiative of IBM, Nuance and several other companies [10]. Another one is called SALT (Speech Application Language Tags) and is mainly result of Microsoft initiative [6]. Despite that both proposals are new and it is still unclear which of them will have better perspectives in the longer term we believe that it is important to develop Lithuanian internet services based on standard approaches and Lithuanian speech processing engines.

There are two alternatives for the developing and deploying of speech-enabled applications: Microsoft Speech Server (MSS) [6] and IBM WebSphere Voice Server [11]. The IBM WebShpere Voice Server is a VoiceXML 2.0-enabled speech environment. The VoiceXML is aimed at developing telephony-based applications, and takes the advantages of Web-based applications delivery to IVR (Interactive Voice Response) applications. Being different from IBM, Microsoft is using SALT 1.0 within Microsoft Speech Server. SALT targets speech-enabled applications across all devices such as telephones, PDAs, tablet PCs, and desktop PCs. The VoiceXML focuses on telephony application development whereas SALT is focused on multimodal speech applications that can be accessed by the whole device [11].

The IBM WebSphere Voice Server and SDK/ Toolkit is a member of the IBM WebSphere software family that is a SUN Java Framework-based Web

application environment. You have to install the SUN Java Framework in both the Voice Server and development environment. The IBM Voice Toolkit for WebSphere Studio enables developers to create voice applications in less time, by using a VoiceXML application development environment, which includes a VoiceXML editor, grammar editor, and a pronunciation builder, and allows application developers to easily add voice technology to middleware applications. The IBM WebSphere Voice Server for Multiplatforms V4.2 includes VoiceXML voice browser, IBM Speech Recognition Engine, IBM TTS Engine, telephony and media component, and so forth. It can connect with many telephony platforms, including WebSphere Voice Response for AIX/Windows, Intel Dialogic, Cisco or Siemens HiPath, and Voice Server Speech Technologies for Windows and Linux. The IBM WebSphere Voice Server is scalable, starting from basic analog telephony boards to high-density digital solutions with a T1/E1 interface, including Intel Dialogic D/120JCT, D/240JCT, D/480JCT, D/300JCT, and D600JCT [11].

Microsoft Speech Server is comprised of two main components, Speech Engines Services (SES) and Telephony Application Services (TAS). SES provides engine services on the server side that perform interpretation of the SALT and speech processing, including a speech recognition engine, prompt engine, and TTS engine for speech-enabling IVR customers through TAS. The Speech Recognition Engine component works on handing a caller's speech input; the Prompt engine joins pre-recorded prompts from a prompt database and plays them back to the caller; the TTS engine works Text-to Speech to synthesize audio output from a text string. TAS serves as a connection proxy between PBXs and SES by managing a set of SALT interpreters, calling control Call control (establishing, managing, and terminating the voice connection) and telephony interface managers [7]. TAS is comprised of both telephony hardware and a software interface. So far, the telephony hardware that can currently work with a TAS server include Intel Dialogic boards D41JCT, DMV160LP, which have 4 and 16 analog channels and boards DM/V480, DM/V960,. which have 48 and 96 digital voice ports. Normally, TAS must work with third-party Telephony Interface Manager (TIM) software that is an interface between TAS and telephony hardware; right now, for TAS software, Intel NetMerge CallManager and InterVoice TIM exist in the marketplace [11].

Since we are working for many years with Microsoft Visual Studio software and Microsoft Windows operation system, our choice was MSS. It combines Web technology with speech-processing services and telephony capabilities in a single system.

2. SALT and VoiceXML

VoiceXML first appeared in 1999, as Hypertext Markup Language (HTML) branch, with a purpose to define markup language for Interactive Voice Response (IVR) software.

In 2002 SALT forum presented new markup language – SALT, which has the same features as VoiceXML and additional ones, such as implementation of speech technologies for more devices, such as PDA's, TabletPC's. SALT allows interactive telephony dialog forms (as VoiceXML) and multimodal dialog forms. SALT allows to add speech content, such as voice input and output, to already created web page. This technology allows to read the content of a webpage using TTS engine or prerecorded files, to browse by speaking voice commands, to fill in internet forms by speaking, to leave voice messages and to create new possibilities for the customer.

SALT standard is a competitor to the VoiceXML standard for speech applications. However, SALT does incorporate some VoiceXML and the related W3C standards SRGS (Speech Recognition Grammar Specification) and SSML (Speech Synthesis Markup Language), so the two standards are not completely different from each other.

SALT appeals more to Microsoft developers and other developers coming in to speech from a Web development background. Experienced Web developers will understand the SALT development model because it uses the event-model most Web applications are built on. VoiceXML appeals more to developers with a background in traditional telephony or Interactive Voice Response (IVR) applications. VoiceXML is a larger standard because it is a complete standalone markup specification, where SALT depends more on existing functionality handled by other Web application specifications.

SALT is a small set of XML elements, with associated attributes and DOM object properties, events and methods, which apply a speech interface to web pages. VoiceXML is a large set of XML elements, since it is intended as a complete, standalone markup language.

SALT feature set is at low-level, to allow flexibility of interactional logic and fine-grained control of the speech interface. VoiceXML feature set is at a higher level, encompassing Web functionality and dialog flow. This allows VoiceXML pages to be used alone, and elementary dialogs to be built rapidly by the novice developer.

In SALT applications speech input and ouput is controlled by developer code in environment supported by the host page (e.g. Scripting module in HTML pages). Web functionality is also handled by the host page, so page navigation and form submission are written as usual in HTML. SALT also contains built-in declarative mechanisms intended for use in less rich device profiles. In VoiceXML applications speech input and output is controlled by VoiceXML dedicated execution environment. SALT is based on more modern Web development architecture than VoiceXML.

VoiceXML is more widely supported standard, with more than 500 members (leaders AT&T Labs, IBM, Lucent, Motorola) involved in the VoiceXML Forum versus SALT with its 70 members (leaders Cisco, Comverse, Intel, Microsoft, Philips, SpeechWorks).

Microsoft Speech Server does not currently support VoiceXML. It is important to mention that Microsoft has a goal to teach about 6 million programmers worldwide to use SALT technology and encourage in this way to create more speech enabled Internet services. This could be significant factor in competition between VoiceXML and SALT.

3. SALT overview

Developers cannot develop speech applications with SALT elements alone. The SALT elements contains no flow control structures. SALT is designed according to the following design principles:

- clean integration of speech with web pages;
- separation of the speech interface from business logic and data;
- power and flexibility of programming model;
- reuse of existing standards for grammar, speech output and semantic results;
- wide range of devices;
- minimal cost of authoring across modes and devices.

Both users and developers benefit from SALT ability to add speech capability to a wide spectrum of devices: from telephones and desktop PC's, to mobile phones and PDA's. SALT specification allows multimodal access, which means, that the user can access speech enhanced application, with various input methods. For example, SALT supports telephone keypads (DTMF) and mixed mode input (typing text with keyboard or pointing the screen with a pen) and obviously by speaking commands. SALT also uses multimodal output, for example, the content of the application can be traditional text, pictures or synthesized speech and audio file playback. There's also a capability to detect the proper output type for the user client device and respond accordingly.

Processing SALT applications is done differently depending on the type of client platform. For example, when a user accesses speech enabled webpage from a PC, the PC will use local resources such as installed TTS engine for speech output and recognition engine for speech input. If a speech enabled webpage (or application) is accessed from PDA or telephone, all processing will be done on server side. For this purpose Microsoft developed Speech Server for its Windows 2003 server family.

There are four main elements of SALT: “listen”, “prompt”, “dtmf”, and “smex”. The other elements: “param”, “grammar”, “record”, “bind”, “value” and “content” occur only as child elements of the top-level elements [1].

The “listen” element is used for speech recognition, for audio recording or for both. A “listen” element which is used for speech recognition contains one or more “grammar” elements, which are used to specify possible user inputs. A “listen” element which is used for audio recording contains a “record” element which is used to configure the recording process. A “listen” element used for simultaneous recognition and recording holds one or more “grammar” elements and a “record” element. In all cases, “bind” can be used to process the results obtained from recognition and/or recording.

The following code is a simple “listen” example which holds a remote grammar file containing city names (miestai.xml), and a “bind” statement to process the recognition result (it displays recognized result in text box "txtBoxIsKur"):

```
<salt:listen id="kelione">
    <salt:grammar
src="./miestai.xml" />
    <salt:bind
targetElement="txtBoxIsKur"

value="/result/is_kur" />
</salt:listen>
```

The “prompt” element is used to specify the content of audio output. The content of prompts may be one or more of the following:

- inline or referenced text, which may be marked up with prosodic or other speech output information;
- variable values retrieved at render time from the containing document;
- links to audio files.

Prompts can be specified and played individually and, in more complex applications, they may be managed through a model of prompt queuing. For example, the following prompt will read synthesized text "Hello, nice to meet you":

```
 <prompt id="labas">
   Hello, nice to meet you.
</prompt>
```

The “dtmf” element is used in telephony applications to specify possible DTMF inputs and a means of dealing with the collected results and other DTMF events. Like “listen”, its main elements are “grammar” and “bind” and it holds resources for configuring the DTMF collection process and handling DTMF platform and collection events.

“Smex” (Simple Messaging EXtension) element communicates with the external component of the SALT platform. It can be used to implement any application control of platform functionality such as logging and telephony control.

Today main manufacturers of programming and web design tools are trying to integrate support of

SALT specification to the newest versions of their products. Still dominates two methods of SALT technology implementation:

- Microsoft's Speech Application SDK (SASDK) with Microsoft's Visual Studio.NET [2];
- VoiceWebSolution's “Voice Web Studio” plug-in for Macromedia's Dreamweaver MX [8].

4. Microsoft Speech Application SDK

Microsoft Speech Application SDK (SASDK) is freely downloadable add-on and documentation package for Microsoft Visual Studio.NET for creating speech enhanced applications. With SASDK user can develop speech applications as: touch tone interfaces (eg., DTMF), voice-driven menus (eg., IVR) and multimodal interfaces (eg., WebPages). Applications are developed using ASP.NET, adding speech to standard Web applications using Speech Application Language Tags (SALT).

SASDK includes speech control, grammar and prompt editors, sample applications and libraries, logging and debugging tools and speech add-ins (debug and end-user versions) for desktop version of Internet Explorer and for Windows Mobile-based Pocket PC’2003 version of Internet Explorer.

Speech Control Editor is the primary development environment for building the components of a speech-enabled application. There are two uses of speech controls:

- multimodal: user can control application keyboard, mouse, or speech-enabled controls;
- voice only: user can control application using only speech or by pressing buttons on the telephone (DTMF tones).

ASP.NET Speech Controls are ASP.NET controls that render SALT in a speech-enabled Web application. Using property builders developer can set most common properties of Speech Controls in an organized, intuitive format. Using Speech Application Wizards developer can quickly configure settings for both voice-only and multimodal applications, creates a template containing prompt projects, a blank grammar file, the grammar library and application-specific debug settings.

Speech Grammar Editor provides a graphical view of the grammar incorporated into an application, breaking it down into different paths and displaying them visually. Semantic authoring tool supports editing the contents of the SRGS (Speech Recognition Grammar Specification) tag element, which associates semantic values with words or phrases in the recognition grammar and fully supports W3C specification for adding semantic information to grammars. Pronunciation Editor enables developers to look up word pronunciations in a copy of the core lexicon of the speech recognition engine and combine or edit the phonemes that specify the pronunciations of words.

Speech Prompt Editor is used to record and manage the audio prompts played by the application. Enhanced Prompt Editing allows to create dynamic prompts. For example, when user input is not recognized, computer can say "Sorry, i don't understand you" for the first time and "Please repeat" for the second.

Speech add-ins allow browser to recognize SALT and execute speech-enabled Web applications. The add-in fully supports SALT 1.0 specification for rendering speech-enabled web pages and displays audio meter tool, which is similar to the display on an audio spectrum which provides visual feedback of the volume level of audio input in multimodal applications.

There are 14 sample telephony applications and one sample multimodal application, bundled with SASDK, that demonstrate how to perform specific tasks and how to use specific components of the SASDK. Debugging tools of SASDK allow developers to test and debug both voice-only and multimodal speech-enabled Web applications. The most useful ones are: Telephony Application Simulator, which allows developers to simulate using a telephony application, including dialogue flow and call management, on a telephony server and Speech Debugging Console, which allows developers to inspect an application dialogue that is running on Telephony Application Simulator, Internet Explorer or Pocket Internet Explorer and examine the state of the dialogue. SASDK allows developers to log application specific events and to view real-time events in a debugging console [2].

5. Simple speech-enabled web application built with SASDK

Creating simple speech enhanced web applications in Visual Studio.NET with SASDK installed is a breeze. Knowing HTML basics and “Visual studio” workspace would end up user with a simple finished project, without going into the deeps of programming.

To create simple text-to-speech application user must choose "New Project" from file menu. This will end up with a template wizard selection. "Speech Web Application" from the “Visual C# Projects” is the obvious choice. In location line the user can specify the location of the project or leave the default one. In "application settings" the user must choose if planned application will be multimodal (voice, text, graphics, etc.) or voice only. After pressing “Finish” “Visual Studio.NET” will generate the basic template.

The goal of the previously noted speech application will be: to read the text from the textbox and to output synthesized voice back to the user. For this drag and drop the following items from the toolbox in “Visual Studio.NET” (Figure 1):

- Prompt from Speech Toolbox;
- Text Field (name it “form”) and Button from HTML Toolbox.

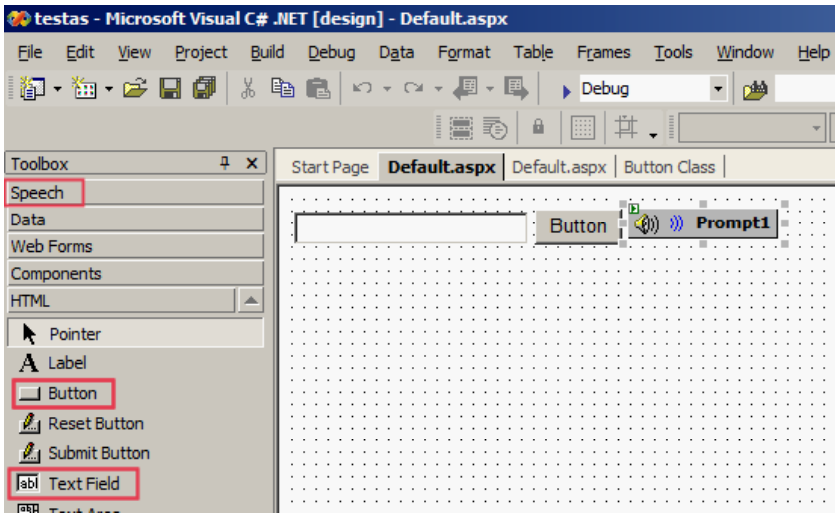

Figure 1. Elements for synthesizing text in "Visual Studio.NET"

```
<%@ Page language="c#" Codebehind="Default.aspx.cs" AutoEventWireup="false" Inherits="testas._Default" %>
<%@ Register TagPrefix="speech" Namespace="Microsoft.Speech.Web.UI" Assembly="Microsoft.Speech.Web, Versio
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
    <HEAD>
        <title>Default</title>
        <meta name="GENERATOR" Content="Microsoft Visual Studio .NET 7.1">
        <meta name="CODE_LANGUAGE" Content="C#">
        <meta name="vs_defaultClientScript" content="JavaScript">
        <meta name="vs_targetSchema" content="http://schemas.microsoft.com/intellisense/ie5">
    </HEAD>
    <body MS_POSITIONING="GridLayout" xmlns:speech="http://schemas.microsoft.com/speech/WebControls">
        <form id="Form1" method="post" runat="server">
            <INPUT type="text" id="forma"> <INPUT type="button" value="Button" onclick="Prompt1.Start()">
            <speech:Prompt id="Prompt1" runat="server">
                <InlineContent>
                    <speech:Value runat="server" TargetElement="forma"></speech:Value>
                </InlineContent>
            </speech:Prompt>
        </form>
    </body>
</HTML>
```

Figure 2. HTML code of speech-enabled Web application with highlighted added code

That's it – application is finished. You can now check your work by debugging from "Visual Studio.NET" or by running it in internet explorer.

6. Voice Web Studio

"Voice Web Studio" (VWS) is an add-in for Macromedia Dreamweaver MX that enables building speech-enabled web applications based on SALT 1.0. Integration within Dreamweaver provides the developer with a familiar user interface, source and design view, and compliance with multiple operating systems. "Voice Web Studio" can speech enable any Web page. Using VWS's menu in Dreamweaver, user can quickly create and insert speech content into the code by filling in interactive forms. In addition, the same tool can be used for future multimodal clients that support WML, XHTML, CHTML, and XML, including various scripting languages. For easier manipulation and editing of speech elements, VWS comes with build-in visual controls for displaying speech elements in Dreamweaver's design view window [8].

There are four main function buttons in VWS: "SALT prompt"  , "SALT listen"  , "SALT create dialog"  and "SALT play dialog"  :

- "SALT prompt" button is used to insert a "prompt" element into the page to play synthesized text or an audio file. There is no number limit of "prompt" elements within a page. To quickly add content to be played, user can highlight HTML or text and press "prompt" button. The inserted "prompt" elements will be inactive, unless instructed to by a dialog object or command;
- "SALT listen" is used to insert a "listen" element into the page, to add speech recognition feature. There is no number limit of "listen" elements within a page. The inserted "listen" elements will

be inactive, unless instructed to by a dialog object or command;

- "SALT create dialog" is used to create speech dialogs between human and PC. Dialog is just a HTML script, but VWS saves user from altering it manually, which can be quite difficult, if big number of dialogs is used within a page. VWS allows to create three dialogs at once. There's no number limit of dialogs within a page.

"SALT play dialog" is used to select a "dialog", "listen" or "prompt" element to activate in a multimodal setting such as clicking a textbox, mouse over an image, using the keyboard to focus in on a selection, pressing a button and so on. There's no number limit of multimodal interactions within a page.

7. Simple speech-enabled web application built with "Voice Web Studio"

Creating simple speech enhanced web applications in Dreamweaver MX with "Voice Web Studio" installed is as simple as in Visual Studio. NET, assuming that you are familiar with Dreamweaver's workspace and interface and know HTML basics.

To create simple text-to-speech application user must choose "New..." from file menu. This will end up with a web application type selection. For this example select "Category: Basic page > HTML page" and click "Create".

The goal of this sample speech application will be to build the same application as in SASDK, but in Dreamweaver's environment. That is: to read the text from the textbox and to output the synthesized voice back to the user. For this add "Text Field" and "Button" form elements from the Dreamweaver's form menu. (Figure 3).

Add this point, all that is needed, is to add speech functions to the page: click on "SALT prompt" button from Dreamweaver's (VWS's) SALT menu, enter prompt name (eg., "test"), set "Read Textbox" to "textfield" and click OK (Figure 4).

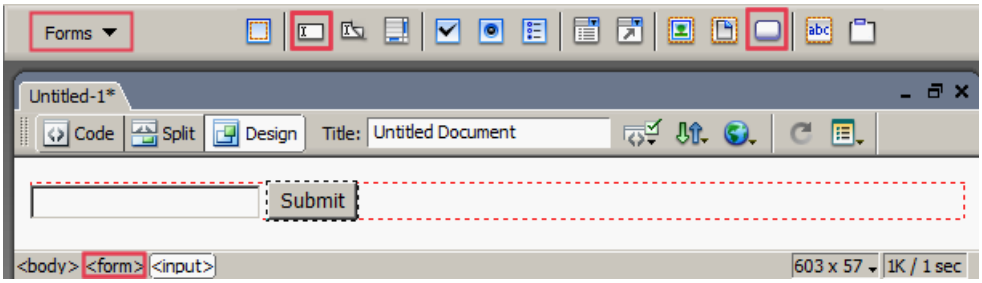

Figure 3. Form elements for synthesizing text in Macromedia Dreamweaver MX

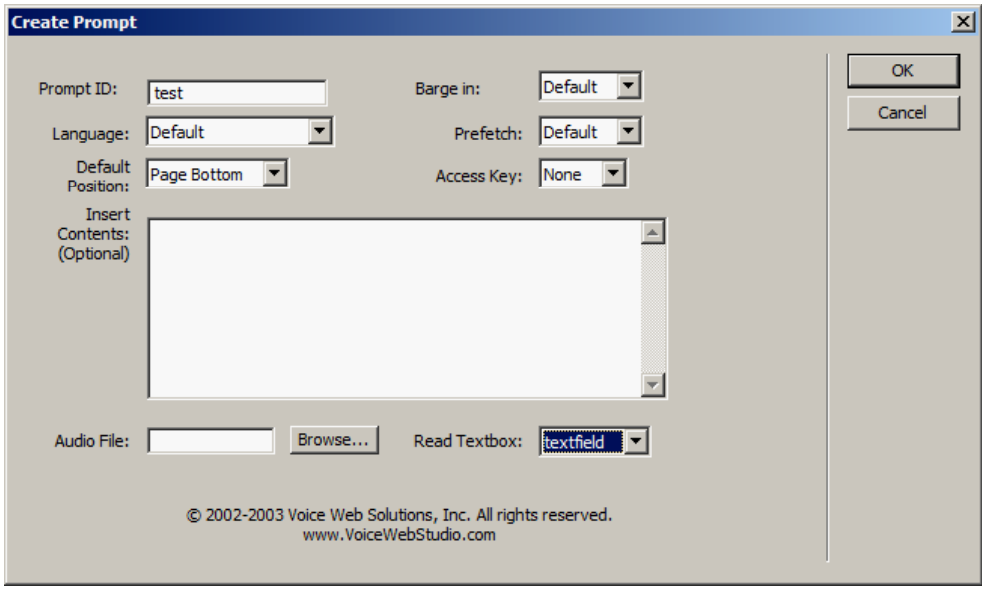

Figure 4. Prompt creation window in Macromedia Dreamweaver MX

To create a handler to the button to start the prompt, which will output the content entered in the textbox:

- select the button in Dreamweaver's design view and set it action to none in properties window;
- click on "SALT play dialog" button from Dreamweaver's SALT menu, select: Bind to->Form Item, onClick->salt:Prompt "test" and click OK. (Figure 5).

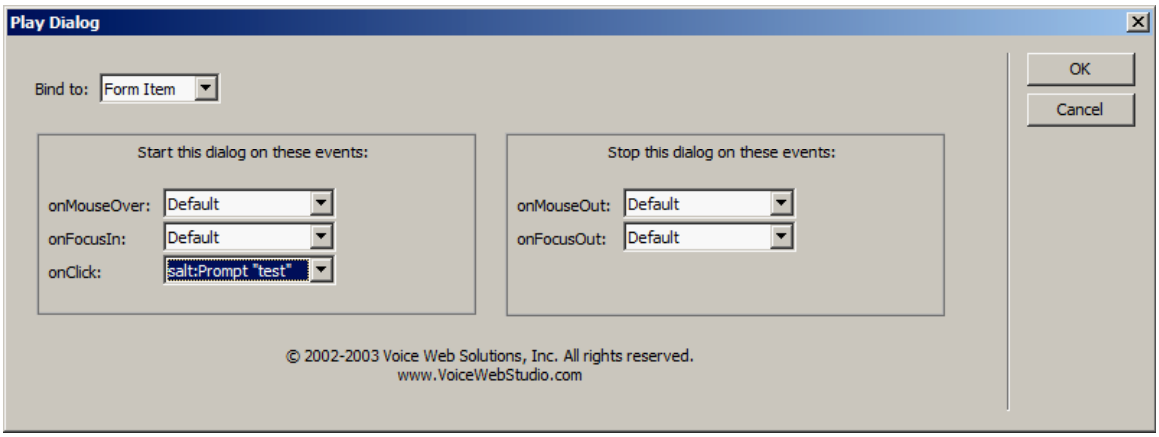

Figure 5. Adding prompt starting handler to the button in Macromedia Dreamweaver MX

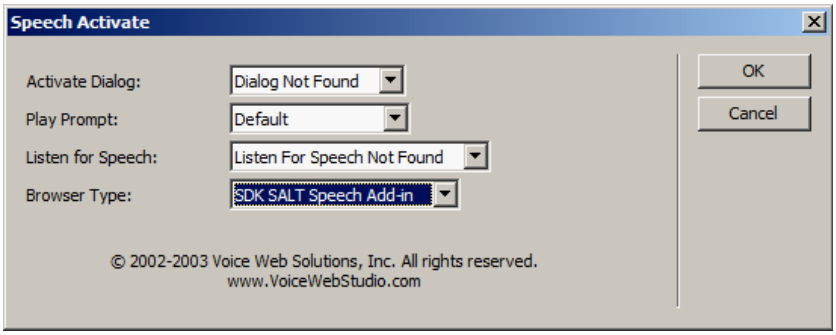

Figure 6. Speech activation window in Macromedia Dreamweaver MX

That's it – application is almost finished. All you need to do is to select "Speech Activate" from Dreamweaver's commands menu and set your internet explorer's speech add-in type (Figure 6).

You can now test your page either from Dreamweaver (F12) or running it directly in internet explorer.

8. "Voice Web Studio" vs "Microsoft Speech Application SDK"

Voice Web Studio and Speech Application SDK are based on SALT specification, version 1.0. Both support open W3C standards such as HTML, XML. SASDK has native support for ASP.NET. SASDK is optimized for creating speech applications for deploying on Microsoft's Speech Server 2004. Open MSS architecture allows integration of third party SAPI based speech engines.

Both software packages have graphical user interfaces: VWS – adds new speech toolbar in „Macromedia Dreamweaver's" environment, SASDK – toolbox in "Microsoft Visual Studio's" environment. VWS's environment is a more common and intuitive to amateur developers, than SASDK's – understandable only to the masters of Visual Studio.

VWS offers not much, but interactive tools, allowing amateur user to master the environment quickly. By filling interactive forms, users can create simple speech enabled HTML webpage in a few minutes. VWS allows the previewing of webpage in the browser directly from the dreamweaver's environment. Microsoft offers a lot more powerful tools and functions in SASDK, allowing to develop more advanced applications. SASDK has special built-in tools for telephony functions, such as "answer call", "make call", etc. SASDK also contains a wealth of speech related resources, including prompt and grammar editors, grammar library, speech recognition and synthesis engines (English, French and Spanish only in version 1.1). SASDK also allows the previewing of webpage in the browser from the Visual Studio's environment.

VWS's code editing and debugging tools are limited to those of Dreaweaver's, as SASDK's are limited to Visual Studio's. Both packages highlight various functions, scripts, tags, etc. However Visual Studio's debugger tools are a lot more powerful than dreamweaver's, with special tools for speech debugging, such as prompt and grammar validation tools.

VWS offers simple online documentation and tutorials, SASDK is bundled with detailed reference documentation, integrated into Visual Studio's help system for quick access within your work environment. SASDK also comes with a lot of sample and reference speech applications, speech application deployment service and a debug version of Speech Add-in for Internet Explorer.

9. Examples of speech-enabled Web applications

To use above mentioned technologies for Lithuanian language, one must have a Lithuanian TTS and recognition engines installed. Two Lithuanian text-to-speech synthesizers "Aistis" and LtMBR are integrated to Microsoft Speech Programming Interface (SAPI) while Lithuanian speech recognizer compatible with SAPI so far is under development [4].

Recently activities to develop Lithuanian speech-enabled Web pages have been started in Speech Research Laboratory of Kaunas University of Technology: Microsoft Speech Server was mastered, Intel Dialogic board D41JCT with 4 analog telephone channels was acquired. Some experience of creating voice-based telecom services is reached [5].

Some demonstrations of speech–enabled Web pages were prepared using SASDK: a) reading of input text by voice; b) filling of forms by voice, c) review of SALT technology and using VWS (virtual discotheque) (http://www.speech.itpi.ktu.lt). Speech prompts by TTS are used in all these applications, voice dialogues are used only in the virtual discotheque and in the filling of forms by voice. User can control these multimodal applications by keyboard, mouse or by voice commands. To test this web page you need to install Windows'2000 or Windows'XP system and freely distributed SASDK. Lithuanian TTS engine is needful for TTS prompts in Lithuanian.

10. Conclusions

There are two alternatives for the developing and deploying of speech-enabled applications: Microsoft Speech Server (MSS) and IBM WebSphere Voice Server. Since we are working for many years with Microsoft Visual Studio software and Microsoft Windows operation system, our choice was MSS. It combines Web technology with speech-processing services and telephony capabilities in a single system.

SALT technology allow to access the content of a webpage using text-to-speech engine, browse by speaking voice commands, fill in internet forms by speaking, leave voice messages and create entirely new possibilities for the customer. It has the same features as VoiceXML and additional ones, such as implementation of speech technologies for more devices, such as PDA's, TabletPC's.

The SASDK package provides an integrated, comprehensive platform for building speech-enabled telephony and Web applications. It is more powerful tool in comparison with "Voice Web Studio".

Some demonstrations of speech–enabled Lithuanian Web pages using SASDK and „Voice Web Studio" were prepared (http://www.speech.itpi.ktu.lt). First attempts to master Microsoft Speech Server and Intel Dialogic board D41JCT were started. The access from telephone to speech enhanced Web pages will be enabled with this telephone board.

References

[1] **B. Graham.** Speak and listen to the Web using SALT. *Retrieved April* 5, 2005, *from http://www.developer.com/voice/article.php*/2174471.

[2] **R**. **Irving** A Lap around the Speech Application SDK. *Retrieved April* 5, 2005, *from http://www.microsoft.com/speech/techinfo/techarticles/default.mspx*.

[3] Microsoft speech server. *Retrieved April* 5, 2005, *from http://www.microsoft.com/speech*.

[4] **A. Rudzionis, K. Ratkevicius, V. Rudzionis.** Speech recognition research and some speech technologies applications in Lithuania. *In: Proc. of the first Baltic Conference HLT*-2004. *Riga, Latvia*, 132 –138.

[5] **A. Rudzionis, K. Ratkevicius, V. Rudzionis**, **P. Kasparaitis.** Voice operated informative telecom services. *Elektronika ir elektrotechnika*, Nr.3(45), 2003, *Kaunas, Technologija*, 2003, 17-22.

[6] SALT forum. Retrieved April 5, 2005, from *http://www.saltforum.org*.

[7] Voice Technology Goes Mainstream with Release of Microsoft Speech Server 2004. *Retrieved April* 5, 2008, *from http://www.microsoft.com/presspass/features/2004/mar04/03-24SpeechServer.asp*.

[8] Voice Web Community. *Retrieved April* 5, 2005, *from* http://www.voicewebsolutions.net.

[9] VoiceXML forum. Retrieved April 5, 2005, from *http://www.voicexml.org*.

[10] **Xiaole Song**. Comparing Microsoft Speech Server 2004 and IBM WebSphere Voice Server V4.2. *Retrieved April* 5, 2005, *http://www.developer.com/voice/article.php/b3381851.html*.

[11] **Xuedong Huang**. Making Speech Mainstream. *Retrieved April* 5, 2008, *from http://www.microsoft.com/speech/docs/HuangSpeechArtfinal.html*.

DOI: 10.5755/j01.itc.34.2.12004