Scientific Journal Impact Factor (SJIF) : 1.711
International Journal of Modern Trends in Engineering and Research
ISSN (ONLINE) : 2349 - 9745 ISSN (PRINT) : 2393 - 8161

WEB APPLICATION FOR VOICE OPERATED E-MAIL EXCHANGE

Sangeet Sagar[1], Vaibhav Awasthi[2], Samarth Rastogi[3], Tushar Garg[4], S. Kuzhalvaimozhi[5]

[1, 2,3,4,5] Information Science and Engineering, National Institute of Engineering, Mysore, Karnataka, India- 570008

Abstract—E-mail, in this age, is the primary mode for passing on crucial information. Along with getting tech-savvy, we need to be the more user-friendly as well. This paper proposes an application that will compose the mails as you speak, send it to a valid e-mail address, read aloud the received e-mails; all of this voice enabled with minimal use of keystrokes. With the use of speech-to-text and text-to-speech synthesizers the voice input will be converted to text and vice-versa respectively. Javascript and PHP codes are used to facilitate the mailing module and integrate it with the speech synthesizer module. The user only needs to worry about speaking clearly and rest is taken care of, thus hugely advantageous for the disabled.

I. INTRODUCTION

The advancement in computer based accessible systems has opened up many avenues for the visually impaired across a wide majority of the globe. Audio feedback based virtual environment like, the screen readers have helped Blind people to access internet applications immensely. The Voice Based Navigation has become quite prevalent today. When scripting a web-based application, tailor macros to one browser. It may even be necessary to script for a particular version (or versions) of a browser. Browser settings affect script performance.

People to access internet applications. Voice recognition systems have been deployed in desktops and smartphones. Making calls, opening apps within phones are some important implications of it. Voice enabled search (by Google and various others) is also an existing application of speech synthesizing. Asking for directions while driving and hearing the response through speech synthesis illustrates how practical "hands-free" applications can be to mobile users.

The most mature product on the market, like Dragon NaturallySpeaking etc., excels at recognizing not only words and numbers, but also commands for formatting and revising text, web-browsing, and controlling windows, dialog boxes, and menus. As we see the text-to-speech and vice-versa API's have not exploited the E-mail services. Therefore, this paper proposes to use a voice recognition engine and integrate it with an mailing system. Thus,both serving as two different modules. Our system will strive to create a voice recognition engine with text-to-speech and speech-to-text conversion compatibilities which will then be integrated with an emailing system, thereby giving an add-on feature to the E-mail service.

II. SYSTEM DESIGN AND ANALYSIS

Features such as user preferences, browser tabs, toolbars, add -ins, and extensions may change the appearance, layout, and behaviour of pages. Some settings help while others hinder. Keep browser settings in mind when developing voice macros for web-based applications, and document the necessary browser settings.

In wide foray of voice recognition applications, the highly popular use these days are voice messages

within phones, tablets and all devices alike. Interactive and action oriented applications based on some voice recognition engine have not been exploited. This is where we aim to deliver, a mailing system completely on a voice recognition engine. Environment like, the screen readers have helped Blind.
The overall system comprises of three very simple parts or module, namely:

- Automatic Speech Recognition Module (ASR)
- The mailing module platform created on HTML will use JavaScript and set of JAVA APIs to embed the voice recognition engine inherent to the HTML.
- Text-to-speech module

A. Speech Recognition Module (ASR)

Automatic speech recognition (ASR) technologies today can correctly recognize and write down more than 90% of a long series of spoken words for many languages. The API itself is agnostic of the underlying speech recognition implementation and can support both server based as well as embedded recognizers. In case of embedded recognizers, the browser itself would have the capability of speech recognition and this would be quite similar to the current software that does speech recognition. In this approach, the browser would record the voice from the microphone and perform the speech recognition process on the input voice locally and generate the resultant text. This would be a fast process and could be done offline as well.[5]

Whereas in the second approach, the browser would record the voice from the microphone and stream the audio data to its server which is responsible for the speech recognition and after the speech recognition process at the server, it would send the result text to the browser. The advantage of using a server based approach is that speech recognition would be more precise and accurate than the local approach because large amount of training data collected at the central servers help improve accuracy of the speech recognition. .

In our demonstration, Chrome is the browser which captures audio and streams to Google's servers for speech recognition and the text is resulted from the servers and sent to Chrome browser. In this demonstration, the software part that has the responsibility to capture audio and stream to servers is embedded directly in the Chrome web browser.

The engine (Webkit) uses the Hidden Markov Model for speech recognition.

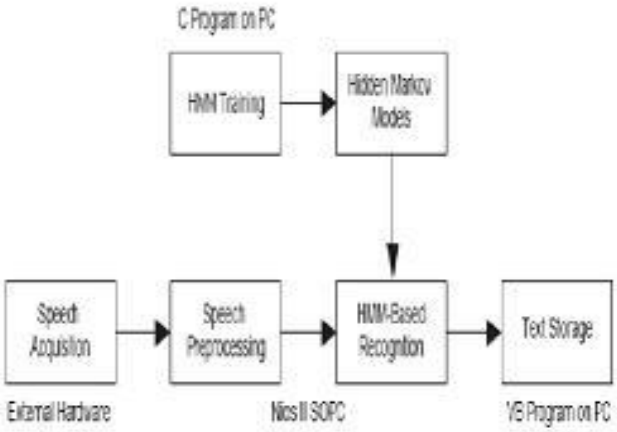

Fig 2.1: Hidden markov model(HMM) block diagram.

An *n* -state *M* -mixture HMM is a Markov process composed of *n* states, denoted by **S=** {SI, *S* , ,. .. , *S* , } , and *m* mixtures for each state. The temporal ordering of speech signals are characterized by the state-transition probabilities between states. The spectral observation probability for each state is represented by a continuous probability density function specified as a mixture of Gaussian densities.[1]

B. The mailing module

The mailing module will facilitate the integration of speech recognition modules with itself to actually compose, send and receive e-mails. The voice input (signal) given by the user will be converted to text by the module's embedded API and then will be taken as input by the form tags in HTML or for some other page request. The mailing interface will consist of basically HTML tags and underlying Jscript for handling various operations as specified in Fig. 2.2

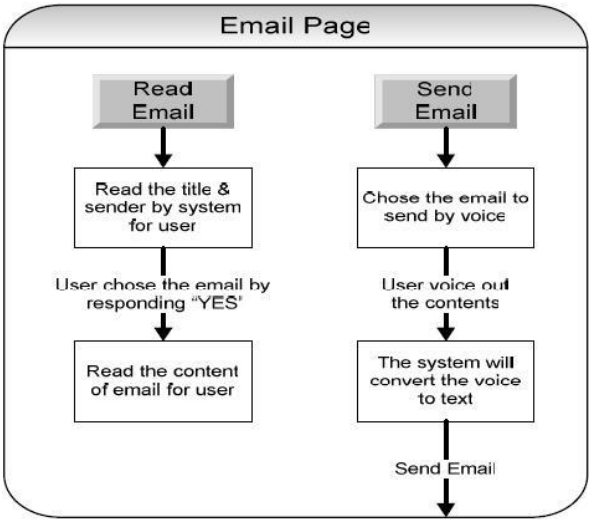

Fig 2.2: Block diagram for E-mail module integrated with speech synthesizers. [5]

C. Text-to-speech module

The Text to Speech (TTS) Synthesizer first converts the input text into its corresponding linguistic or phonetic representations and then produces the sounds corresponding to those representations. With the input being a plain text, the generated phonetic representations also need to be augmented with information about the intonation and rhythm that the synthesized speech should have. This task is done by a text analysis module in most speech synthesizers.

The transcription from the text analysis module is then given to a digital signal processing (DSP) module that produces synthetic speech.[3] Syntactic processing has to be done on the input text to obtain a phonetic transcription. In this stage, the sentences in the text are divided into words and the numbers, abbreviations and acronyms are expanded. This is commonly done by using regular grammars. Synthesis units with varied prosodic and spectral characteristics from a large, single speaker speech database are selected and concatenated to generate synthesized speech[1]. Given a phonetic transcription along with prosodic information, the DSP module tries to mimic the human speech production system to produce speech Text is synthesized by selecting appropriate units from a speech database and concatenating them.[3]

III. SYSTEM IMPLEMENTATION

As seen in fig. 3, the first step of implementation includes prompting the user to login by speaking out loud. Thus, bringing into picture the

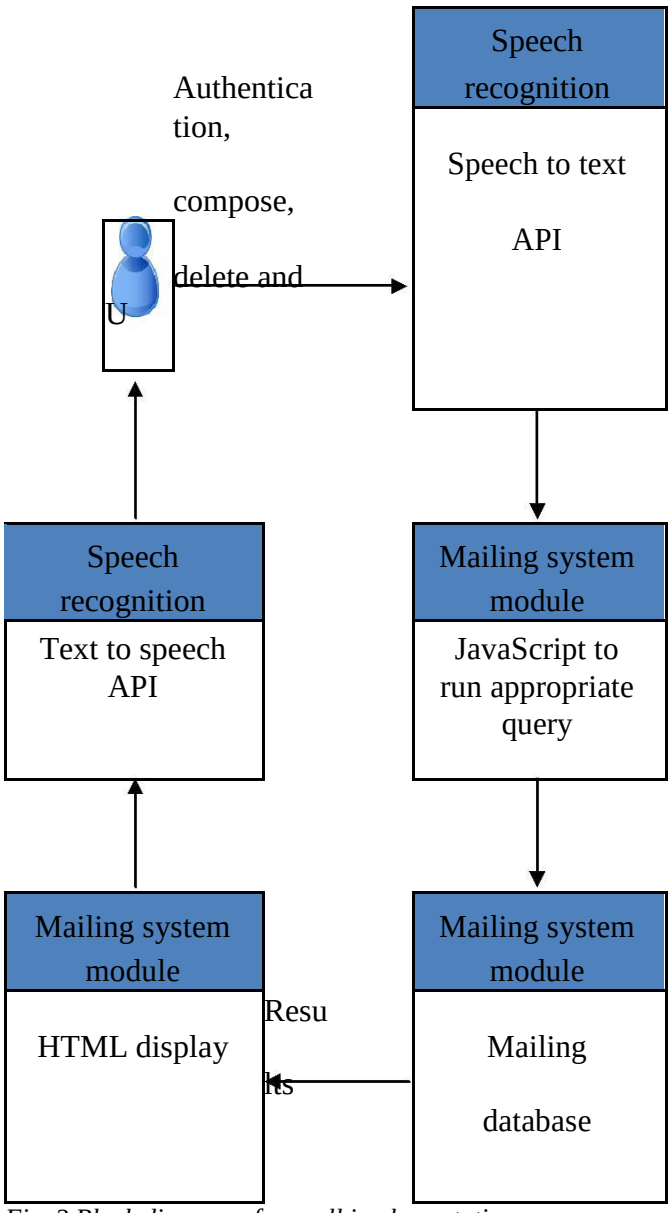

Fig. 3 Block diagram of overall implementation.

ARS, followed by the mailing system and the the text-to-speech module. ARS is implemented using the Webkit speech engine. Webkit is the web browser engine (called layout engine or rendering engine) of Google Chrome web browser. Each browser has an underlying engine that does the work of interpreting HTML, CSS and JS and laying out the elements on the browser screen. For instance, Gecko is the layout engine of Firefox, Trident is the layout engine of Internet Explorer and Presto is the engine for Opera. These layout engines are the core or kernel of any web browser and most of them are open source including gecko, webkit and others. Audio is collected from the microphone, and then sent to a Google server (a Gwebservice) using HTTPS POST, which returns a JSON object with results. The HTML code that does this is very simple can be used to invoke the Webkit services.

<input id="speech" type="text" speech="speech" x-webkit-speech="x-webkit-speech" onspeechchange="processspeech();" onwebkitspeechchange="processspeech();" />
speech="speech": tells the browser that it is not a normal <input> element, rather it is an <input> element that can take input by speech or voice. This adds a small mic to the right of the <input>

element which can be clicked so that the browser can capture voice from the microphone.

x-webkit-speech="x-webkit-speech", this attribute is just a redundant attribute which will possibly be removed.

onspeechchange="processspeech();"

This subscribes the processspeech() event handler to the speech change event which occurs when the speech or voice input changes the value of the <input> element. processspeech() is just a function name and could have been anything else.

After getting the converted to text input , it needs to be redirected to a certain e-mail ID. This is easily achieved by a few lines of PHP code. PHP seems to make most things extremely easy. Sending email from a PHP script is no exception. All it takes is the right configuration (to send mail using a local or a remote server) and one function.

MAIL( )

A simple implementation of the function is :

mail($admin_email, "$subject", $comment, "From:" . $email);

This completes the task of sending the E-mail. A notification will be displayed to the user through a voice/audio file. Now the task of receiving and retrieving the E-mails from the inbox will begin. For this purpose we will be using two of Java's web based servers called POP3 and IMAP servers. With this we can receive emails and can synchronize these servers with the Mobile Application. [3]

There are two main ways that users can access their email. POP3 (Post Office Protocol 3) is the standard way which has been around for decades. It is very similar regular mail. Messages are delivered to your computer, put in your mailbox, and are then your responsibility.

The other, newer, method is IMAP (Interactive Mail Access Protocol). As you might guess by the name, it is not like the mailbox on your house. With IMAP mail is delivered to the server, and you connect to the server to see your mail. The mail is not stored on your machine. When a message is marked as read, it is marked as read on the server, not on your computer.At first this may seem kind of backwards, but what IMAP lets you do is access your mail from different programs, different computers, or even via a web page and your mail will always reflect all your changes.

Advantages of IMAP

- Email is available from any machine you happen to use.
- Email is stored on the server, so your email cannot be deleted/destroyed if your computer should happen to crash, be stolen, or destroyed.
- You can access IMAP mail via the web, without even needing a mail client installed. This means you can check your mail from someone else's machine or even a public terminal and not have to worry about the security of your password

Now after getting access to the mails in the user's inbox, we use our TTS synthesizer to convert it into speech and read it aloud. For this purpose a Screen Reader can be used which will read aloud simple text as it appears on the screen. Screen readers can query the operating system or application for what is currently being displayed and receive updates when the display changes. For example, a screen reader can be told that the current focus is on a button and the button caption to be

communicated to the user. Microsoft provides us with Narrator is a Text-to-Speech utility for users who are blind or have impaired vision. Narrator reads what is displayed on your screen: the contents of the active window, menu options, or the text that you type. Narrator is designed to work with the Notepad, WordPad, Control Panel programs, Microsoft Internet Explorer, the Windows desktop, and Windows Setup. Narrator may not read words aloud correctly in other programs. Narrator has a number of options that allow you to customize the way screen elements are read.

- You can have new windows, menus, or shortcut menus read aloud when they are displayed.
- You can have typed characters read aloud.
- You can have the mouse pointer follow the active item on the screen.
- You can adjust the speed, volume, or pitch of the voice.

IV. CONCLUSION

In order to provide access to next generation information systems to vision impaired users and to make this Application much more advanced, it is necessary to focus upon the fact of including additional features like "Attaching files" to the emails and also to download the same from the inbox. This not only makes the email access more complete but also could be a harbinger to the new world of Assistive Web Technologies.

As the mails will be received on an e-mail address, using either the windows TTS engine or a third party software, the contents of the mail will be read out to the authenticated user.

Many mobile app service exists which can read aloud the text messages from the phone but a certain web application solely dedicated for e-mails is unheard of.

REFRENCES

[1] Chung-Hsien Wu and Jau-Hung Chen. *Speech activated telephony email reader (sater) based on speaker verification and text-to-speech conversion*, IEEE Transactions on Consumer Electronics, Vol. 43, No. 3, AUGUST 1997.
[2] Nuno Baptista, Rui Prior, Manuel E. Correia. *Telephone Interface for the Email Service*, Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns,2009.
[3] Gaurav Anand, Geethamsi S, Mr. R V R Chary, CH.Madhu Babu. *Email Access By Visually Impaired*, International Conference on Communication Systems and Network Technologies,2013.
[4] K.V.N.Sunitha, N.Kalyani. *VMAIL-Voice Enabled Mail Reader*, International Conference on Recent Trends in Information, Telecommunication and Computing,2010.
[5] Halimah B.Z. , Azlina A. , Behrang P., Choo W.O. . *Voice Recognition System for the Visually Impaired: Virtual Cognitive Approach, 2008.*

