IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 4, April 2016

Voice Browser

Aishwarya Chitanvis[1], Ankita Joshi[2], Bhumik Dedhia[3] , Prof. Grishma Sharma[4]

Department of Computer Engineering, K.J. Somaiya College of Engineering, Mumbai, India [1,2,3,4]

Abstract: A web browser is used to display web pages, navigate from one web page to another with the use of hyperlinks, and download any type of data right from PDF files, Presentations, Word files to music, videos and images. Browsing is using the mouse, keyboard and touch (in case of smartphone applications. But what about the handicapped and the visually impaired users? How would they use the search engine? We have tried to come up with a solution by creating a Voice Based Browser that is a completely hands-free search.

Keywords Speech Recognition, Text to Speech, Web Browsing, Speech Synthesis, web crawler.

I. INTRODUCTION

A web browser is used to display web pages based on what the user has searched for and download any type of data. Speech Recognition is quickly gaining pace in applications like Google Now, Siri, Cortana, etc. Speech Recognition is important in building application for the handicapped and visually-impaired users. We have implemented this feature in our Voice Based Browser. With the help of Speech to Text feature, the user will speak out the words he/she would like search for on the browser. The Search Box will again repeat the words using the Text to Speech feature so that the user will know the right words have been uttered. Once the web pages are displayed, the hyperlinks would be spoken to the user with the Text to Speech feature. Thus, the disabled users can use and navigate through the browser easily.

II. RELATED THEORY

In many studies, algorithms and applications have been implemented to facilitate the internet browsing or to revolutionize the traditional way of surfing the www in different angles. One application was implemented to display enumerated links in the browser window and to have also a compass mouse with a curser positioned over a mouse-over pull-down menu by speech recognition.

In modern day many users do not realize but they interact with voice web components through VoiceXML regularly. There are other standards that are also supported as a part of VoiceXML.

VoiceXml (VXML): A language that creates audio dialogs that use synthesized speech, digitized audio, recognition of speech, recorded speech and telephony;

Speech Grammar Recognition Specification (SRGS): A grammar data document that is used to specify words, phrases, patterns of words, sentences etc. in the manner they are required to be listened by the listener/recognizer;

Semantic Interpretation for Speech Recognition (SISR): A data document that defines the rules and protocols of grammar and language for extracting semantic outputs from recognizer;

*Pronunciation Lexicon Specification (PLS)***:** A document representing phonetic information to be used in speech recognition and synthesis;

*Speech Synthesis Markup Language (SSML)***:** It is a markup language for rendering a combination of pre-recorded speech and live speech and other audio files.

III.PROPOSED THEORY

This web browser is implemented using MS Visual Studio 2015 which helps in converting Speech to Text as well as Text to Speech using C# Language. The commands like Back, Go, Refresh, Back, Forward, Home, Speak, Listen, Stop will perform functions as assigned in program.

The steps performed are:

I. Web Browser takes voice input through microphone and converts it into text. This helps the user to select the specific URL needed.
II. The commands will be matched with the command set where the specific commands with specific actions are stored e.g.: GO, etc.
III. If the match occurs, then that respective action will be executed.
IV. By giving the voice command user can operate the web browser.
V. User will be able to open web pages.
VI. The text appearing on the URL bar will converted back to speech, giving output in voice.
VII. Keywords are assigned for every link in URL.
VIII. Dialogue box with the keywords open and hence the respective link opens.

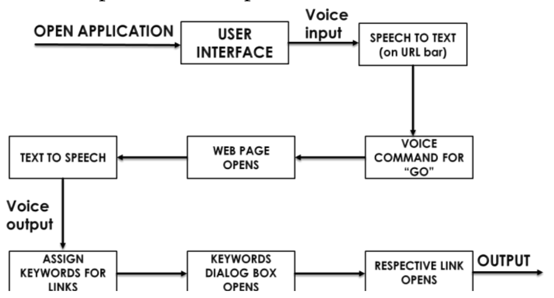


Fig 1: Flow diagram for voice based web browser

Description:

- User: The user interacts with the browser by giving voice commands to the browser and also reads text back to speech.
- Web Browser: The web browser takes the command from the user and sends it to the speech to text converter for text conversion.
- Speech to Text converter: This section converts the obtained voice commands to text and matches with the command set.
- Text to Speech converter: This section converts the obtained text to speech.
- Command Set: This section contains all the commands user can give to the web browser and given commands are matched here. If it matches then specific action is performed else control is sent back to browser.
- Performs action: This section performs the specified action and displays it on the browser.

These are the commands in the browser:

Table 1: Speech commands

Command	Description
Go	It will navigate to the entered URL
Back	It will open previous web page
Forward	It will open previous web page
Refresh	It will refresh the web page
Home	It will open home URL
Speak	It will take voice input
Listen	It will record the voice
Stop	It will terminate the process
Links	It will speak link titles of the webpage

IV. IMPLEMENTATION

Visual Studio 2015 will be used to build the browser using the C# programming language. Speech is an effective way of interacting with applications completely hands-free. It is useful for the handicapped and the visually impaired users. Here, the web browsing would be voice based and a hands-free application for such users.

Benefits of using the "*System.Speech*" framework in C# are: [1] [2] [3]

1. Speech Recognition: The System.Speech.Recognition framework helps with the speech input. The benefits of Speech Recognition are as follows:

a) *Speech Input*: It takes any audio or voice as input for the application and notifies if there are any errors in the program

b) *Grammar*: Using the GrammarBuilder and Choices classes we can provide set of cases or alternatives we would like to have in our program. The choices given in the Choices class could then be implemented in switch cases or if-else statements.

c) *Events*: Events like recognizing speech is done by the application. For example the SpeechRecognized raises events when speech is detected with proper volume and frequency levels.

d) *Recognition Engines*: Here you can work around the configuration of the input, enable and disable recognition or change certain properties of the engine that affects the recognition process.

2. *Speech Synthesis (Text-to-Speech):* This feature converts the textual words on the screen to speech. The volume and speed of the speech can be regulated as per requirements.

a) *Create TTS contents (or Prompts):* Whatever the engine speaks is a prompt. The PromptBuilder class can be used for text to speech and also regulate speed and volume of the speech.

b) *Manage the Speech Synthesizer:* with this, the user can select a speaking voice, specify output, handlers for events, start, pause, record or continue the speech. The speaking could either be asynchronous using SpeakAsync().

c) *Control voice characteristics:* The properties of the output speech can be controlled viz. the gender of the speaker, rate, volume of the speech, etc.

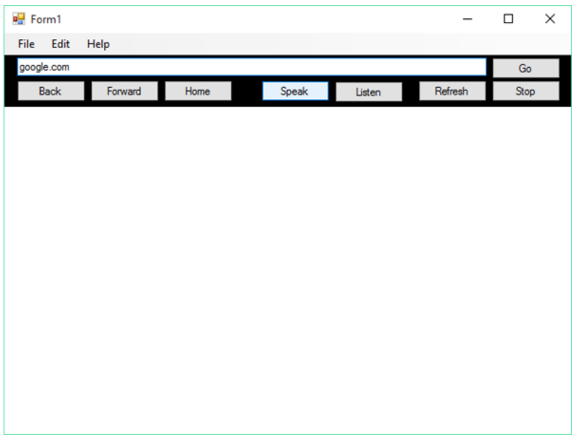


Fig 2: Voice based input to the browser

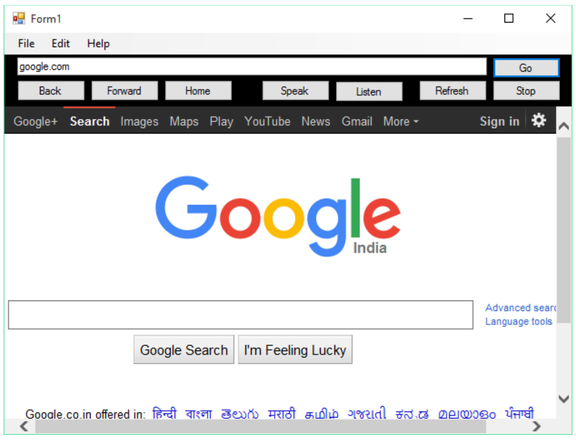


Fig 3: Opening of the web page on "Go" (speech command)

Problems related to the voice interface [4] [7]

With the excellent tools and technologies available like SALT (Speech Application Language tags) with ability to plug-in the right places to develop speech enabled applications, at least at first glance look as easy as web page creation. However, in spite of many excellent tools available today, web pages that are difficult to understand are hard for developers to spell bound.

1. Organization before implementation
To construct any system or software, organization and analysis of its plan is an important task. With considerations to be taken of user and functional requirements. Therefore, construct a road map to the existing applications by knowing the need of amendments in the current scenario. Information for the analysis may be available in the Internet or can be gathered manually.
However it not really easy to build the speech based applications, which means the environment plays an important role in the running of an application. As the level of automation increases in an application the level of accuracy falls, this marks the basic principle of speech technologies. This simply does not mean that the developer should disregard the principles of design related to speech applications and their interfaces.

2. Prompt Clarity
Listening is a difficult task both for the user and the machine. Assuming that the user will be in an environment where the noise around him/her will be minimal and the foreground sound signal will be clear is a mistake. Environment cannot be defined since it is dynamic in real world and changes arbitrarily. Thus the system needs to adapt to the changing environments in which the user is.
For example, a locomotive conductor works in an entirely different environment than an office worker. Likewise, a balance inquiry system for a bank will have a completely different sense of urgency compared to a city's emergency line. By using different sentence frames, pausing between interval points, and perhaps most importantly, by employing the right voice character and voice talent, one will be able to develop resistance to prompts and background noise that truly facilitate listening.

3. So Many Things to Say
Listening is a difficult task at best. It is natural that humans tend to speak a lot and elaborate on everything they want to convey. Thus for the speech recognition machine, listening and recognizing long conversations and converting it to text would become impossible. Thus user must speak with appropriate speed with pauses allowing the recognition engine the time to convert the so far listened speech to text.

Use of Web Crawler
A web crawler (otherwise called a web arachnid or web robot) is a system or mechanized script which peruses the Internet in an efficient, computerized way. This procedure is called Web creeping or crawling.

Numerous honest to goodness destinations, specifically web indexes, use crawling as a method for giving avant-garde information.

Web crawlers are for the most part used to make a duplicate of all the visited pages for later preparing by an internet finder that will file the downloaded pages to give quick quests.

Crawlers can likewise be utilized for computerizing upkeep errands on a Site, for example, checking interfaces or accepting HTML code.

Likewise, crawlers can be utilized to accumulate particular sorts of data from Site pages, for example, collecting email addresses (ordinarily for spam).

Here the web crawler is used to harvest URLs and URL titles from the web pages so as to give the collected data to the prompt builder to speak the necessary URL titles for the blinds. To crawl through the webpages, use of regular expressions to match the desired patterns of results is used most of the times when working with C#.

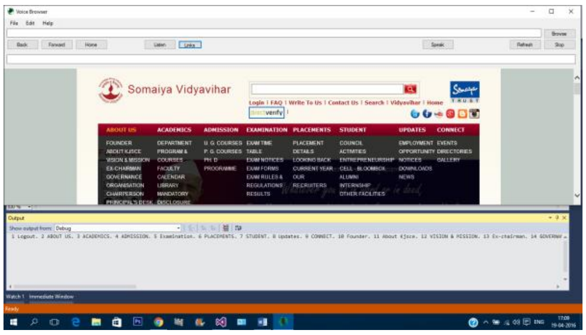
This concept can also be extended to extract downloadable files, images, phone/fax numbers etc.

1. Use of Regular Expression[8]
C# language does support regular expressions through various classes in the *System.Text.RegularExpression* namespace in the .NET framework. To be able to use the regular expression classes one must import *using System.Text.RegularExpression;* namespace in the source file.

The *Regex* Class in the namespace *System.Text.RegularExpression* allows to perform string matching and extracting useful information from text with its interface of Regular Expressions.

Thus, here the text is the HTML webpage source out of which we extract URLs and URL titles.

Here to find all the URLs or URL titles, a global search or a complete search over the webpage source code needs to be done and accumulate the results. To do this a static method *Regex.Matches()* is used to match the strings found and to collect all the matches *MatchCollection* can be used which can be iterated and processed over.


Voice Browser
File Edit Help
Back Forward Home Listen Links Speak Refresh Stop Browse
Somaiya Vidyavihar
ABOUT US ACADEMICS ADMISSION EXAMINATION PLACEMENTS STUDENT UPDATES CONNECT
Output
Show output from: Debug
Watch 1 Immediate Window
Ready

```
1 Logout. 2 ABOUT US. 3 ACADEMICS. 4 ADMISSION. 5 Examination. 6 PLACEMENTS. 7 STUDENT. 8 Updates. 9 CONNECT. 10 Founder. 11 About Kjsce. 12 VISION & MISSION. 13 Ex-chairman. 14 GOVERNA
```

Fig 4: Extraction of URL titles using regular expression by voice command “Links”

V. CONCLUSION

Access to information has become a major economic and social factor. Voice browsing technology is a rapidly-growing field. Whether or not it proves to be the next internet, it deserves a careful examination in its present form, as the need for an easy and direct way to access the internet has become a demand for many types of people, especially the handicapped. From this, the idea of the project was raised in order to help implement a voice based web browser.

ACKNOWLEDGMENT

We are grateful for the support and guidance of our mentor **Prof Grishma Sharma** for the in-depth knowledge given to us in the implementation of the project and leading us in the correct direction for building the project.

REFERENCES

[1] Benefits of System.Speech: https://msdn.microsoft.com/en-us/library/office/hh361629(v=office.14).aspx
[2] Speech Recognition: https://msdn.microsoft.com/en-us/library/office/hh361633(v=office.14).aspx
[3] Text to Speech: https://msdn.microsoft.com/en-us/library/office/hh361644(v=office.14).aspx
[4] Voice user interface design: https://msdn.microsoft.com/en-us/library/ms994651.aspx
[5] 2006 paper: Voice Browsing Approach to E-Business Access: A Blind's Perspective: http:// www.ccsenet. Org /journal/index.php/cis/article/viewFile/7169/6092
[6] Speech recognition for visually impaired: http://wseas.us/elibrary/conferences/2007venice/papers/570-546.pdf
[7] Voice User Interface Design - Tips and Techniques – MSDN https:// msdn.microsoft.com/en-us/library/ms994651.aspx
[8] Regex Class: https://msdn.microsoft.com/en-in/library/system.text.regularexpressions.regex.aspx