SOURCE CODE GENERATOR USING SPEECH

[1]Akshata V Kulkarni, [2]Deekshitha R, [3]Fanoos Fathima, [4]Dr. R.Kanagavalli

[1,2,3,4]Department of Information Science and Engineering,

[1,2,3,4]The Oxford College of Engineering, Bengaluru, India

ABSTRACT: Researchers have shown that most effort of today's software development is maintenance and evolution. Developers often use integrated development environments, debuggers and tools for code search, testing, and program understanding to reduce the tedious tasks. Interaction with software development environments can be frustrating for the growing numbers of developers who suffer from repetitive strain injuries (RSI) and other disabilities that make typing difficult or impossible. Speech interfaces can be used to help developers reduce their dependence on typing, reducing the onset of RSI among computer users, and increasing access for those who already have motor disabilities. We have proposed a system to generate code automatically based on speech. The speech input is processed and based on that the code will be automatically generated.

Key Words: Speech to text, code generation, mapping the text, Speech Interface

I.INTRODUCTION

Speech Recognition, it is the ability of the machine or program to evaluate word, idiom or a sentence in spoken expression and convert those words into a machine readable format. The more sophisticated software has the ability to obtain natural language as well. Speech recollection works using algorithms through acoustic and language modeling. In addition, acoustic modeling represents the link between linguistic units of speech and audio signals; whereas language modeling matches sound with string to help categorize between words that sound similar. Additionally, Hidden Markov Models are used as well to make materialistic patterns in a speech to enhance accuracy with the system. Furthermore, it is seen that a person working on a computer cannot work or type for longer duration because if they can then there will be an issue of back or wrist pain that will be pernicious for the human body, but it can be avoided easily by switching from typing to speaking whenever needed. Research has shown that more than 60% of software engineering resources are spent on maintenance. Software maintenance is the process of modifying a software system after delivery to fix bugs, improve performance, or adapt to a changing environment. Software maintenance requires code comprehension, as reading and understanding source code is the prerequisites of any modification. Program comprehension is time-consuming and cost most of developers' time.

II.PROBLEM STATEMENT

Code Generation techniques depends on textual documentation which are highly prone to syntactical errors. Manually defining every programming construct is tedious and time consuming. There is a lack of flexibility and efficiency of software developers.

PROPOSED SYSTEM

The proposed system helps in generating the source code and also compiling the code to check for various errors and bugs, which helps in reducing the time taken to code by the developers. The proposed system allows the user to be able to generate the syntax of various inputs that helps reduce the time taken for developing a project. Speech to text conversion is done using an android app. Process the text using NLP by using NLP server. The text generated is mapped with the corresponding instructions. Finally, generated code is added in the appropriate part of the program.

Advantages:

- Easy to code
- Less time for the development phase
- Cost efficient
- Bug-free code
- Programming-by-voice can enable motor-impaired software engineers to program, albeit at reduced efficiency compared to an unimpaired programmer.

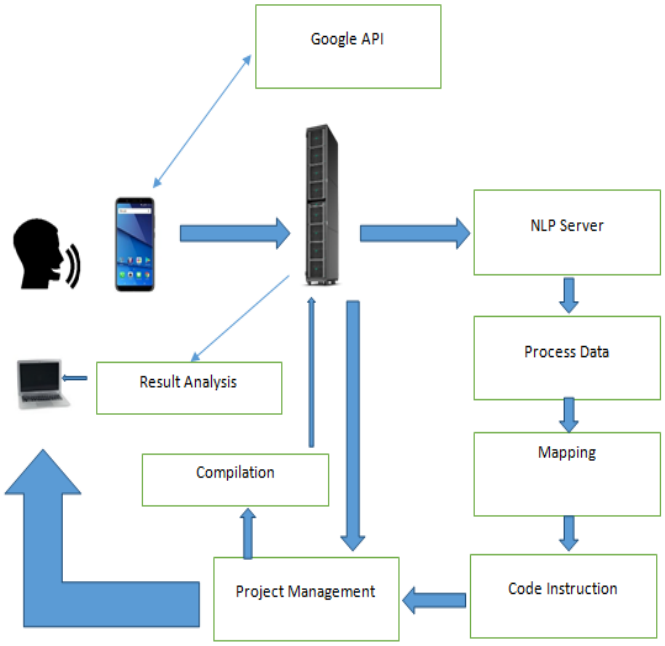

Fig: System Architecture of the proposed system

III.RESEARCH METHODOLOGY

3.1 Modules Description

Speech Processing:

The speech input is processed using Google API and is converted into text. The generated text is then sent to the Tomcat server and further sent to the NLP server.

The communication between the Android app and the Tomcat server is done using HTTP protocol. The user can work on multiple projects from the same app. Based on the selected project, the app sends the request to the web server and the server on receiving the request starts the respective project.

NLP Server

The NLP server starts listening on a particular port number. Once the server receives a request from the user, it sends the instruction to the NLP server to process it. Once NLP server receives the input from the web server, it processes the input to find out the part of the speech. Based on the process input, it starts the next process that is mapping. If there is no pre-defined command, it invokes the machine learning algorithm like naïve based classification and finds the appropriate command.

A white point is a set of values or that serve to define the color "white" in image capture, encoding, or reproduction. It is used to calculate the traffic density by comparing the number of white pixels to the number of black pixels. This gives an estimation of the traffic density in the lane.

Mapping:

Once the mapping process is invoked, it tries to map the input with the predefined instructions. If it finds exactly the matched instruction, it adds the code segment into the project or to a respective class. As we are handling object oriented programming, user has simple and easy-to-use instructions and the details are found in the app itself.

For example:-

a) If the user wants to create a new class he/she can just say "create a class" then the mapper will send the instruction to the server to ask for the name of the class. User can tell the name of the class and based on that, the mapper will create a class with the given name. If a class with the same name is already present in the same package of the project, then server will not create any class for it and sends the notification to the user.

b) if the user gives the input like, "create a method with the name 'xyz' with no return type" then the app sends the instruction to the server to create a method with the name xyz which does not return any type and server adds it in the current class.

Like this, the user can create different types of methods with parameters or without parameters, with different return types. The mapper is capable of handling few features of object oriented programming like creating a class, handling inheritance and polymorphism, different logic like- if else, for loop, while loop, do-while loop and many more.

The structural setup methodology is wired with working up a fundamental essential framework for a system. It incorporates perceiving the genuine parts of the structure and exchanges between these fragments. The starting design technique of perceiving these subsystems and working up a structure for subsystem control and correspondence is called development demonstrating plot and the yield of this framework method is a depiction of the item basic arranging. The proposed design for this framework is given beneath. It demonstrates the way this framework is outlined and brief working of the framework.

3.2 Result Analysis

In this section, we discuss the time taken for the development time of source codes. The development time has been readily decreased with the use of speech as input. The dataset required for the code generation is mapped to a dictionary for speech recognition. The dataset contains readily available syntaxes that are required for the code generation and hence the time consumed is considerably decreased. The syntax is automatically generated and the corresponding packages that are needed for the java programming is also called. Bug free source codes can thus be generated by using this system.

3.3 Future Work

Since the system developed uses only one programming language and the generation of only that language's code/syntax is done. In the future work, we can integrate multiple programming languages and their functionalities. Java may also contain ambiguity with respect to variables and the interfaces used in the programs which creates a limitation for automatically generate the code.

IV. CONCLUSION

Speech-based programming also may provide insight into better forms of high-level interaction. With more study, different user interface designs, and better analysis, software developers will one day be able to use speech-based programming to compete effectively in the workforce. The system will be able to add pre-defined logic automatically. The proposed application will be able to generate code based on speech as an input by recognizing the speech, converting to text and then map it to the pre-defined code and thus ad the corresponding code.

V. ACKNOWLEDGMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible whose constant guidance and encouragement crowned our effort with success. We would like to express our gratitude to Dr. Praveena Gowda, Principal, The Oxford College of Engineering for providing us a congenial environment and surrounding to work in. Our hearty thanks to Dr. R. Kanagavalli, Professor & Head, Department of Information Science and Engineering, The Oxford College of Engineering for her encouragement and support. Guidance and deadlines play a very important role in successful completion of the project report on time. We convey our gratitude to Dr. R. Kanagavalli, Professor & Head, Department of Information Science and Engineering for having constantly monitored the completion of the Project Report and setting up precise deadlines. Finally a note of thanks to the Department of Information Science and Engineering, The Oxford College of Engineering, both teaching and non-teaching staff for their cooperation extended to us.

REFERENCES

[1] Juan Zhai, Lin Tan, Feng Qin-Automatic Model Generation from Documentation for Java API Functions.978-1-4503-3900-1/16/05…$15.00©2016 IEEE
[2] Kaveendra Lunuwilage, Thelijjagoda ,Sameera Abeysekara -Web Based Programming Tool with Speech Recognition for Visually Impaired Users.978-1-5386-4602-1/17/$31.00©2017 IEEE
[3] Fagui Mao,Xuyang Cai,Beijun Shen,Yong Xia-Operational Pattern Based Code Generation for Management Information System: An Industrial Case Study.978-1-5090-2239-7/16/$31.00 copyright 2016 IEEE
[4] Lutfi Kerem Senel, Ihsan Utlu ,Veysel Yucesoy.-Semantic Structure and Interpretability of Word Embeddings.2329-9290©2018 IEEE
[5] Takafumi Moriya, Tomohiro Tanaka-Evolution- Strategy-based Automation of System Development for High-Performance Speech Recognition. 2329-9290 © 2018 IEEE
[6] Shahab Nadir, Prof. Detlef Streitferdt -Software code generator in automotive field 978-1-4673-9795-7/15 $31.00 © 2015 IEEE
[7] Victor Guana, Eleni Stroulia-ChainTracker: Towards a Comprehensive Tool for Building Code-Generation Environments. 1063-6773/14 $31.00 © 2014 IEEE
[8] GlotNet—A Raw Waveform Model for the Glottal Excitation in Statistical Parametric SpeechSynthesis
Lauri Juvela ; Bajibabu Bollepalli ; Vassilis Tsiaras ; Paavo Alku
[9] An Efficient Framework for Sentence Similarity Modeling Zhe Quan ; Zhi-Jie Wang ; Yuquan Le ; Bin Yao ; Kenli Li ; Jian Yin
[10] Semantic Speech Retrieval With a Visually Grounded Model of Untranscribed Speech Herman Kamper ; Gregory Shakhnarovich ; Karen Livescu IEEE/ACM Transactions on Audio, Speech, and Language Processing