VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI-590018

A PROJECT REPORT
ON
PARSELTONGUE

Submitted in partial fulfillment of the requirements for the award of degree,

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE & ENGINEERING

Submitted By

1.	**KANIKA DHYANI**	**1MJ16CS722**
2.	**SAURAV MISHRA**	**1MJ16CS745**
3.	**ABHIRAM BVS**	**1MJ16CS004**
4.	**LAVANYA S**	**1MJ16CS068**

Under the Guidance of

Mr. Karthick Myilvahanan
Associate Professor,
Department of Computer Science & Engineering

MVJ COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow

Approved by AICTE, New Delhi
Affiliated to VTU, Belagavi
Recognized by UGC under 2(f) &12(B)
Accredited by NBA & NAAC

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MVJ COLLEGE OF ENGINEERING
Whitefield, Near ITPB
BANGALORE-67

Academic Year 2019-20

MVJ COLLEGE OF ENGINEERING

Near ITPB, Whitefield, Bangalore-67

Department of Computer Science and Engineering

COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow

Approved by AICTE, New Delhi
Affiliated to VTU, Belagavi
Recognized by UGC under 2(f) &12(B)
Accredited by NBA & NAAC

Certificate

This is to certify that phase II of the project work, entitled "**PARSELTONGUE**" is a bona fide work carried out by

1.	**KANIKA DHYANI**	**1MJ16CS722**
2.	**SAURAV MISHRA**	**1MJ16CS745**
3.	**ABHIRAM BVS**	**1MJ16CS004**
4.	**LAVANYA S**	**1MJ16CS068**

in partial fulfillment for the award of degree of Bachelor of Engineering in Computer Science & Engineering of the Visvesvaraya Technological University, Belagavi during the academic year 2019-20. It is certified that all the corrections/suggestions indicated for Internal Assessment have been incorporated in the Report. The project report has been approved as it satisfies the academic requirements.

Signature of Internal Guide	**Signature of HOD**	**Principal**
Mr. Karthick Myilvahanan **Assistant Professor** **Department of CSE** **MVJ College of Engineering**	**Dr. K.S. Arvind** **HOD of CSE** **MVJ College of Engineering**	**Dr. P Mahabaleswarappa** **MVJ College of Engineering**

Name of the examiners:

1. Signature with date
2.

MVJ COLLEGE OF ENGINEERING

Whitefield, Near ITPB, Bangalore-67

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow

Approved by AICTE, New Delhi
Affiliated to VTU, Belagavi
Recognized by UGC under 2(f) &12(B)
Accredited by NBA & NAAC

DECLARATION

We **KANIKA DHYANI, SAURAV MISHRA, ABHIRAM BVS** and **LAVANYA S** hereby declare that the entire work of project titled "**PARSELTONGUE**" embodied in this project report has been carried out by us during the 8th semester of BE degree at MVJCE, Bangalore under the esteemed guidance of **Mr. Karthick Myilvahanan,** (Associate Professor, Dept. of CSE, MVJCE) affiliated to Visvesvaraya Technological University, Belagavi. The work embodied in this dissertation work is original and it has not been submitted in part or full for any other degree in any University.

1. **KANIKA DHYANI (1MJ16CS722)** _____
2. **SAURAV MISHRA (1MJ16CS745)** _____
3. **ABHIRAM BVS (1MJ16CS004)** _____
4. **LAVANYA S (1MJ16CS068)** _____

Date: 20/08/2020

Place: Bangalore

ABSTRACT

Dragonfly is a speech recognition framework. It is a Python package which offers a high-level object model and allows its users to easily write scripts, macros, and programs which use speech recognition. Dragonfly was written to make it very easy for Python macros, scripts, and applications to interface with speech recognition engines. Its design allows speech commands and grammar objects to be treated as first-class Python objects. This allows easy and intuitive definition of complex command grammars and greatly simplifies processing recognition results. Dragon NaturallySpeaking (DNS) is a speech recognition software we will be using to give speech input. Dragonfly uses Natlink to communicate with DNS. The code that will be entered by the users will be entered using the continuous speech mode supported by the Dragon NaturallySpeaking.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of people who made it possible, success is the epitome of hard work and perseverance, but steadfast of all is encouraging guidance.
So, with gratitude we acknowledge all those whose guidance and encouragement served as beacon of light and crowned our effort with success.

We thank our Principal, **Dr. P Mahabaleswarappa** and Vice Principal, **Prof. M. Brindha** for their constant support and guidance.

We are also thankful to our beloved HOD, **Dr. K.S. Arvind** for his incessant encouragement & all the help during the project work.

We consider it a privilege and honor to express our sincere gratitude to our guide **Mr. Karthick Myilvahanan**, AP, Dept. of CSE for her valuable guidance throughout the tenure of this project work, and whose support and encouragement made this work possible.

It is also a great pleasure to express our deepest gratitude to the other entire faculty members of our department for their cooperation and constructive criticism offered, which helped us a lot during our project work.

Finally, we would like to thank all our family members and friends whose encouragement and support was invaluable.

Thanking you

KANIKA DHYANI (1MJ16CS722)

SAURAV MISHRA (1MJ16CS745)

ABHIRAM BVS (1M16CS004)

LAVANYA S (1MJ16CS068)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF GRAPHS

CHAPTER 1
INTRODUCTION

1.1 PROJECT OVERVIEW

Speaking through the ages has been the most efficient way to communicate and get ideas and thoughts across without being misinterpreted. And so, it is not surprising that speaking to machines is not very unfamiliar in today's world. From leading software giants competing for the best home assistants to security firms employing voice recognition. Speaking to machines is becoming a common practice nowadays. Just as you can command your phone to make a call or save a reminder, imagine being able to command a machine to code. The possibilities are endless.

Voice coding is using your voice as an input to write code. It has a huge and unrealized potential. Typing code can sometimes be frustrating and so being able to code by voice gives programmers flexibility to switch between typing and speaking. This makes coding more efficient and hassle free. Voice coding underlies a wide variety of science and any researcher who writes code could use it.

1.2 PURPOSE OF PROJECT

- Parseltongue is an attempt in popularizing the use of code dictation
- It is an environment to make code dictation easy and make the process of writing codes efficient
- This can be implemented on any computer device and used by anyone who codes or wants to start coding
- Parseltongue aims to be easy to use and habitual to switch to for coders. Physical limitations can and should not cage anyone's mental potentials

1.3 SCOPE OF PROJECT

The basic logic behind Parseltongue is to enable a user to dictate code to the Parseltongue editor. The user can dictate commands to execute, save or use any other functionality of the editor as well.

1.4 PROBLEM STATEMENT

Some software developers may have physical limitations to use their full mental potential. For these developers, the usage of keyboard and mouse may not be very feasible. Weather it may be developers who have been coding for years and suddenly could not do it anymore or beginners who want to learn and discover the universe of software development.

When the world is moving towards neutral networks and AI, it's necessary that the communication with any machine becomes more natural and surpasses these physical limitations. The simple concept if coding by using voice command may sound very acceptable but indeed is a myth.

Voice coding can not only enable the genius coders who have lost their ability to code but not the immense creative minds, but also enable us to communicate with machines in a more natural way. It will help in bridging the gap between humans and machines and the complex human emotion of yes, maybe, not sure etc.

1.5 EXISTING SYSTEM

Voice coding has been around for a while but has never been popular. The reasons being the highly personalized code bases that people usually make for their personal use. These codebases may be available online but understanding and implementing them on systems is a hassle.

1.5.1 Disadvantages of Existing System

- First and foremost, complex to understand.
- Might take time and a prior knowledge of the subject matter to set it up
- Getting to know the whole codebase may be very difficult and hence the potential of the system may not be realized.
- Software requirements and the layout is too complex.
- Too many commands may overwhelm anyone new to coding.

1.6 PROPOSED SYSTEM

We introduce a text editor that enables users to use code dictation. This editor is capable of everything that the existing system offers and more without the hassle of too many software and the intimidation of a huge code base.

1.6.1 Advantages of Proposed System

- Easy to install and use.
- Takes care of all the background requirements so you can focus on coding.
- User interface extremely easy to navigate.
- Easter eggs to encourage codding for beginners.
- Easy to switch between coding with voice or typing.

1.6.2 Objectives

- The objective is to reduce the amount of typing done by the user to input data.
- To make it easier to set up the environment and start coding.

CHAPTER 2
LITERATURE SURVEY

2.1 OVERVIEW

Literature study is carried out to get all the related information of current project, which is used to get an idea for the enhancement as well as changes that can be made to improve existing approaches. A literature study is done on sentimental analysis, intent analysis and many more. Following section describes about all the related papers which is used in current project.

2.2 RELATED PAPERS

2.2.1 PAPER 1

Title

Programming by Voice, VocalProgramming

Year

2000

Abstract

A system that enables a person to program without typing is needed because of the high incidents of repetitive stress injuries among people who program. This paper presents a design for a system that generates environments that enables people to program by voice and a method of determining if the system is successful. It also shows how this generator can be used to support entering data and writing XML documents.

We have designed a generator for voice recognition syntax-directed programming environments. This generator, VocalGenerator, takes as input a context free grammar (CFG) for a programming language, such as Basic, C, C++, Java, or an XML DTD or XML Schema, together with a voice vocabulary for that language. The voice vocabulary includes the literals from the programming language; the names of classes and functions from the class and function libraries available to the programmer; and a list of class, function, and variable names specific to

the program file. These are associated with a list of pronounceable words and phrases that the programmer would use to enter and edit a program in the language.

VocalGenerator generates as output a programming environment in which the programmer can write programs by voice input alone. This programming environment includes a voice recognition syntax-directed program editor. This voice recognition syntax-directed program editor aids the programmer by providing automatic completion of program text and appropriate navigation relative to the specific programming language. A voice recognition syntax-directed program editor can also be used to edit programs that have already been developed. It analyzes a given program and generate the vocabulary used in the specific program.

We refer to the generated voice recognition syntax directed programming environments with names, such as VocalBasic, VocalC, VocalC++, or Vocal Java. Once VocalGenerator has been built it can be used several times to generate programming environments that support voice recognition syntax-directed programming in specific programming languages. Relatively little effort is required to prepare the input grammar and voice vocabulary for each additional programming language.

A voice recognition syntax-directed program editor makes it easier to write a program by voice than a standard text editor. Standard voice recognition text editors are not effective in entering and editing computer programs.

Methodology

Syntax-directed editors provide two major capabilities that standard text editors do not: navigation and selection of sections of code based on the structure of the program and automatic completion of program constructs.

A syntax-directed editor, like VocalJava, makes selecting a whole class or program block a simple operation because it recognizes these as structures within the program text. As an example, a programmer could ask to move up two classes and to the first block within the class definition and delete that block. A syntax-directed editor, like VocalJava, would also be able to complete statements based on the beginning of the statement. For example, the word "if" would result in the completion of the structure for an if-statement consisting of "if", an "(", a space for the condition, an ")", an "{", a space for a block of statements, and an "}". A voice recognition

syntax-directed editor has an additional advantage over a voice recognition text editor because of its limited vocabulary. Programming languages have a very small vocabulary compared to natural languages. The voice recognition engine will have fewer words to recognize, so it will have a better chance of getting the word right.

Future Work

The potential users of this type of technology is not limited to computer programmers. A new extensible markup language, XML, will fast become a standard for information interchange on the Internet. The use of XML will be pervasive in e Commerce and on the Web in general within a very short period of time.. An XML Schema is a grammatical definition of a collection of XML documents containing data with the same meaning, structure and representation. The same way a grammar for a programming language defines all well-formed and valid programs that can be written in the language defined by the grammar, an XML Schema defines all well-formed and valid XML documents that can be written in the language defined by the XML Schema. With some modifications, VocalGenerator can be cloned to understand not only BNF definitions of CFGs, but also XML Schema definitions.

Disadvantages

Computer languages are not spoken, they exist only in written form. A significant effort will be necessary to determine how to vocalize these languages. This is an issue that has not been explored yet.

2.2.2 PAPER 2

Title

Web Based Programming Tool with Speech Recognition for Visually Impaired Users

Year

2017

Abstract

Today programming requires to be able to code textually, but it also requires to be visual. Therefore, both of have left visually impaired users in a very difficult position when it comes for

learning and applying programming. There are tools that enable visually impaired users to use computers, but none of them has been specialized to aid the visually impaired users in various programming issues. The aim of this project is to develop a web application, especially for visually impaired people to facilitate to learn programming concepts.

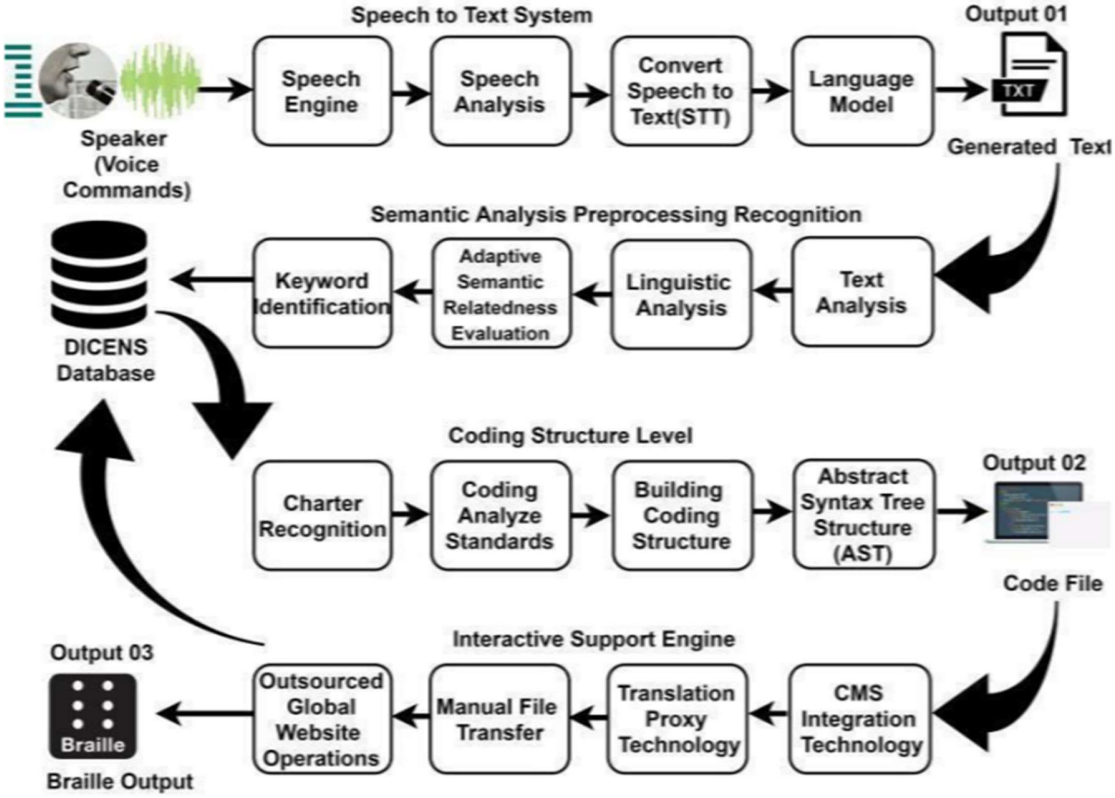


Fig 2.1 - System Diagram of DICENS

Designed solution is an application, based on speech recognition, semantic analysis, user interaction, and braille engine, which can use as a smart programming tool for visually impaired and differently abled people. It' s a method of implementing a new way to help and motivate disable people, to use latest technologies in Information Technology and software engineering aspects.

Methodology

To carry out this research project, the waterfall Model utilized as a methodology all through the System Development Life Cycle (SDLC). The waterfall model is a direct model

which is effective to build up the application considering the fact, that the necessities of the application are clear and does not change with time. Project group members have chosen the waterfall technique after breaking down the necessities of the task. Some reasons have been recognized after there is clear compartmentalization of work and control. The model is basic, straightforward. It is more straightforward to set the timetable for the assignments to be finished

inside the specific time frame. The stages are not covering in this methodology. So, the framework can continue through the improvement procedure and hypothetically, be conveyed on time. Advancement process moves from necessity gathering, planning, execution, testing, and the organization of the framework lastly upkeep.

Future Work

This paper provides possible approaches that are best suitable to develop a speech recognition system to learn programming concepts for blind users. Over the last few decades, advancements in web phone technologies have reached an impressive level. With the advancement of web technology, the society has changed drastically. In the present, almost everything is linked with mobiles in some way. There is definite interest among blind population for programming and computers. There are some associations and foundations that support blind people, including their effort to compete as a programmer. There are existing methods and tools to aid blind in using computers and the same tools are used to aid in programming.

These tools are completely sufficient when talking about textual programming, but, lack features when it comes to visual design. Possible approaches, as well as existing efforts, can be recognized and a step toward the complete solution can be proposed. This research provides a solution to these problems by developing the system "DICENS" to take them forward. This system mainly focused on developing software that acts as an assistant, especially for visually impaired people. “DICNES” is an intelligent system, based on speech recognition, semantic analysis and user interaction system. Capturing and recognizing the given sound input is one of the most important features.

2.3 Paper 3

Title

Voice Recognition Applications for Programming Environments

Year

1989

Abstract

The increased availability of voice recognition products has made it possible to explore more applications areas. Voice input has been traditionally used for hands free data entry, for automation and for the handicapped. In this paper the use of voice recognition for the programmer is explored. Two implementations will be described and analyzed. INTRODUCTION The area of speech analysis has been investigated as a natural form for Human-Machine communication.

Advances in the understanding of speech generation, coding, transmission, and recognition have been underway from the beginning of this century. Mathematical background and detailed presentations are covered in several textbooks such as Shaghnessy (1987) and Lea (1980). Considerations for voice input are mainly to minimize operator learning cycles, reduce potential errors in key entry. Also, human response time for machine interaction and human fatigue should be reduced.

Military systems are one of the areas with complex controls. Human factor design plays a great role in such systems. Several factors are considered in the design of a helicopter control panel. The use of Voice input by the pilots was considered and a study indicated that vibrations of the cabin in a simulated environment did not affect speech recognition (Cruise et al. 1987). In this case the initial concern was the effect of vibration on the pilot's voice and the sensitivity of the vocabulary to such variations in voice. Several applications reported at the National Communications Forum are concerned with recognition of digits transmitted over telephone lines. This area of study is of great importance for applications such as balance inquiries, credit verifications, and catalo orders.

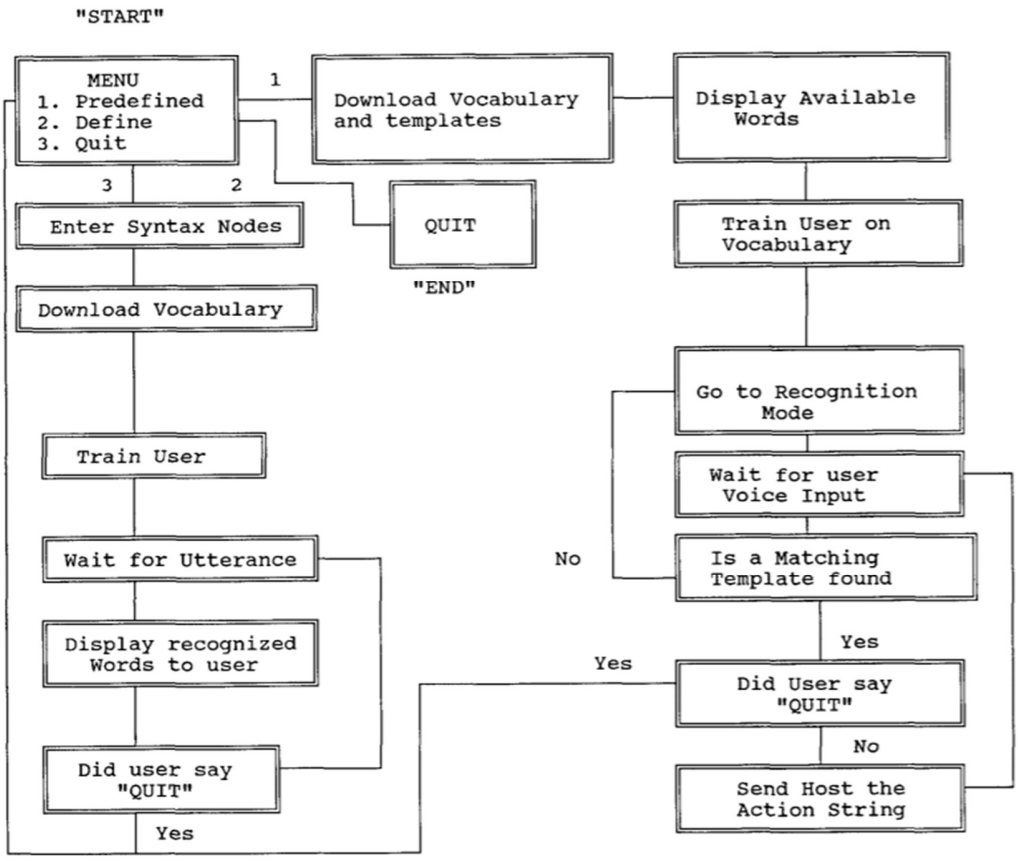


Fig 2.2 – Software Design

Methodology

Design of such a system can be viewed by scribing a sample menu-based approach. gruel shows a simplified diagram of software design. At the top level, the users can select a predefined vocabulary or define their own vocabulary. Then a vocabulary is selected or defined by user is expected to go through an aiming session. In the training session 9 system prompts the user to utter each and repeat it approximately three times. While the user is uttering the various words, the system goes through its defined algorithms and programmable beholds and creates unique templates for utterance. Part of the system support are scoring mechanisms that allow application developer to analyze stances between words to iambize incorrect recognition. Once a selected vocabulary is completed the plates can be stored on the host imputer for later reference, in this case, user's own templates are downloaded to the voice recognition hardware.

Future Work

Based on the presented case studies, one can conclude that the use of voice recognition technology is appropriate and feasible for programming environments. These can include Operating System level commands creating a standard user interface. Also, for a structured language ich as Pascal, voice editing or dictation in such programs are feasible. Additionally, one should not ignore the importance of data acquisition using voice input, since the amounts of data seem to help growing and to arrive in many different formats. Work is needed and is planned to create neural purpose tools to support the creation of Syntax-based recognition. so, integration of

such a tool with her measurement and sensitivity analysis tools to evaluate various sign considerations would be helpful. is should assist an application developer design various system. Finally, while typing the conclusions, started to reach my back and eyes, I wondered if voice input would help, and an infinitely difficult challenge to face full document dictation as compared to a Pascal program.

2.3 Summary

This chapter discuss about the literature study for the proposed system, related paper which gives idea of enhancements to the existing work. The related paper gives information about the existing systems, its advantages and disadvantages.

CHAPTER 3

SYSTEM REQUIREMENT

3.1 MINIMUM HARDWARE REQUIREMENTS

- RAM: 4 GB
- CPU: Intel® dual core or equivalent AMD processor (Faster processors yield faster
- performance)
- HDD: 8 GB for DNS and 150 MB for Parseltongue
- OS: Microsoft Windows 7 or newer
- Sound card: Sound card that supports 16-bit recording
- Microphone: Built-in or External microphone

3.2 SOFTWARE REQUIREMENTS

- Dragon NaturallySpeaking
- Python 2.7 or above
- Python 3.6 or above
- Natlink
- Python Modules
    - Dragonfly 2
    - WxPython
    - PyWin

3.3 ARCHITECTURE

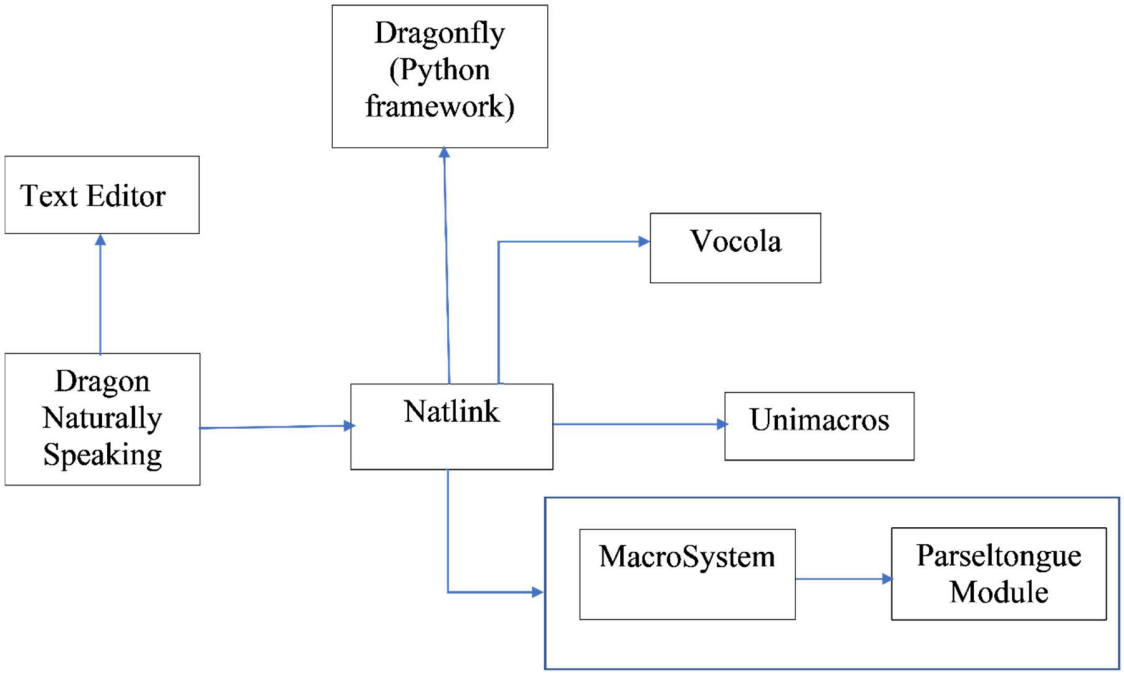


Fig 3.1 – Parseltongue Architecture

PARSELTONGUE EDITOR

The Parseltongue Editor provides the user with an interactive interface to code. It has the basic functionality of any standard editor with an extra feature to run and compile python codes. It is designed keeping in mind the requirements of the user and the ease of navigation.
It can be downloaded from our website easily and is frequently updated to include more features.

DRAGONFLY

Dragonfly is a speech recognition framework. It is a Python package which offers a high-level object model and allows its users to easily write scripts, macros, and programs which use speech recognition. Dragonfly was written to make it very easy for Python macros, scripts, and applications to interface with speech recognition engines. Its design allows speech commands

and grammar objects to be treated as first-class Python objects. This allows easy and intuitive definition of complex command grammars and greatly simplifies processing recognition results.

DRAGON NATURALLY SPEAKING

Dragon NaturallySpeaking (DNS) is a speech recognition software we will be using to give speech input. Dragonfly uses Natlink to communicate with DNS. The code that will be entered by the users will be entered using the continuous speech mode supported by the Dragon NaturallySpeaking.

VOCOLA

Vocola is a voice command language—a language for creating commands to control a computer by voice. Two versions are available: Vocola 2 works with Dragon NaturallySpeaking (DNS) and Vocola 3 works with Windows Speech Recognition (WSR) on Windows 8, 7, and Vista. While DNS and WSR handle the heavy lifting, Vocola (pronounced "vo-CO-luh") concentrates on features and ease of use. Vocola offers the following:

- Easy to use:
- Simple, concise command syntax—most commands are one-liners
- Easy to view and modify commands
- Changed commands are loaded automatically
- Large set of useful sample commands

UNIMACROS

With Unimacro grammars, lots of things can be done without extensive programming yourself. Many things can/must be tuned through.

Some of the other features:

- Use the power of AutoHotkey for actions that are difficult to do within Unimacro
- Going to and do actions with line numbers in IDE programs and for example Excel. For some applications you can choose to jump to a line number module hundred.
- Putting all sorts of brackets around selection or dictated text (nice start example too!)
- Numbers lists as spoken forms

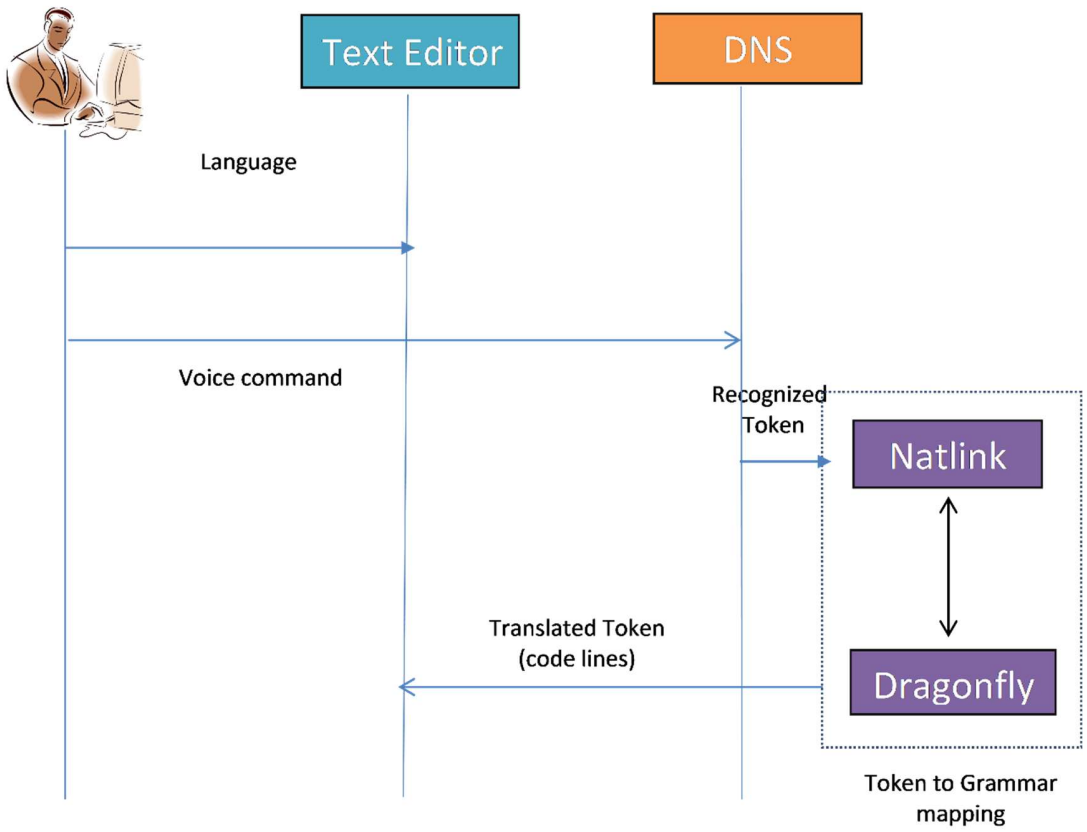


Fig 3.2 – Sequence Diagram

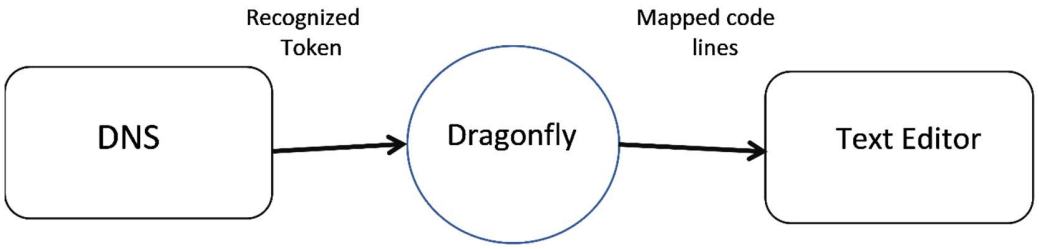


Fig 3.3 – Dataflow Diagram

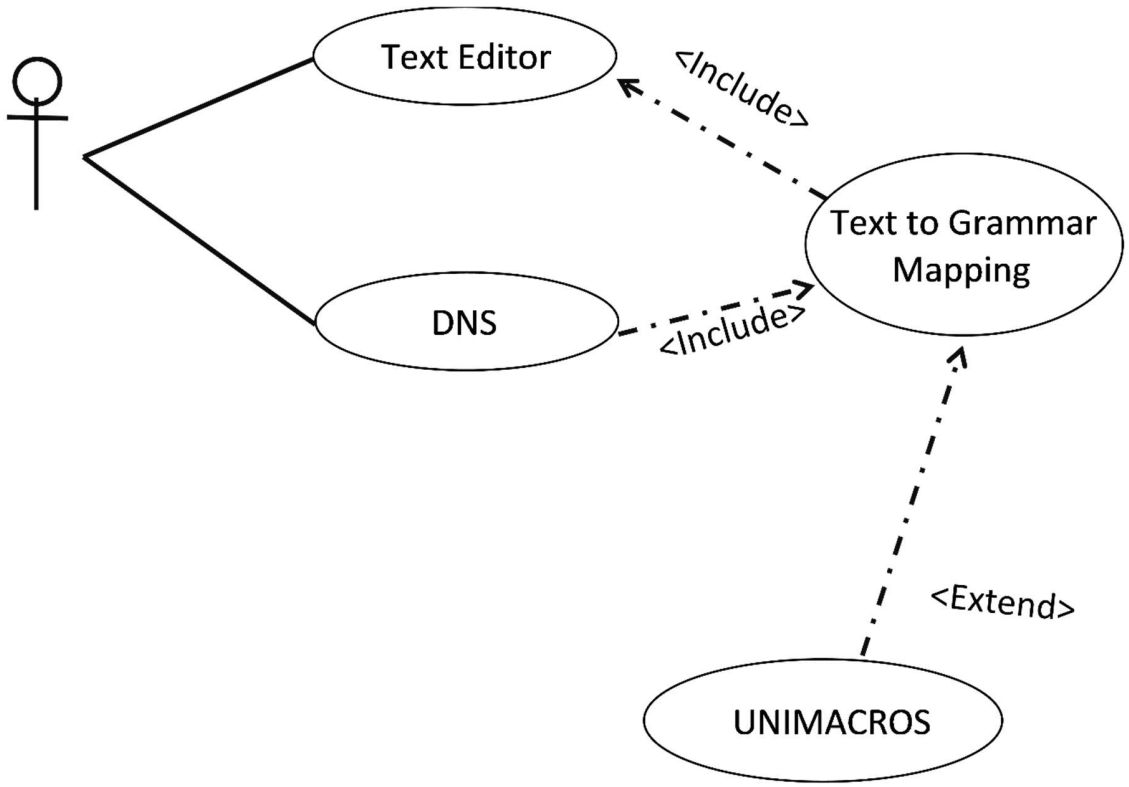


Fig 3.4 – Use Case Diagram

3.4 DEPLOYMENT

Parseltongue requires Parseltongue module to work which contains the command and functions which converts speech to meaning Python 3 syntax. This module is uploaded on "The Python Package Index" or PyPI which is the largest repository of Python packages.

Parseltongue also requires too many software and python module having specific versions to work. So, if a user wants to deploy Parseltongue to his computer, he will have to go through a long process to install and configure all the software one by one which becomes hectic even if the instruction manual is provided. To overcome this issue, our team came up with Parseltongue Bundle, Documentation and a Website.

PARSELTONGUE BUNDLE

Parseltongue Bundle is an installer which is contains Parseltongue Module along with all the required software like Python, Parseltongue Editor or Natlink and Python modules.

This bundle, which is written in Pascal Language, will install the software one by one and configure them as soon as they are installed. If the user already has any of the software which the bundle provides, he can choose the software to be installed from the interface provided using checkboxes. Once the software is installed, it requires internet connection to install python modules required to run Parseltongue, including "Parseltongue Module"

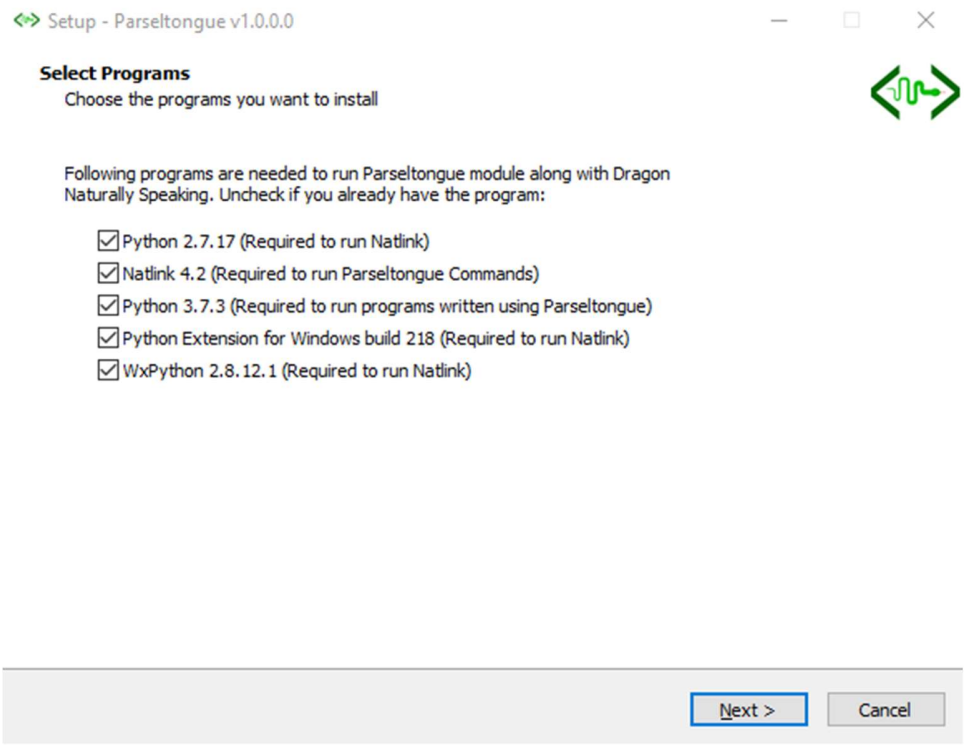


Fig 3.5 – Software to be installed

DOCUMENTATION

Once the Parseltongue is installed successfully and is ready test, user will not know how to start and which commands to use. For this, Bundle installs Parseltongue Editor which is used

as a primary text editor to write Python programs using Parseltongue. This editor contains a documentation which has all the commands which Parseltongue supports. Documentation can also be downloaded from Parseltongue website.

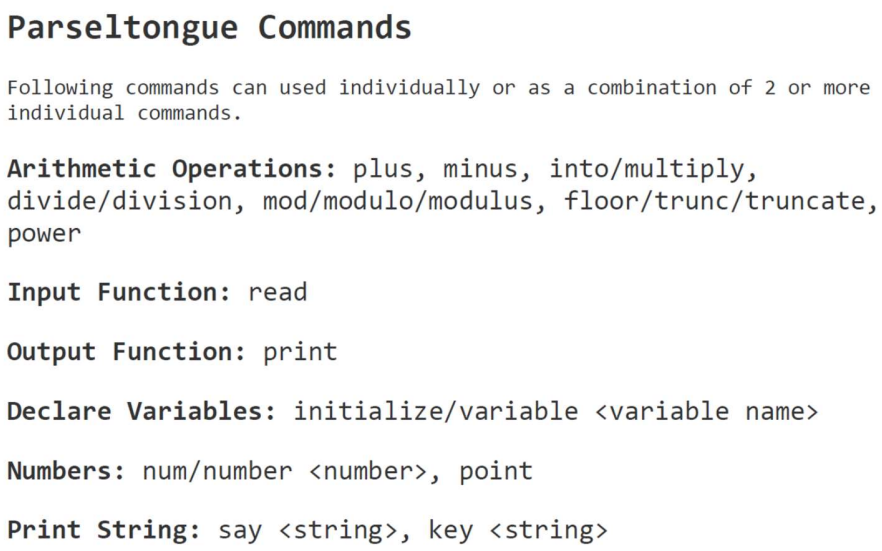


Fig 3.6 – Documentation

WEBSITE

Parseltongue has an official website “parseltongue.ml” which contains a link to download Parseltongue requisites. It contains Parseltongue Editor, Parseltongue Bundle, Documentation and link to official open source Parseltongue module on PyPI. This website also is updated regularly as soon as a new update of one of the requisites of Parseltongue are released or modified

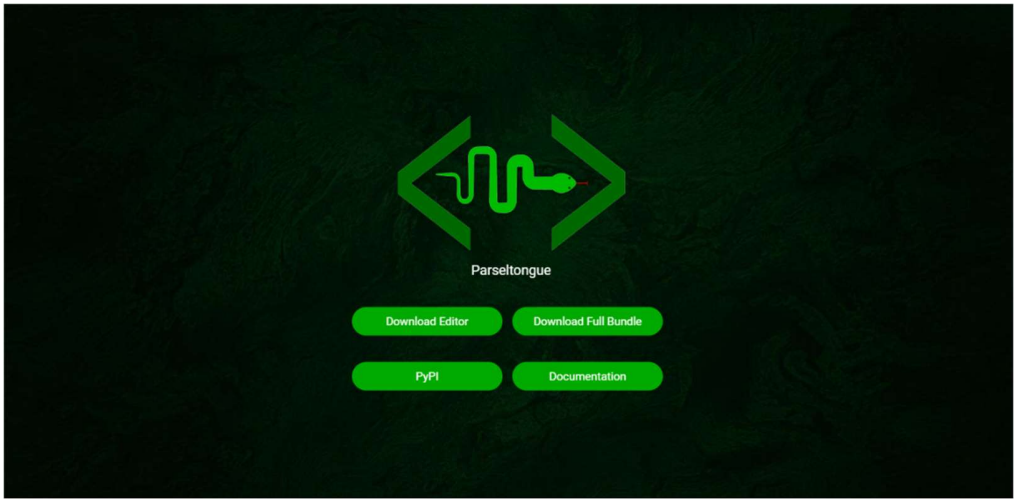


Fig 3.7 – Parseltongue Website

CHAPTER 4

TESTING AND MAINTENANCE

Testing is an important part of project where testing of each module is done. Testing guarantees that proposed system is well organized analysed to meet require project goal. Testing is last stage of project which guarantees the system is error free and ready to give desired output. The goals of testing are given by:

- Give operational quality to system
- Search and remove errors.
- Best quality project is produced.
- To approve the product as a solution for the first issue

4.1 TEST STRATEGY AND APPROACH

4.1.1 Test Objectives

- Working of text editor
- Connection of all the components
- Checking if the voice is recognized or not

4.1.2 Features to be tested

- Proper voice recognition
- Proper working of all the components and modules

4.2 BLACK BOX TESTING

Black box testing is a type of software testing in which the software is tested without the knowledge of any inner functionalities of software. Here the software which is to be tested is considered as a black box that is you just can see the outside look not the inner functionalities.

The test gives some inputs and replies to the output without any Knowledge of the working of the software.

4.3 WHITE BOX TESTING

The white box testing is a type of testing in which the testing part considers the entire inner functionalities of the given software. The white box testing reaches the testing level which the black box cannot do.

4.4 FUNCTIONAL TESTING

Functional testing involves in testing the various functionalities of the software. The Functional testing part mainly involves the valid inputs to the system and the output, or the results obtained from the system and the functions of the system are considered. Here the system procedure is also considered. The functional testing considers or focuses mainly onrequirements.

4.5 TEST CASES

The test cases are the documents maintained by providing inputs and noting the outputs of the given software. A good test case is one which provides the required output without any error in the execution.

4.6 UNIT TESTING

Unit testing is a method of testing, where the whole system is divided into smaller sub systems or units and each unit is tested for correct execution. In this project, the system is divided into two parts and each part is tested separately. Initially we test the working of Text editor by checking it reads the data provided by user. Then we check whether the Natlink, Vocola and dragonfly age starting together once we turn on the DNS.

Functionality which is tested	Inputs	Testing	Observation
Working of text editor	Speech	Accuracy is calculated	Success
Connection of all the modules	Start Dragon NatuallySpeaking	All the modules should be invoked and connected	Success

Table 4.1: Unit Testing

4.7 MAINTENANCE

This is the final stage of the software development life cycle. This is very important to eliminate the errors of the system during working stage. This also maintains the variations in the working environment. This phase takes the review of the system time to time. The review should be done to know the performance of the system, capabilities of the system and to know the additional requirements and changes. If any major changes are required during the maintenance stage, a new system is developed to carry out the changes. The project or system will undergo all the previous steps.

The Python Package Index, abbreviated as PyPI and known as the Cheese Shop, is the official third-party software repository for Python programming language. Since PyPI is used in the project it should be updated. The Text Editor – Parseltongue which is used in the project should also be updated which is built in the editor.

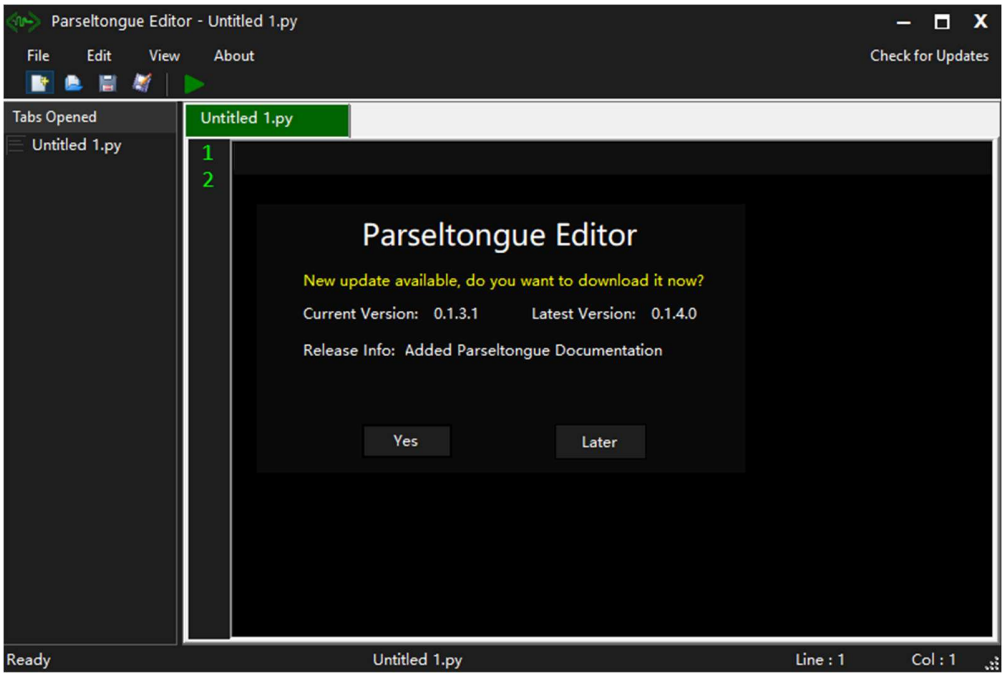


Fig 4.1 - Parseltongue editor (Check for updates)

CHAPTER 5

PERFORMANCE ANALYSIS

5.1 INTRODUCTION

Performance analysis is the technique of studying or comparing the performance of a specific situation in contrast to the aim and yet executed. In Human Resource, performance analysis can help to review an employee's contribution towards a project or assignment, which they allotted him or her.

To check the performance of Parseltongue, we typed our test cases then dictated the same using Parseltongue. It is obvious that speaking is faster than typing but speaking right commands to write the code as compared to typing is difficult. So, we took 2 parameters Accuracy and Speed for comparison and then combined both the parameters based on Speed-Accuracy tradeoff to get an overall performance.

5.2 PERFORMANCE BASED ON SPEED

To test the performance based on speed, we took 5 beginner level programs as test cases and then wrote each one of them by typing and by dictating using Parseltongue.

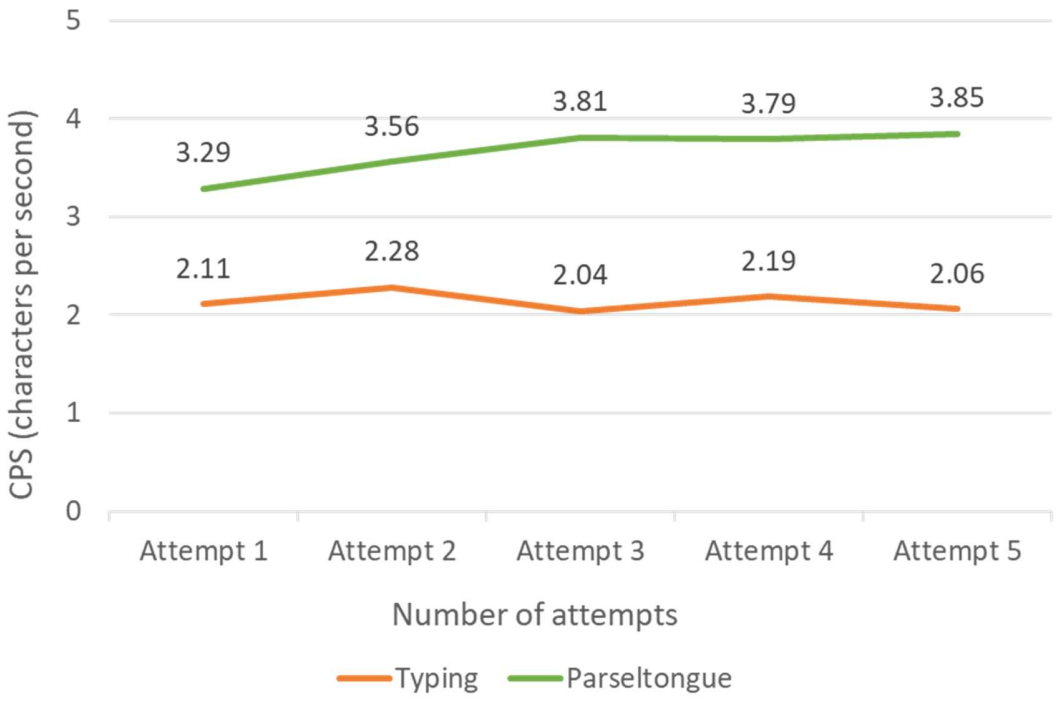
Each team member tested each program using both the methods 5 times in a span of 1 month from 1st June 2020 to 29th June 2020, i.e. a total of 200 tests. Then for each program, the average of time taken to type, and dictate was calculated. Based on the time taken and number of characters in it, CPS (characters per second) was typing and dictating was calculated.

Final graph shown is plotted by taking the average of CPS for all the programs in each attempt tested by all the team members.

Case\Scenarios	Typing	Dictating	No. of Characters	CPS (Characters per second)		Performance Increase
				Typing	Dictating	
Test Case 1 (Print "Hello, World!")	11 seconds	6.25 seconds	22	2.00	3.52	83%
Test Case 2 (Sum of two numbers)	37.75 seconds	23.25 seconds	85	2.25	3.66	77%
Test Case 3 (Check odd or even)	58.25 seconds	34 seconds	124	2.13	3.65	69%
Test Case 4 (Check Leap Year)	101.5 seconds	60.5 seconds	214	2.11	3.54	75%
Test Case 5 (Check prime number)	131 seconds	74.75 seconds	292	2.23	3.93	61%

Avg. typing speed: 2.14 CPS Avg. dictating speed: 3.66 CPS Avg. Performance Increase: 73%

Table 5.1 – Typing Speed vs Dictation Speed



Graph 5.1 – Speed Comparison Graph

5.3 PERFORMANCE BASED ON ACCURACY

Accuracy of Parseltongue when compared to typing depends on two different parameters: Human's accuracy and Computer's accuracy.

5.3.1 Human Accuracy

Typing

When it comes to typing, human can make many mistakes. Typing accuracy is calculated by taking the percentage of correct entries out of the total entries typed. Typing speed is usually calculated in words per minute (wpm). Average human accuracy for typing is 92% and for a good typist, it is more than 95%. When it comes to programming, programmers' types syntax of the program instead of English language.

Parseltongue

When Parseltongue is being used to write programs, the programmer can make mistakes while dictating using Parseltongue commands. This could be due to lack of knowledge of different commands.

For Parseltongue, human accuracy is calculated by testing correct commands spoken out of total commands spoken.

Train data	Number of test	Tests spoken successfully
print	200	200
while	200	192
define	200	180
read	200	188
for	200	197
variable	200	183

Accuracy: 95%

Table 5.2 – Human Accuracy Test

5.3.2 Computer Accuracy

Typing

When person is typing, computer will not make any mistake in understand the letter typed on the keyboard. User can make mistake in typing the correct letter by computer exactly understands what the user has typed. Hence, we consider this accuracy as 100%

Parseltongue

For Parseltongue, accuracy for computer depends on how good it understands the user's English accent. In Parseltongue, speech is detected and converted to text using Dragon NaturallySpeaking (DNS). Since, DNS tries to understand the speech based on the previous speech commands, accuracy for Parseltongue gradually increases with more and more testing.

Here, computer's accuracy is calculated by testing correct commands understood out of total commands spoken.

Train data	Number of test	Tests detected successfully
print	200	197
while	200	178
define	200	192
read	200	190
for	200	162
variable	200	181

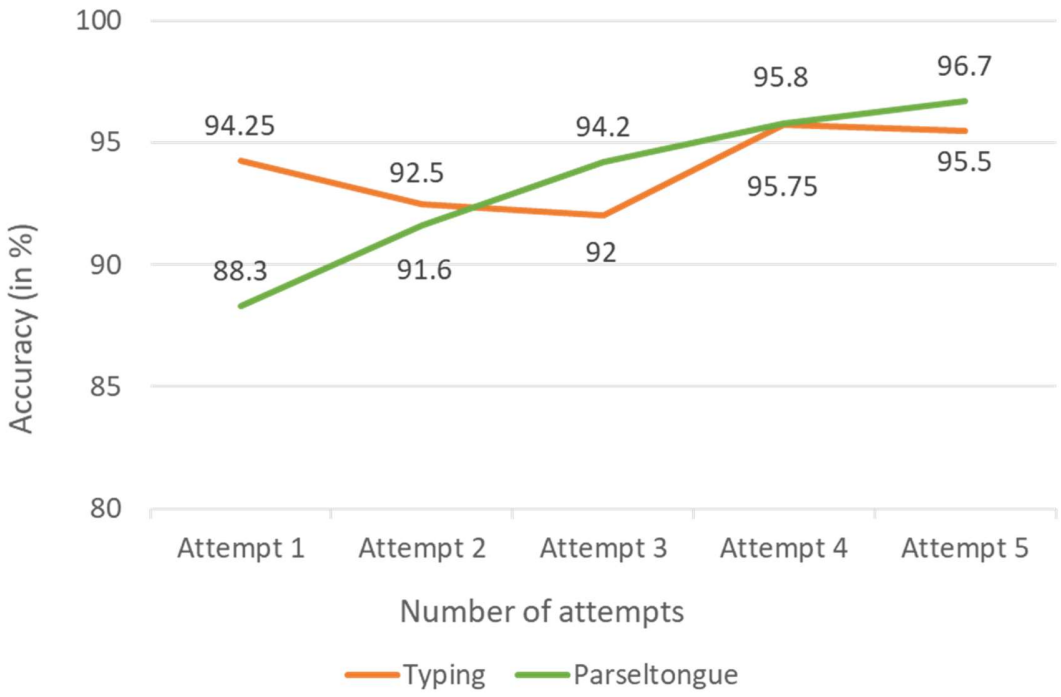
Accuracy: 91.67%

Table 5.3 – Computer Accuracy Test

To test the performance based on accuracy, we took 6 most common python commands and functions which are very different in pronunciation. Then again, these commands were typed and dictated using Parseltongue.

Each team member tested each command using both the methods 10 times in a span of 1 month from 1st June 2020 to 29th June 2020, i.e. a total of 200 tests.

Following graph is measured by taking the average accuracy of all the commands in each attempt for both typing and dictating using Parseltongue.



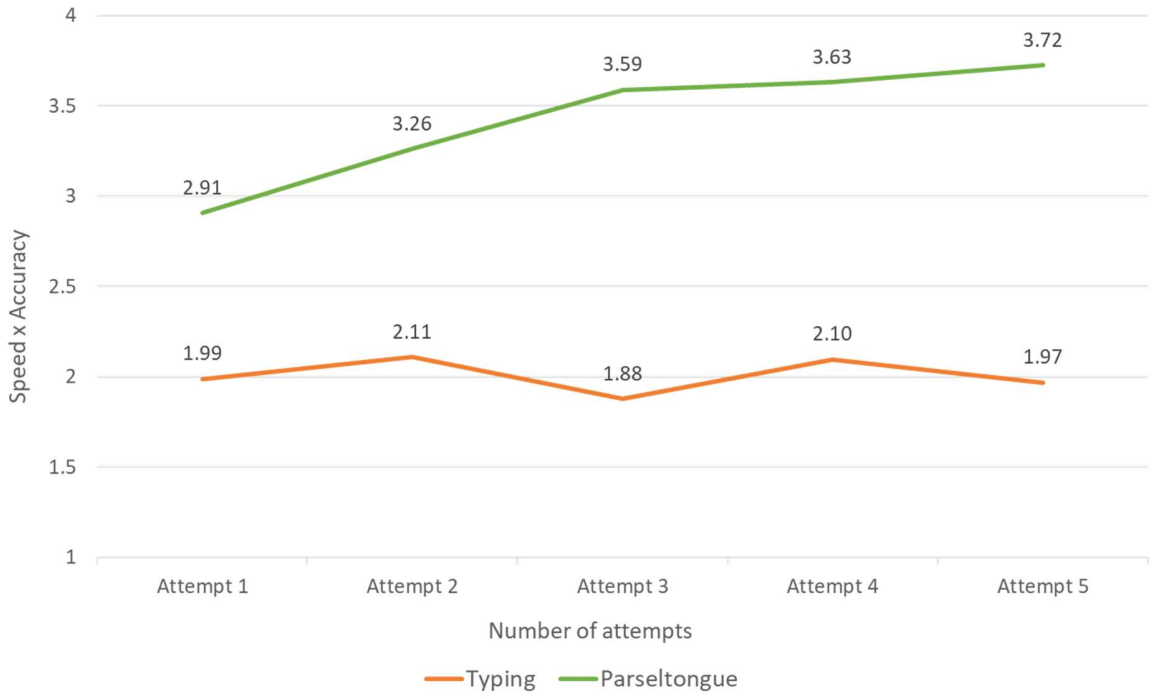
Graph 5.2 – Speed Comparison Graph

5.3.3 Overall Performance

When speed and accuracy is compared, performance analysis becomes complex. It is because if someone tries to type faster, his accuracy will be lower and if someone tries to type more accurately, his speed will be slower. Hence, there should be a good balance of speed and accuracy to achieve maximum performance. This is called as Speed-Accuracy tradeoff.

Speed-Accuracy tradeoff is the complex relationship between an individual's willingness to respond slowly and make relatively fewer errors compared to their willingness to respond quickly and make relatively more.

So, finding out overall performance varies from condition to condition. In this case, we cannot plot speed against accuracy because we will not have any equal intervals in any of the parameters. Hence, we will take number of attempts on x-axis. For y axis, we should take the ratio of speed and accuracy, but since they are inversely proportional, they will be multiplied instead taking the ratio. Following is the overall performance graph:



Graph 5.3 – Overall Performance Graph

CHAPTER 6

CONCLUSION

Voice coding unleashes a whole new perspective to speech input and its uses. We have slowly conquered the arena of Natural Language Processing and it is time to build on the acquired knowledge. We made a small attempt of doing our part in popularizing voice coding by the means of this project.

Being able to code using speech will not only prove beneficial to the programmers who have been in the field for long but also people who want to start with coding

Typing code continuously can be frustrating. So, programmers can use voice coding and shift between typing and voice for coding making it easy for the programmers to deal with hefty work. Voice coding underlies a wide variety of science and any researcher who writes code could use it.

CHAPTER 7

FUTURE SCOPE

Parseltongue is flexible enough to be used in different environments

It can be used:

- By beginners to learn Python or new programming language
- To teach students more on understanding the code rather than remembering the syntax
- By developers to code faster
- In testing by providing commands to make the process faster

These following implementations are milestones we aim to achieve

- Increase the number of commands so that the user can access vast amount of python functions and libraries
- Modify Parseltongue to support multiple programming languages
- To simplify the installation process, therefore making it more accessible

CHAPTER 8

REFERENCES

- Code By Voice - https://github.com/simianhacker/code-by-voice
- Dragonfly - https://code.google.com/archive/p/dragonfly
- Dragonfly 2 - https://github.com/dictation-toolbox/dragonfly
- Hands Free Coding - https://handsfreecoding.org/

- Programming by voice, voice coding paper by Stephen C.Arnold and Leo MarkDragon Systems, Dragon NaturallySpeaking SDK, C++ and SAPI Guide and Reference, Newton, Mass. 1999.
- Web Based Programming Tool with Speech Recognition for Visually Impaired Users Kaveendra Lunuwilage1 Faculty of Computing, Sri Lanka Institute of Information Technology
- Voice Recognition Applications For Programming Environments Adel S. Elmaghraby Engineering Mathematics & Computer Science University of Louisville 1 le, Kentucky