Ok Eclipse+: Enabling Voice Input to augment User Experience in Integrated Development Environment

Srujan Barai
NC State University
sjbarai@ncsu.edu

Daxkumar Amin
NC State University
dkamin@ncsu.edu

Aayushi Agrawal
NC State University
agrawa@ncsu.edu

Shenee Ashara
NC State University
spashara@ncsu.edu

ABSTRACT

Whenever you are working with an Integrated development environment (IDE), it happens a lot that you might forget the shortcuts to commands in menu bar or forget where that command is present on menu bar because of higher level hierarchies of commands present there. Ok Eclipse is intended towards improving user experience by embedding speech recognition as a plugin in Eclipse IDE. This was implemented by the first group for few commands and showed that it can be possible for other commands on menu bar. This report gives a glimpse of what they implemented and how it works. Later we are providing brief details of what improvement we are planning to implement on the base created by the first group. This report also includes survey from fellow classmates which would help us access utility of our idea and also welcome their ideas regarding future scope of this project. We further define the plan of work which we will take in consideration for achieving milestones.

Keywords

Speech to text, Integrated development environment, Q&A repositories, plugins, error solving

1 INTRODUCTION

In the era of abundant voice libraries, it was felt that IDEs are not taking advantage of this facility. While using an IDE, users rely on mouse, keyboard and touch displays as means of giving input to the system. Thus, it was realised that voice can be considered as a potential source input when it comes to IDEs. This would reduce the time consumption as well as reduce the work to be performed by the user. But, according to an evaluation conducted by expert Java developer it was concluded that having only voice as a source of input would in fact decrease the time efficiency.[10] Thus, using voice commands for simple tasks like accessing commands from menu bar would prove efficient. Taking the above facts in consideration 'Ok Eclipse' was made as a plugin for Eclipse IDE which listens to user and accesses the menu for them. Thus, user won't be required to remember the shortcuts or where the command is present on menu bar, ultimately increasing the time efficiency. Also a Q&A section has been implemented which takes message from console logs and queries stackoverflow repositories for appropriate solutions.

In a survey result of 30 developers who use eclipse in their daily life, it was found out that developers would prefer remembering shortcuts over speaking them to do things. Rather than being discouraged by this fact, we decided to take this to another level where 'Ok Eclipse' can do things beyond refractoring and declaring a new variable. The most annoying thing for a developer is fixing errors. Some of these are logical and hence depends on the logic of the code but rest of the errors can be solved without depending on the code. To solve these kind of errors we are planning to introduce a magical keyword 'FIX IT'. This keyword would work in any context, meaning irrespective of what context the error is, the error can be fixed with this magical keyword.

A brief of errors, this keyword - 'FIX IT' plans to fix is, for example, if user is developing Android applications using Eclipse (though, which now is deprecated), there are some APIs that are restricted to only new versions of Android SDK. In this case, there will be some API calls that does not work with older Android versions. In such scenarios, the 'FIX IT' command will modularize code according to the SDK levels to make sure certain code only runs on SDKs that facilitates those APIs.

2 PREVIOUS WORK DETAILS

The first group detected the problem that user has to do a lot of work while working in IDE. By adding voice commands and automatting some functions in IDEs greater efficiency can be achieved. Now the reason to choose Eclipse as an IDE to implement project is because its open source and is formed of various plugins and thus, creating a plugin for speech detection for current version of Eclipse make sense. They addressed and solved following two problems:

1. **Listening Menus:** Eclipse is composed of various plugins for editor, views and menus which are driven by specific set of commands. Navigating views with a mouse is cumbersome especially in scenarios where certain menus require traversing multiple levels down the menu. This is solved by identifying every menu item associated with a command. Now each command has a unique command ID and thus, creating a look-up table which maps them will lead to execution of

that command. Now to convert speech to text CMU Sphinx is used. Here, the user is expected to recite the label text and the engine will invoke the appropriate command using the look-up table. The working of this functionality can be understood from the following figure:

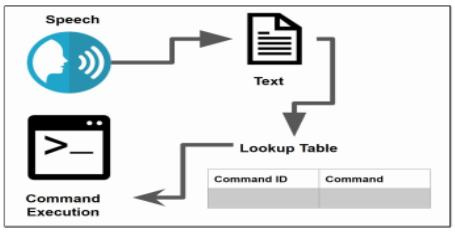

Figure 1: Listening Menus

2. **Loud Console:** In Eclipse, the console portrays exact behavior of code at the given time- Errors, user defined messages and exceptions. The first group has recommended fixes by extracting the relevant text of interest from console log to form a query and mine Q&A (stackoverflow.com) repositories for appropriate solutions. Whenever the user feels like he might need a recommendation, he/she can invoke the 'Find' command to get a suitable recommendation for the relevant console log displayed at a given time. The working of this functionality can be understood from Figure 2 and to incorporate all these features they have considered following use cases which can be followed from Figure 3:

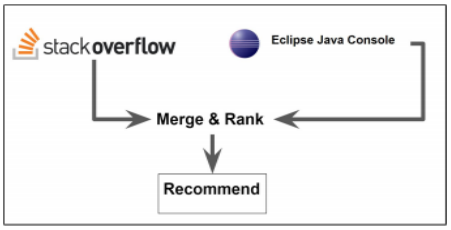

Figure 2: Loud Console

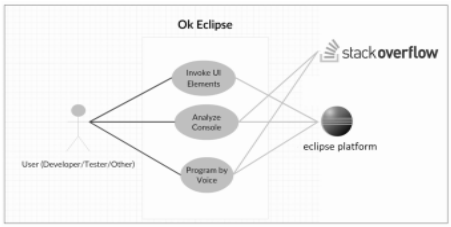

Figure 3: Use Cases

3 USER EVALUATION

1. Which one would you prefer - A complex keyboard shortcut or voice command for a task?

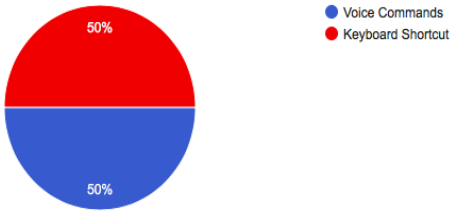

Figure 4: Question 1

We asked this question to gauge how many developers would prefer voice commands for IDE. This gave a rough estimation of how many many people would be interested in using the Ok Eclipse plugin. The survey results show that 50 percent of the survey takers prefer voice commands over complex keyboard shortcuts. In a powerful IDE such as Eclipse there are many functionalities and respective keyboard shortcuts. Remembering a large number of keyboard shortcut can be a cumbersome task. And hence simple voice commands can be very helpful.

2. Would you like Ok Eclipse to recognise a larger set of commands with a little trade off to response time?
This questions was asked to know if users would be interested in additional commands to the already existing version of Ok Eclipse.

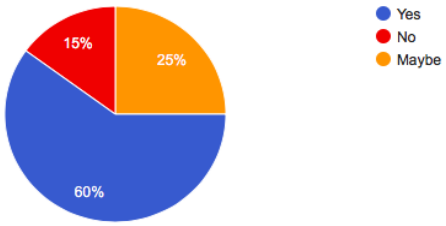

Figure 5: Question 2

There can be a trade-off between the number of commands supported by the system and the response time. Larger set of commands can increase the response time, as it would take more time to map the voice signal to corresponding text in the Spinx library. However, our survey responses indicate that 85 percent of the users want to have more voice commands in the plugin, which would make various tasks for them easy.

3. In comparison to other IDEs how much do you prefer using Eclipse?
This question would make us understand that how often developers use Eclipse. This would also help us understand the usefulness of the application. The survey results show that 70 percent developers use Eclipse very often. Also, Eclipse was ranked most popular IDE in 2017. Total market share of Eclipse is around 23 percent. Hence, Ok Eclipse would be quite helpful to the developers in general.

4. Which IDE do you use the most?
We were interested to know what other IDEs are popular

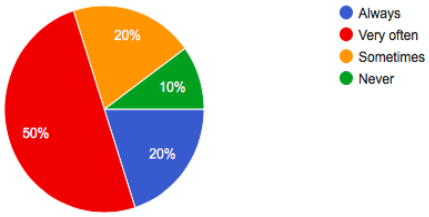

Figure 6: Question 3

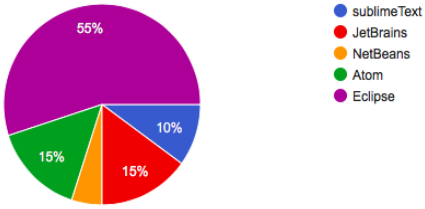

Figure 7: Question 4

among the developers. This would help us to analyze to what other IDEs we can extent this application.The results of the survey show that Eclipse is by far the most popular IDE. We only included the IDEs that are open source. JetBrains and Atom are the next best IDEs.

5. Would you prefer a two way communication with the IDE?
This question was asked to evaluate the need of one of the functionality, we are going to implement. Two way communication here means that when you provide a voice command to the application, it give an appropriate voice response.

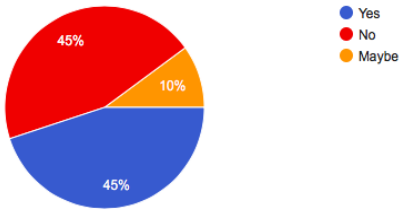

Figure 8: Question 5

Around 55 percent of the survey takers say that they want the plugin with two-way communication.

6. How cool would it be - if you can solve the errors in your code just by saying 'Solve error'
This question was again to evaluate the popularity of one of the features we are going to implement. Eclipse does provide the suggestions, if there are any syntactic of structural error in the code. We are trying to leverage that functionality and provide a voice command over it.More than 85 percent of the users rated this feature 4 and above - on scale of 1-5.

4 Ok Ecipse+ Implementation

To enhance the previously implemented project, we decided to implement automatic solving of basic errors after analyzing many options like creating a two way communication with IDE or creating a plugin through which user can use speech commands to draw UML diagrams in Eclipse. But, these ideas were not possible to implement in the given time constraint. Our idea to implement automatic solving of basic errors uses the first implementation of project as base and implements on top of that which would let us complete the project in given time. We are also looking into implementing NLP if time permits, this is to make sure users do not have to remember certain commands to do get that thing done, instead user can speak in a normal language and the system will detect the appropriate function for the same.

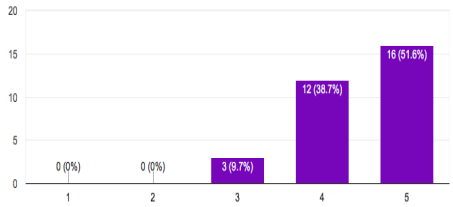

Figure 9: Question 6

For the feature that uses NLP to understand the commands, we are taking the speech vector and giving it to CMU sphinx library which converts the vector to text. Now if this text contains the pre-decided keyword 'FIX IT', it will look for a function that returns the possible solution provided by Eclipse for the current error displayed. When the user gives command and if it isn't the pre-decided keyword, we plan to use Stanford's NLP library. We have also considered a paper NLP for NLP (Natural Language Processing for Natural Language Programming) which can be useful. We are still evaluating it for its implementation in the project.

In the above case, if there is only one solution to the error, the command for that solution is executed.Alternatively we have also considered a scenario where more than one solution is possible in which case, all these solutions are presented to the user by editor and the user will be given choice to choose whatever they finds useful. Once the user selects the solution, command for that is executed.

4.1 FLOW DIAGRAM

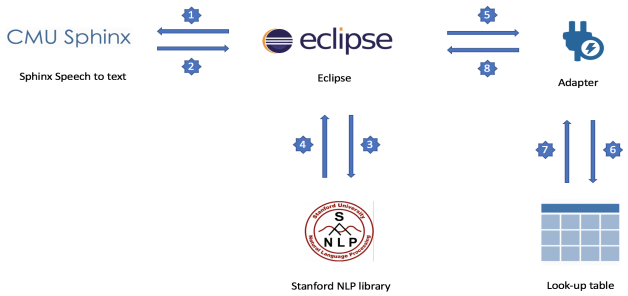

Figure 11: Flow Diagram

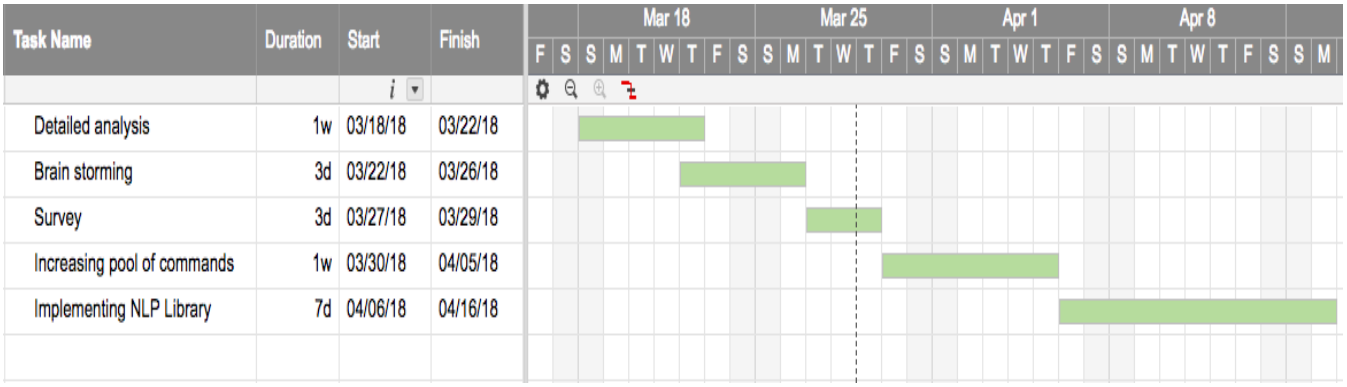

Figure 10: Gantt Chart

1. When user speaks a command, the speech vector will be sent to CMU Sphinx library that converts the speech to text. This text will be used to recognize what user is attempting to do. The Sphinx library uses hidden Markov acoustic models (HMMs) and an n-gram statistical language model which assures great quality conversion with high accuracy.

2. The CMU Sphinx library responds to the editor with a String object which contains the statement spoken by the user. There is no limit to the words CMU Sphinx can convert but the user is not expected to take breaks between words in a statement.

3. The eclipse sends the String to Stanford NLP library for analysing the content to know what users wish to do from the set of commands that 'OkEclipse' can actually do.

4. Eclipse will receive a JSON object from the Stanford's nlp algorithm which will contain segregated keywords based on types of speech and sentimental analysis. These words can be used to identify users actions. The reason for using the JSON object from Stanford NLP library over direct conversion from CMU Sphinx is that even if user mess up with the command little bit, the language processor should understand what the user is trying to do.

5. Eclipse will looks for the keywords that makes sense to the system. These keywords will then be sent to the adapter whose work is to link the keywords to appropriate function depending on the functions that already exist in Eclipse.

6. The adapter will search in the lookup table for an appropriate function that matches the keywords requested by the user. The adapter is expected to find the best matching function from the table, user can set a custom threshold where a function will be chosen only if it matches the string by a certain extent.

7. The result of the lookup table should be only one specific function, the adapter will keep on looking up unless it find a good match or returns a null value if it cannot find any matching functions. In case the adapter find multiple functions that are matching the criteria it should show a popup asking user which function they intend to use. User can select the function from the popup which adapter returns further.

8. Once the adapter find an appropriate functions, it will return it to the main code that requested the function search. On arrival it will execute the function and user can see the results.

5 PLAN OF WORK

The Gantt chart in Figure 10 shows the anticipated schedule of our project. The whole project is divided into major tasks and each task is given a duration. This chart will be used by the teammates to look at regularly. This will help us divide the major tasks into smaller sub-tasks and keep the track of timeline.

- In the first week we plan to do the detailed analysis of the already implemented project, which would include tasks such as understanding the technology stack and reviewing the implemented code.
- Next step would be brainstorming to think of new ideas which would augment the currently implemented application.
- We will also evaluate the user-needs and feasibility of our ideas. For this we planned to roll out surveys to around 30 developers.
- We would then increase the pool of commands in the existing application. We have got a good response for the same from the stakeholders.
- Lastly, we want to implement Stanford NLP library in the plugin, that would help with context recognition.

6 CONCLUSION

Currently Ok Eclipse is able to identify commands using voice recognition. However it does not help in solving runtime/compile time errors. Hence, we have proposed a method for error handling in Ok Eclipse using voice recognition as one of the solution to improve the user experience. In this report we have defined the problem and our motivation. Manageable requirements have also been stated along with

a suitable plan of work that we presume will be the most effective way to evaluate the intended goal of our project.

References

[1] Shaik, S., Corvin, R., Sudarsan, R., Javed, F., Ijaz, Q., Roychoudhury, S., Bryant, B. (2003, October). SpeechClipse: an Eclipse speech plug-in. In Proceedings of the 2003 OOPSLA workshop on eclipse technology eXchange(pp. 84-88). ACM.

[2] Begel, A.,& Graham, S. L. (2006, September). An assessment of a speech-based programming environment. In Visual Languages and Human-Centric Computing, 2006. VL HCC 2006. IEEE Symposium on (pp. 116-120). IEEE.

[3] Diaz, Jaime, and Raiza Muniz. "Voice Recognition System." (2007).

[4] Bell, Donald. "UML basics: An introduction to the Unified Modeling Language." The Rational Edge (2003).

[5] Meyer, Stephenie. Eclipse. Little, Brown Books for Young Readers, 2007.

[6] Writing an Evaluation Plan. https://www.brown.edu/research/conducting-research-brown/preparing-proposal/proposal-development-services/writing-evaluation- plan.

[7] B.Boehm. Software Productivity and Reuse. IEEE Computer Society. Sept. 1999.

[8] University of Calgary Kent Beck et al. JUnit - Unit testing framework. http://junit.org. [Online; accessed 1/30/2018]. 2018.

[9] Eclipse Foundation. Eclipse Next-Generation IDE. https: //www.eclipse.org/che/. [Online; accessed 1/30/2018]. 2018.

[10] S.L. Graham A.Begel. An assessment of a Speech-BAsed Programming Environment. Visual Languages and HumanCentric Computing, 2006, VL/HCC, IEEE Symposium.