The Implementation of a Vocabulary and Grammar for an Open-Source Speech-Recognition Programming Platform

Jean K. Rodríguez-Cartagena
UPR Río Piedras
San Juan, PR
jeank02@hotmail.com

Andrea Claudio-Palacios
University High School
San Juan, PR
apclaudio1211@gmail.com

Natalia Pacheco-Tallaj
University High School
San Juan, PR
nataliamariapt@gmail.com

Valerie Santiago-González
UPR Río Piedras
San Juan, PR
vsg126@gmail.com

Patricia Ordonez-Franco, PhD
UPR Río Piedras
San Juan, PR
patricia.ordonez@upr.edu

ABSTRACT
Speech recognition software technology provides people with limited hand mobility or visual impairments the opportunity to work with computers through an alternative approach. However, when it comes to programming, voice recognition systems leave much to be desired. To tackle this problem, we have developed a generic vocabulary and grammar to help a user to program by voice in any C-based language. With the purpose of validating the aforementioned vocabulary and grammar, receiving feedback from the programming community and determining future features for enhancement, we administered a survey and distributed it among faculty members and advanced computer science and engineering students. Our objective is to create an assistive technology tool that will serve the community with limited hand mobility or visual impairments yet will be accepted and embraced as the spoken programming language and model of choice for implementation.

Categories and Subject Descriptors
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces| *Input devices and strategies, Voice I/O*; K.4.2 [Computers and society]: Social Issues| assistive technologies for persons with disabilities

Keywords
Programming; voice recognition; Simon; open source

1. INTRODUCTION
Every programming language contains its own vocabulary and grammar to function. To write code, programmers need to follow guidelines, also known as syntax rules. Nonetheless, every programmer has a personal way of referring to code when saying it out loud. This variety creates ambiguity for speech-recognition software (SRS) as a result of its lack of structure. Since coding requires intensive typing and therefore the use of repetitive finger movements, many programmers suffer from carpal tunnel syndrome (a condition that causes numbness and tingling in the fingers and hands) or other repetitive motion disorders. With the purpose of creating clearness and an unambiguous vocabulary for an accurate dictation, we proposed the use of gerunds as keywords to differentiate voice commands. It was determined that gerunds provide certainty to the spoken programming platform because they are more phonetically distinguishable by speech-recognition software.

ASSETS'15, October 26–28, 2015, Lisbon, Portugal.
ACM 978-1-4503-3400-6/15/10.
DOI: http://dx.doi.org/10.1145/2700648.2811346

2. RELATED WORK
Voice-recognition technology is used by professionals for the purpose of improving personal efficiency levels; others use it for enhancing the lives of people with disabilities [2]. For the former, programmers have been developing methods to write code using voice commands instead of the keyboard typing. For example, in Pycon 2013, Tavis Rudd, a professional programmer, explained how he writes code by dictating instead of using the keyboard. He used Dragon Naturally Speaking and a Python speech extension, and stated that it was more efficient than with the traditional way of typing on the keyboard [5]. His video, "Using Python to Code by Voice" emphasized programming through a speech-driven platform is a "highly effective tool that could benefit all programmers" [5]. He implemented a grammar which used short sounds and mundane words as the input, but this grammar would represent struggle for people who are not programmers and need or want to learn to code initially by voice. Nevertheless, it precipitates the development of a vocabulary and grammar which is feasible for users to learn through a universal design.

With this objective, a Java program was created to develop a verbal system to use while programming, where Java developers should use Spoken Java to create programs. Spoken Java is a language system designed to speak Java naturally, based on Andrew Begel's Research Paper "An Assessment of a Speech-Based Programming Environment". As a result, it was concluded that the programmers had little trouble learning Spoken Java [4].

For these contributions to be available to the public and make programming universally accessible, we selected an open-source, voice-recognition software named Simon for this platform [6].

3. METHODOLOGY

After analyzing a Perl script that contained all the elements of the desired generic C-based programming language, the authors developed a vocabulary and grammar for a generic spoken programming language. This language was refined through the gathering of the feedback obtained from the programming community through a detailed survey that was developed. The survey's questions portrayed an array of commands on the programming control flow structures. (See Table 1)

The programming community consisted of random senior students and faculty members of the Computer Science and Computer Engineering Departments at the Polytechnic University of Puerto Rico and the University of Puerto Rico Río Piedras. Thirty-three (33) responses were received during the months of November and December of 2014.

Each question contained the purpose of the question, describing the **intended C++ action in black and bold**, the **proposed voice command in bold, Arial, and red**, the ***corresponding C++ code***, as a consequence of the voice command, ***in italics, bold, and black*** (this is the output produced by the voice-recognition software in an editor), and ***variables and literals presented in bold, italics, and green***. The survey contained ten (10) questions in all.

4. RESULTS

The final stage of the project consisted of performing a quantitative and qualitative analysis of the results obtained from the survey to determine the final vocabulary and grammar below.

Table 1: Selected elements of the generic open-source, spoken programming language of the Kavita Project [7]

Programming Structure	Voice Commands
First statement: defining a variable	**defining integer variable** *int ;* **total** *int total;*
Second statement: printing a string	**printing literal** *cout << " ";* **hello world** *cout << "hello world ";*
Third statement: printing a variable	**printing variable** *cout << ;* **total** *cout << total;*

The remaining programming structures not listed in Table 1 were 4) printing the result of an equation, 5) storing user input, 6) creating an expression using variables and modulus operator, 7) creating a relational expression, 8) creating an expression using logical operators, 9) creating a for loop structure, and 10) creating an if /else structure. Due to page limitations, more of examples could not be included in the abstract, but may be found at the project website [7].

5. FUTURE WORK

This work is the beginning of the creation of a generic spoken programming language for C-based programming languages. Currently, we have a working implementation of the language in Java. The next step is to evaluate the project with its intended audience, users with limited mobility in their hands. Afterwards we will begin to incorporate more C-based languages into the platform.

6. CONCLUSION

The final proposed vocabulary and grammar represents the first phase of a larger initiative, the Kavita Project [7], whose aim is to develop an open-source, generic, spoken programming language platform whose interface is so natural it will be attractive to all programmers. However, our primary intention is to make programming accessible to more people.

7. ACKNOWLEDGMENTS

The researchers would like to convey their gratitude to Dr. Blanca Tallaj for helping to distribute the survey in the Polytechnic University of Puerto Rico and to Xiomara Figueroa Fontánez, Rafael Esparra, Gustavo Gratacós, José Montero Felix, and Jessica Pagán for their role in the development of this project and their assistance in the process.

8. REFERENCES

[1] Pacheco-Tallaj, Natalia M., and Claudio-Palacios, Andrea P. "Development of a Vocabulary and Grammar for an Open-Source Speech-driven Programming Platform to Assist People with Limited Hand Mobility". Research report submitted to Keyla Soto, UHS Science Professor.

[2] Stodden, Robert A., and Kelly D. Roberts. "The Use Of Voice Recognition Software As A Compensatory Strategy For Postsecondary Education Students Receiving Services Under The Category Of Learning Disabled." Journal Of Vocational Rehabilitation 22.1 (2005): 49-64. Academic Search Complete. Web. 1 Mar. 2015.

[3] Begel, A., Graham, S.L. "An Assessment of a Speech-Based Programming Environment," Visual Languages and Human-Centric Computing, 2006. VL/HCC 2006, pp.116,120, 4-8 Sept. 2006.

[4] Rudd, T "Using Python to Code by Voice". Pycon US 2013. Retrieved April 25, 2015 from http://pyvideo.org/video/1735/using-python-to-code-by-voice

[5] Figueroa, X., Ordóñez, P. "Improving Programming Interfaces for People with Limited Mobility Using Voice Recognition", Assets 2014.

[6] Simon Listens. http://www.simon-listens.org/ Last accessed: 2015-08-24.

[7] The Kavita Project – 2014. http://www.thekavitaproject.org/. Last accessed: 2015-08-24.