Combining Visual and Textual Languages for Dyslexia

Luis F. González
Universidad Politécnica de Madrid
Madrid, Spain
Politécnico Colombiano Jaime Isaza Cadavid
Medellín, Colombia
lfgonzaleza@elpoli.edu.co

Abstract

This paper describes the research of a Software, Systems and Computing PhD thesis conducted at the Universidad Politécnica de Madrid.

The aim of this research is threefold: i) design a model to promote interaction between programmers with and without dyslexia during software development as part of a team, ii) overcome the obstacles facing dyslexic programmers when they are writing a program, and iii) increase software development performance and efficiency levels when one of the programmers has symptoms of dyslexia.

1 Motivation

Visual programming languages, although they have limitations, have become powerful tools for you perform tasks that are more complex with textual programming. Some activities such as modeling or teaching processes in children are easier to perform using visual languages. Communities such as Scratch [1] have successfully demonstrated that anyone who does not have much programming knowledge, is able to create complex programs with little training, learning im-portant problem-solving strategies, designing projects and communicating ideas [2].

The World Health Organization states that the majority of people live with a certain degree of "functional diversity" (physical or cognitive limitation) and dyslexia has the highest number of patients [3].

Some symptoms of dyslexia, are manifested in reading and writing comprehension problems. People who suffer from it often mistake certain letters when reading or writing. That is why these problems, specifically in computer programmers, have become a barrier that limits the efficiency in software development processes.

Statistical data have been reported that 12.4% of a group of computer programmers believe they are dyslexics [4] and although this is above the global prevalence rates established [5] (who state that 5.3% to 11.8% of a population is dyslexic), the dyslexia has not been much studied in people who develop software.

When programmers with dyslexia develop software in a traditional development interface, they usually face several problems, for example: It is difficult for them to remember details of the code, they do not manage a good order in the presentation of this one, they have difficulty to define the variables correctly (specifically combining letters in the names of these and follow certain patterns of the defined standards) and they experience slowness in the coding of the program and search for errors; however, all these problems can be minimized when programmers with dyslexia work with programming tools with visual or graphical aids, making them more successful in software development and monitoring [6]. It has been proven that programmers with dyslexia are 37% more effective in software development , when they use a visual programming language [4] and the use of these, is reflected in more levels of efficiency and quality of software developed.

Currently in to software development is taking force the work trend known as "Pair Programming", which requires two people working together for a common purpose. According to the previous statistical data, it is possible to find that one of these programmers presents symptoms of dyslexia, (factor that can decrease or hamper the expected results of

the group). Therefore, we propose to design a model that, through bidirectional transformation between codes of visual and textual programming languages, it allows the development of software in team and helps the interaction of programmers with and without dyslexia, where each them might select their preferred development environment (visual or textual) and he will see in their own environment what the partner is making. This would not only improve the levels of personal motivation of programmers with dyslexia, but it might be a great contribution in their area, achieving a positive impact in the levels of software development, which would be reflected in the increase of efficiency and decrease of the development times, that are presented in a group of programmers with these characteristics.

2 Problem

Adult dyslexia has been a hidden learning difficulty, according to studies [7], only 0.95% of respondents that stated that they believed they were dyslexic had been diagnosed by a professional. Hence, we can infer that this disorder is often ignored, incorrectly managed and not treated in a timely fashion. As a result, it is not a functional diversity to which adult education centers attach importance, whereas it has a major impact on the personal development of sufferers. Evidence to this effect is that higher education institutions have negligible information about students with dyslexia, and adults with dyslexia have not had full access to research advances in the past.

Just as, the effects of dyslexia can be a barrier to proper job performance for software developers. Dyslexic programmers face several learning, communication and interaction problems, and they have trouble with activities associated with code generation and correctness when developing software in a traditional development environment; in brief, computer programmer's with dyslexia may have difficulties with spelling, syntax rules, processing sequenced symbolic information and short-term memory, all of which are important for programming [8]. Some of the most common potential difficulties in dyslexic programmers are:

- Trouble remembering code details.
- Poor handling of code ordering.
- Issues with correctly defining variables
- Slowness in coding the program and searching for errors.

Now them, these difficulties establish a highlighted difference in performance between programmers with and without dyslexia, becoming an efficiency problem in software development teams, issues that can increase production times and reduce software quality. However, these problems can be considerably minimized if there is the possibility of being able to develop computer software in such a way that each programmer can make use of a textual or visual programming environment according to their preferences.

For the above reasons, it confirms the need to define with clarity: What is the model that allows the bidirectional transformation of codes of languages of visual and textual programming, in order to mitigate the differences of performance in the interactive work of programmers with and without dyslexia, when developing software in team?

3 Aproach

As a step forward in problem solving, we have the following activities have been established.

3.1 Activities Performed or in Progress

Mapping of needs, skills and characteristics of people with dyslexia. (Performed activity). The main characteristics and difficulties of adults with dyslexia have been identified. The most relevant characteristics helped to define the profile of programmers with dyslexia.

Classifying, typing and characterizing visual programming languages. (Performed activity). We defined a visual programming languages taxonomy, which helped to select the visual language ideal one, to make different performance experiments of the programmers with and without dyslexia; moreover, we did a classification and analysis of 31 visual programming languages, identified as the best known and used in the medium, in order to select the language to be included in the code transformation model. After studying variables such as: purpose, relevance, validity, technical characteristics and language facilitating characteristics for the accessibility of programmers with dyslexia, we concluded that the Alice programming language, it would appear a priori to be the language that programmers with dyslexia are most likely to be confident working; instead of preference a text-based programming language. Taking into account that Alice's, diagrams are quite complex and it has visual characteristics that may be assist to programmers with dyslexia, these can minimize syntax errors, which are considered to be one of the biggest problems that programmers with dyslexia have when developing software. Alice programming language, is recommended as the ideal to be included in the code transformation model, that will help mitigate the effects of dyslexia on programmers who suffer from this functional diversity.

Classifying, typing and characterizing textual programming languages. (Performed). After classifying and analyzing 27 textual programming languages identified as the most used by 351 programmers, it was determined that Java would be the ideal language to be included in the model design. For the analysis above, there were variables such: purpose, relevance, validity and frequency of use were taken into account. In the group of programmers with dyslexia surveyed, the frequency of use of Java was 92%, followed by 64% for C ++ and 59% for JavaScript.

Characterizing the elements that interact in a visual programming language. This characterization is required

to be able to determine the potential relations among the different elements of the selected visual language. At the moment a characterization is being made, that will help to determine the possible relations between the diverse elements of the selected visual programming language and after that, formalizing those relations and to establish levels of comparison and equivalences, with the relations that are obtained of the elements of the textual programming language.

3.2 Pending Activities

Modeling the relation between the visual language elements. The aim is to help formalize the relations specified in the previous task, and thus, be able to establish similarities and equivalences with the textual programming model.

Characterizing the elements that interact in a textual programming language. This is required to be able to determine the potential relations, between the different elements of the selected textual language.

Modeling the relation between elements of the textual language. The purpose here it is helping to formalize the relations specified in the previous task, and so on, be able to establish similarities and equivalences with the visual programming model.

Summarizing transformation rules from visual to textual code and vice versa. The defined rules of transformation will be used as input for the design of the algorithms that will be proposed.

Algorithms for transforming visual to textual code and vice versa. They will be designed by applying the defined rules and they will be used to establish the proposed model.

4 Evaluation Methodology

4.1 Hypothesis

The proposed model allows the development of software under a single interface, which can interact between visual and textual codes, helping dyslexic programmers in the process of interaction with their co-worker. This happens when two programmers with different preferences related to the programming environment, work as a team and each of them can select their favorite programming environment (whether visual or textual) Independent of the model selected, each one can see what of their partner is making in the other environment.

4.2 Experimental Setup

To obtain the map of needs, skills and characteristics of programmers with dyslexia, a quantitative cross-sectional study, was carried out using a convenience sample of 351 computer programmers.

To determine the efficiency of the proposed model, we will carry out a pilot tests, based on a comparative and analytical study in the use of visual and textual programming languages, between programmers with and without dyslexia. For that study there will be posing several problems that would solve in both languages.

The developed programs will be analyzed and evaluated, according to the following aspects:

According to the programmer's experience:

- Success Rate: Time of development, percentage of programmers who completed the program, precision and functionality.
- Level of satisfaction: Typical usability surveys, to evaluate the experience using the new work environment.

According to the quality attributes of the program, that will permit us to know:

- Number of errors.
- Level of efficiency and complexity.
- Code analysis
- Rate of maintainability
- Complexity of the resulting code
- Rate of sustainability .

At last, another experiment will be conducted using a prototype which will implement the designed model. In the same way, it will be measuring the variables defined in the last experiment, besides, to evaluate the level of acceptation and interaction provide by the tool, with respect to team work.

The obtained results will be subjected to several statistical tests, to determine whether each of these variables have a statistically significant correlation with respect to the problem solution.

References

[1] MIT Scratch Team. About Scratch. Accessed 01/23/2017, https://scratch.mit.edu/about.

[2] Resnick, M. Aprender a Programar para Aprender. Accessed 07/04/2017, https://scratch.mit.edu/about

[3] World Health Organization. 2011 World report on disability.

[4] Fuertes, J., González, L. and Martínez, L. 2016. Characterization of Programmers with Dyslexia. In Computers Helping People with Special Needs. LNCS, Linz, Austria.

[5] Katusic, S. , Colligan, R. C., Barbaresi, W. J., Schaid, D. J. and Jacobsen, S. J. 2001. Incidence of reading disability in a population based birth cohort. Rochester, Clinic Proceeceedings, 1081.

[6] Dixon, M. 2007. Comparative Study of Disabled vs. Non - disabled Evaluators in User - Testing: Dyslexia and First Year Students Learning Computer Programming. In Stephanidis, C.ed. Springer Berlin Heidelberg, Pp. 647-656.

[7] Fuertes, J., González, L. and Martínez, L.2016. Características de los programadores disléxicos. Actas de Ingeniería.

[8] Powell, N., Moore, D., Gray, J., Finlay, J. and Reaney, J. 2004. Dyslexia and learning computer programming. 36,3.