Usability Perception of Visual Programming Language: a Case Study

Jenny Morales[1], Cristian Rusu[2]

[1] Facultad de Ingeniería, Universidad Autónoma de Chile, Av. 5 Poniente No. 1670, Talca, Chile

jmoralesb@uautonoma.cl

[2] Pontificia Universidad Católica de Valparaíso, Av. Brasil No. 2241, Valparaíso, Chile

cristian.rusu@pucv.cl

Abstract. Block programming languages are considered as facilitators of programming learning. Programming languages like Scratch are used by adolescents and children from different countries. Usability is an important aspect to facilitate interaction with interfaces, in this case, the interaction and use of the programming language. This work presents a usability perception study of Scratch, based on the System Usability Scale (SUS). The survey involved 96 teenagers and SUS was applied after the experience of using Scratch in the laboratory. The results indicate that Scratch's overall usability is a bit under satisfactory. They also highlight elements to improve.

Keywords: Usability, System Usability Scale, Scratch, Survey

1 Introduction

Today there are a large number of programming languages and programming environments for end users. They are successful because they promise to take away the complexities of language syntax, and focus on developing skills like programming logic. They are also aimed at various groups of people, including children, programming novices, and non-programmer adults. Considering that these programming languages facilitate the learning of programming, questions are raised about the usability they have, being the subject of various usability studies. Koitz and Slany carried out comparative studies between two visual programming languages for teenagers, based on user tests and questionnaires [1].

In this paper, we present a usability perception study of Scratch through a survey based on the System Usability Scale (SUS), applied to adolescent students from two Chilean schools. The results obtained based on 96 answers indicated that Scratch reaches a score of 66.3, which places it on the average level of usability, with aspects to improve.

This work is organized as follows: section 2 describes the background, section 3 presents the methodology we used in data collection, section 4 explains the results. Finally, section 5 shows the conclusions and future work.

2 Background

To learn programming is often complex and discouraging for students. One way of improving students' perception on the difficulty of programming is based on visual block languages and/or through tangible interfaces. Usability of these artifacts is relevant, as it can make programming more appealing.

2.1 Usability

System interfaces allow the users to interact with the systems. One way to establish if the system is easy to use is through usability attributes. Nielsen [2] established five usability attributes that are:

- Learning, related to the ease of learning to use a system, this attribute is relevant for users who are using a system for the first time.
- Efficiency, an attribute that is related to the speed, number of steps or processes with which the user can achieve her/his goals.
- Memorability, an attribute that is associated with the ability of the system to be remembered by users, an important attribute for sporadic users.
- Errors, a good indicator in this attribute is related to a low error rate made by users when performing their tasks.
- Subjective satisfaction, which is related to user satisfaction in relation to the use of the system.

In addition to the Nielsen attributes mentioned above, the International Standard Organization (ISO) 9241-11 defines usability as the "extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" [3]. It considers the effectiveness, efficiency and satisfaction as relevant usability aspects.

2.2 Languages and Programming Environments for Non-Programmers

There are various attempts to facilitate programming for non-professional programmers (end users), in different age groups: children, teenagers or adults. Cao et al. address the difficulties that end users have when carrying out programming tasks, as well as suggestions on how to help them [4].

Several usability studies focus on programming environments. Holwerda and Hermans perform an analysis through cognitive dimensions [5]. Some studies are comparing the usability of visual and textual programming languages related to [1], [6]. They conclude that both approaches can have positive and limiting aspects, and they also consider that those who are hybrids can enhance positive elements of both above mentioned kind of programming languages.

Chawla et al. are focusing on tangible interfaces [7]. They propose a programmable robot that may facilitate programming learning for children.

Zhang et al. focus their analysis on the limitations of block visual language, establishing Scratch-based studies and the difficulties of mixing codes and tracking the flow of data and control within the environment [8]. The difficulty of handling large projects, or when projects become more complex has also been raised [9]. These studies expose the difficulties of Scratch; however, these problems are not only typical for this particular language. We may consider that this more visual and blocky form of programming encourages the design of languages and programming environments closer to natural language, making programming easier for beginners [10].

2.3 Scratch

Scratch is a programming language created in a project of Lifelong Kindergarten Group the Media Laboratory of the Massachusetts Institute of Technology (MIT). The first version emerged in 2013. Currently Scratch is used in more than 150 countries, and it is available in more than 40 languages.

Scratch is used by people of all ages. However, it is addressed mainly for children and young people. Scratch is free for use, and its focus is towards the development of thinking skills and problem-solving, thus enhancing digital literacy [11].

In Chile, as well as in other countries, it has also been promoted to learn programming through the implementation of the Hour of Code [12].

3 Methodology

System Usability Scale (SUS) was created by John Brooke in 1986 [13]. It is used to evaluate the users' perception on different kinds of interactive software systems. The scale has 10 items. The responses to each item are based on a Likert scale in a range from 1, which means "strongly disagree", to 5 which means "strongly agree". The maximum score is 100, and the average established is 68 as a minimum for considering a system as usable. The total score is obtained considering based on a specific procedure. In the case of odd items the score, is calculated subtracting 1 from the obtained value in each item; in the case of the even items, the score is calculate subtracting a 5 the value obtained in the item. Finally, the sum of these 10 scores is multiplied by 2.5, thus obtaining the overall usability score. We adapted SUS to be applied in the context of Scratch. The items of the adapted SUS that we used are shown in Table 1.

We carried out a total of four practical activities in computer laboratories, guided by two programmers in Scratch, each session with a duration of 1 hour. At the beginning of the session, the programming language and the environment to be used, in this case a web site (scratch.mit.edu), were presented. Subsequently, the activities were described and carried out with the help and supervision of the programmers. The maximum quota of participants in each session was 25 students. At the end of the session, SUS was applied. When necessary, assistance was provided regarding the meaning of some items. At the beginning of the questionnaire some general (demographic) questions were included, such as age and gender.

Table 1. System Usability Scale applied in this work.

Nº	Item
1	I would like to use Scratch frequently
2	I found Scratch unnecessarily complex
3	I think Scratch was easy to use
4	I think that I would need the support of a technical person to be able to use Scratch
5	I think that the various functions in Scratch are well integrated
6	I think there is too much inconsistency in Scratch
7	I would imagine that most people would learn to use Scratch very quickly
8	I found Scratch very uncomfortable to use
9	I felt very confident using Scratch
10	I needed to learn a lot of things before I could get going with Scratch

4 Results

The survey included 96 respondents, belonging to two different Chilean schools, each of which was represented by 48 students, that is 50% each. The mean age was 15.6 years, the youngest age recorded was 10 years and the oldest age recorded was 20 years. The distribution by gender is shown in Fig. 1. There were 65 males (68%), 26 females (26%); 6 respondents (6%) did not identified a binary gender.

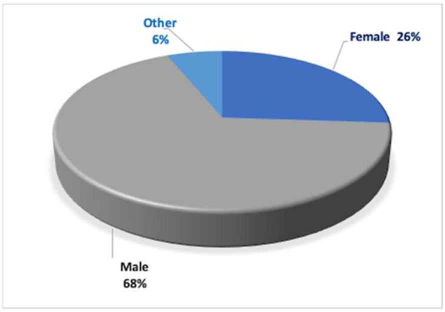

Fig. 1. Gender distribution in both groups.

The score obtained by Scratch in SUS was 66.3. This places it a bit under the average SUS score (68), according to the SUS interpretation scale of [14]. The standard deviation obtained was 14.8, which means that there is a high dispersion of the data obtained. The lowest average score obtained was 15 and the highest was 95 points. 51 responses scored above the 66.3 average, and 45 responses scored below the average. Students' responses (scores) are shown in Fig. 2.

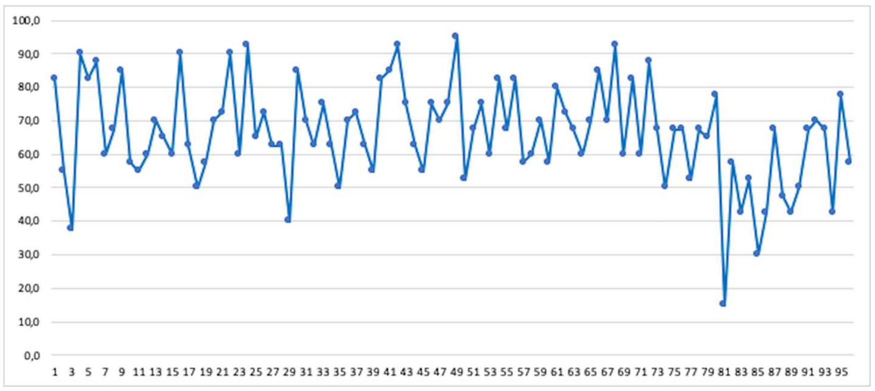

Fig. 2. Responses of all students.

The average scores was calculated as indicated in the methodology section. The lowest scores were obtained for items 2 (I2), 4 (I4) and 10 (I10). These correspond to the usable factor (I2) and the learnable factor (I4 and I10) [15]. Item 5 (I5) obtained the highest score. The integration of language was perceived favorably by the participants.

The result indicates that Scratch was perceived by the students as complex and difficult to learn. On the other hand, the students valued positively the good integration of the functions that Scratch possesses.

5 Conclusion and Future Work

Programming is a difficult task to perform; an attempt to simplify its learning and to reach different users are block and/or visual programming languages. Scratch is one of them, worldwide known and used by many people of various ages.

In this work, an evaluation of the usability perception of Scratch was carried out, based on the SUS. The result obtained locates it a bit under the average of 68, with a score of 66.3. We found that both items related to learnability obtained low results, so we concluded that the students presented difficulties to learn the language. At the same time, only one item related to the usable factor obtained to a low score. Although the score obtained is a bit less than average, we generally consider that the result is positive. In previous works [16], the SUS results for two widely used integrated development environments in higher education in Chile, obtained much lower values: 50.3 for Dev-C++, and 59.6 for NetBeans.

It is generally agreed that in complex projects the work with block languages becomes more difficult. As future work we would like to consider the element of projects' complexity in the experiments with users, and to evaluate a block language other than Scratch.

Acknowledgments. We would like to thank to all the participants in the experiments.

References

1. Koitz, R., Slany, W.: Empirical comparison of visual to hybrid formula manipulation in educational programming languages for teenagers. In: Proceedings of the 5th Workshop on Evaluation and Usability of Programming Languages and Tools, (2014, October) 21-30
2. Nielsen, J.: Usability Engineering. AP Professional (1993)
3. ISO 9241-210: Ergonomics of human-system interaction- Part 11: Usability: Definitions and concepts, International Organization for Standardization, Geneva (2018)
4. Cao, J., Fleming, S. D., Burnett, M.: An exploration of design opportunities for "gardening" end-user programmers' ideas. In: 2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC). IEEE, (2011, September) 35-42
5. Holwerda, R., Hermans, F.: A usability analysis of blocks-based programming editors using cognitive dimensions. In: 2018 IEEE symposium on visual languages and human-centric computing (VL/HCC). IEEE, (2018, October) 217-225
6. Booth, T., Stumpf, S.: End-user experiences of visual and textual programming environments for Arduino. In: International symposium on end user development. Springer, Berlin, Heidelberg (2013, June) 25-39
7. Chawla, K., Chiou, M., Sandes, A., Blikstein, P. : Dr. Wagon: a'stretchable'toolkit for tangible computer programming. In: Proceedings of the 12th international conference on interaction design and children, (2013, June) 561-564
8. Zhang, Y., Surisetty, S., Scaffidi, C.: Assisting comprehension of animation programs through interactive code visualization. Journal of Visual Languages & Computing, Vol. 24 (5), (2013) 313-326
9. Tanrikulu, E., Schaefer, B. C.: The users who touched the ceiling of scratch. Procedia-Social and Behavioral Sciences, 28, (2011)764-769
10. Good, J., Howland, K.: Natural language and programming: Designing effective environments for novices. In: 2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), IEEE, (2015, October) 225-233
11. Scratch.: https://scratch.mit.edu/. Accessed July 26, 2020
12. Hora del Código Chile.: http://www.horadelcodigo.cl/. Accessed July 25, 2020
13. Brooke, J.: SUS-A quick and dirty usability scale. Usability Eval. Ind. 189(194), (1996)
14. Lewis, J. R., Sauro, J.: Item benchmarks for the system usability scale. Journal of Usability Studies, Vol. 13(3), (2018) 158-167
15. Lewis, J.R., Sauro, J.: The Factor Structure of the System Usability Scale. In: Kurosu M. (eds) Human Centered Design. HCD 2009. Lecture Notes in Computer Science, Vol. 5619. Springer, Berlin, Heidelberg (2009)
16. Morales J., Botella F., Rusu C. Quiñones D.: How "Friendly" Integrated Development Environments Are?. In: Meiselwitz G. (eds) Social Computing and Social Media. Design, Human Behavior and Analytics. HCII 2019. Lecture Notes in Computer Science, Vol. 11578. Springer, Cham. https://doi.org/10.1007/978-3-030-21902-4_7 (2019)