







How “Friendly” Integrated Development Environments Are?

Jenny Morales^{1,3}, Federico Botella², Cristian Rusu³,
and Daniela Quiñones³

¹ Facultad de Ingeniería, Universidad Autónoma de Chile, Temuco, Chile
jmoralesb@uautonoma.cl

² Universidad Miguel Hernández de Elche, Elche, Spain
federico@umh.es

³ Pontificia Universidad Católica de Valparaíso, Valparaíso, Chile
{cristian.rusu, daniela.quinones}@pucv.cl,
jenny.morales.b@mail.pucv.cl

Abstract. Programmers and software developers are using different Integrated Development Environments (IDEs) to perform their daily work. IDEs are often complex applications, not friendly for novice programmers, with a learning process of several weeks and with usability and satisfaction of use not always as good as expected. The Programmer eXperience (PX) is a particular case of User eXperience (UX), based on the use of the IDEs and other artifacts. We have found studies about the programmer’s behavior and work, and also articles addressed the usability and new tools proposals for IDEs. In this work, we conducted a survey to evaluate the usability of several IDEs. The survey was based on the System Usability Scale (SUS), which we adapted for the purpose of our research. We focus the study on popular IDEs such as Dev-C++, Eclipse and NetBeans. The survey was conducted in two Chilean universities and one Spanish university, with students enrolled in two undergraduate programs in Informatics Engineering. The results obtained show that the IDEs evaluated have several issues related to the usability perceived by our participants. An interview was conducted with six experienced programmers that are working in different programming environments, in order to consult them on what aspects they would like to improve the IDEs. Their comments indicate that IDEs should incorporate connection with other programmers, and also, they claim for more intuitive interfaces and understandable error messages.

Keywords: Usability · User eXperience · Programmer eXperience · Integrated Development Environment · Survey · Interview

1 Introduction

The programming environment plays a very important role in the software development. Learning to program is a difficult process that requires several months or years. Students and practitioners interact with the different interfaces that provide the different programming environments or languages. Most of programming environment

interfaces do not accomplish with the basic principles of usability and/or accessibility, so the result is an environment difficult to learn and difficult to handle for users.

In this work, we present a study of the usability perceived by user of programming environments centered in three of them: Dev-C++, Eclipse and NetBeans. We selected these Integrated Development Environments (IDEs) because they are used in the Informatics Engineering program of the three universities where we conducted this study: Universidad Autónoma de Chile, Pontificia Universidad Católica de Valparaíso (Chile), and Universidad Miguel Hernández de Elche (Spain). We conducted a survey to students of these three universities and we performed several interviews to professionals working in different computer programming Chilean enterprises.

The results show in general low usability in the IDEs. Eclipse is the IDE that obtained the best score. Both NetBeans and Dev-C++ are considered as IDEs with a low degree of usability, obtaining the lowest score Dev-C++. The interviews provide interesting results on the preference of the participants and aspects to improve in the IDEs.

This work is organized as follows: in Sect. 2 we describe the background, in Sect. 3 we introduce the methodology used, in Sect. 4 results obtained are presented, and finally, Sect. 5 contains conclusions and future work.

2 Background

The work carried out by a programmer are challenging and allows the programmer to interact with various elements such as programming environments and other development artifacts. Usability is one of the important aspects of the programmer work. However, usability aspects are not enough, and some models of User eXperience explain more complete aspects that influence in the programmer experience. Some of these aspects can be the interaction of the programmer with systems, languages and programming environments to reach a more pleasant programming experience.

2.1 Usability

Usability is defined by the International Organization for Standardization (ISO) 9241-11 of 2018 as: “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” [1]. Usability has specific attributes that allow its assessment, such as: Learnability, facility to be learned and very important to reach for novice users; Efficiency, related to the speed in which the user can reach their objectives; Memorability, related to the ability of users not frequent to remember how the system is used; Errors, related to the number of errors that a user commits when performing a task; and Subjective Satisfaction, a subjective attribute that measures the subjective impression that the user has of the system [2].

2.2 User eXperience

Nowadays the User eXperience (UX) has become the concept very important for the systems, products or services. ISO 9241-210 of 2010 defines the UX as: “person’s

perceptions and responses resulting from the use and/or anticipated use of a product, system or service” [3]. There are several models that explain the user experience, one of them is Honeycomb, proposed by Morville [4]. This model considers that UX has seven aspects that go beyond of the usability. In this work, we take this model as a reference for systems such as IDEs. The aspects defined in Honeycomb model are:

- Valuable, the system must deliver value and contribute to the customer satisfaction.
- Usable, the system should be simple and easy to use. The usability is necessary but is not sufficient.
- Useful, the system or product must be useful and satisfy a need, otherwise there is no justification for the product.
- Desirable, the visual aesthetics of the product or system must be attractive and easy to interpret.
- Findable, the information should be easy to find and easy to navigate. The users can find what they need.
- Accessible, the system should be designed so that even users with disabilities can have the same user experience as others.
- Credible, the company and its products or services need to be reliable.

2.3 Programmer eXperience

Programmers use IDEs and several development artifacts, so UX concept may be particularized as Programmer eXperience (PX). Due to the complexity of the programmers’ tasks, and the fact that they interact with several artifacts of diverse nature, we may consider programmers as customers. The Customer eXperience (CX) concept is therefore relevant in the software development process.

In the review of articles that address the programmer’s experience, researches were found on the benefits of programming languages and graphic environments. These articles show that readability of the codes can be improved, presenting advantages over other that are textual and positively impacting the development of software [5, 6]. Development environments that implement graphic aspects can improve and facilitate various tasks of the programmer, such as the finding information in programming environments during software maintenance [7] and the monitoring and understanding of the code [8].

Software maintenance is an arduous task. Programmers requires to read and to analyze programmed codes. We found several works related to software maintenance: (1) how facilitate the reading of codes [9]; (2) how instructions can be more predictable by the reader [10]; (3) and how codes can be implemented in a more readable way [11, 12].

Two interesting studies about programmers were found. The first one evaluates whether the programmer’s experience influences the quality of the code he writes, obtaining as a result that years of experience are not a good predictor of programmer performance. A good predictor could be the academic background and the specialization [13]. The second study address on the productivity of the programmer, specifying new metrics to measure productivity for both for lonely programmers and programmers in pairs, these metrics are applicable to the entire life cycle of development [14].

2.4 Integrated Development Environments

IDE usually contains a source code editor, a debugger, automatic building tools, and some of them also have IntelliSense, a tool that helps programmers to fill automatically the code. One IDE can have a compiler, an interpreter or both. Examples of the most used IDEs are Eclipse, NetBeans and Visual Studio. It is worth to mention that an IDE can usually support several programming languages and also has a graphical interaction interface in which the tools available are displayed.

Several articles reviewed about IDEs showed the interest to improve the interaction of the programmer with the IDEs and to facilitate the work of the programmer. We identify: (1) the incorporation of complements to facilitate editing and changes in the codes [15, 16]; (2) the incorporation of social aspects and the benefits that brings for programmers [17, 18]; (3) the improvement of specific aspects of debugging that IDEs do not cover today [19]; (4) the incorporation of users that can define usability experiments integrated into the environments [20]; and (5) the implementation of a specific tool for IDE that was also analyzed in order to make programming easier and minimize errors [21].

We found studies about the usability of the programming environments, one of them show the need for a multitouch environment which contributes to ease of use [22]. Other studies explain that overloaded assistive features in environment present difficulties for the programmer [23].

One of the elements that programmers use in their work is Application Programming Interfaces (APIs). Studies on usability of APIs were found in relation to (1) the relevance of documentation about ease of use [24]; the most important element in the API is providing easy communication, which would favor and minimize the errors that can be generated in the development [25]; and (3) the ease of the use and implementation which indicates that their use is not as simple as expected [26].

3 Methodology

This study has two parts: (1) one survey about usability perceived by users of different IDEs based on the System Usability Scale (SUS) and (2) several interviews performed to different professional programmers in order to complement the answers obtained on the survey.

3.1 Survey

We conducted a survey to evaluate the usability of several IDEs. The survey was based on the SUS, which we adapted for the purpose of our research. SUS is a general tool developed by Brooke [27], (arguably) applicable to any type of interactive system. SUS includes 10 questions that allows to obtain a general measure of usability perceived by users that could cover several aspects of usability like effectiveness, efficiency and satisfaction. Each item of SUS consists of 5-point Likert Scale: 5 means “strongly agree” whereas 1 means “strongly disagree”. In all odd items, the result will be obtained by subtracting 1 from the participant’s response. The result of the even items

will be obtained by subtracting to 5 the answer of the participant. So, the scale for each item is from 0 to 4. The original total score of the survey is in the range [0–40] that will be converted to [0–100] by multiplying by 2.5 each item score. One system is considered “usable” when scores above 68. Table 1 shows the adapted survey applied in our study.

Table 1. Survey applied in our study.

No	Questions
Q1	I think I would like to use this IDE frequently
Q2	I found the IDE unnecessarily complex
Q3	I thought the IDE was easy to use
Q4	I think I would need the support of a technical person to be able to use this IDE
Q5	I found that the various functions in this IDE were well integrated
Q6	I thought there was too much inconsistency in this IDE
Q7	I would imagine that most people would learn to use this IDE very quickly
Q8	I found the IDE very cumbersome to use
Q9	I felt very confident using the IDE
Q10	I needed to learn many things before I could work with this IDE

Among the most used programming languages are C/C++ and Java [28]. We focus the study on popular IDEs that support these languages: Dev-C++, Eclipse and NetBeans.

The survey was carried out in two universities in Chile: Universidad Autónoma de Chile and Pontificia Universidad Católica de Valparaíso, and one university in Spain, Universidad Miguel Hernández de Elche. Participants were selected from students enrolled in undergraduate programs in Informatics Engineering.

The study involved 140 participants and was conducted from June to October 2018 in Chile and in January 2019 in Spain. Students participated in the survey voluntarily. In both countries, students from 1st year were asked to conduct the survey about Dev-C++ IDE. Since C is the initial language in the teaching of programming in the participating universities of the study. Other students with more experience in Chile from 2nd to 5th year answer a randomly about NetBeans, Eclipse and Dev-C++. In the case of Spain, students of the 2nd year were asked to fill the survey to evaluate Eclipse IDE, as they are working with Java in their second year. Moreover, students of 4th year were asked to fill both Dev-C++ IDE and Eclipse IDE as they have worked with both IDEs and have experience with more IDEs when they arrive at their last degree year; we split the group in two parts to get responses to both surveys.

The survey was conducted in the classrooms of the different universities, in students’ usual environment. Mainly we want them to answer about the IDEs that they have usually had the opportunity to work this semester and the previous semesters. The time allotted to answer the survey was 10 min maximum. Generally, the students responded in less time. The data was collected through a printed form in Chile, and a digital form in Spain (we used Google Forms).

3.2 Interviews

The purpose of the interview was to collect relevant information about the opinions of professionals who are users of development environments. The type of interview used was partially structured, in which six open-ended questions were considered, in which there is the possibility of deepening in some of them if necessary. The reason for selecting this type of partially structured interview allows to collect more information than the users and in turn to guide the interview process towards the points of interest.

The interview is composed of two initial questions that characterize the user. The third question is a general question about the environments they have used. The fourth question is focused on the advantages and disadvantages of IDEs. The two last questions are devoted to what they would like to improve in the environments.

The six questions defined in the interview are:

1. What IDE have you used?
2. What functionalities of the IDEs have you used?
3. Referring to the IDEs used Which one do you like best and which one less?
4. You can tell us three favorable things and three unfavorable things of the IDEs that you have used.
5. If you had to recommend an IDE, what would it be and why?
6. What would you like to improve in IDEs?

We contact professional informatics engineering graduates that have experience in software development (experience over 3 years) in Chilean enterprises. The interviewees were all male between 25 and 35 years old. They freely decided to participate in the interview. They were anticipated that the interview has six questions, so it would be short enough. The interviews were conducted in their offices directly and notes were collected in notebooks manually. Although the interview has well-defined questions, it was intended that the interviewee feel comfortable. The interviews last between 10 and 15 min.

4 Results

The analysis of the results of the survey will be carried out by interpreting single items from the SUS proposed by Sauro [29]. Five grades are detailed for SUS, from A to F, being A the highest grade and F the lowest. It also provides a description of scores for each question based on previous studies, which allows establishing whether the scores obtained are related to average scores or good scores. The analysis of the answers of the interviews will be carried out through a qualitative analysis for each question.

4.1 Dev-C++

This IDE was initially developed by Bloodshed Software until 2005 and then by Orwell since 2011. Its latest version available is from 2015. Dev-C++ allows programming in languages C/C++. This environment is used in the first year at all universities that participated in this work [30].

The total number of participants who answer about Dev-C++ in Chile is 35, of which 23 are first-year students. The general score obtained is 53.0, which places it in grade D of SUS, which means that it is very low as expected to be considered usable.

The lowest scores are in questions Q4 and Q10, what shows that students consider it difficult to use and complex to work. The highest scores were obtained in questions Q5 and Q6. The score obtained in Q5 does not reach to be average. In the case of Q6, his score places him in grade A, that is, a good score. This means that students consider Dev-C++ very consistency.

The total student that answered in Spain is 22, 13 of them are students of first year. The score obtained in general is 47.5 which is very low, considering it very unsatisfactory on the part of the students, this score places it in an F grade of SUS. The lowest scores were obtained by Q7 and Q10, which means that students consider that Dev-C++ is difficult to learn and difficult to use. The highest scores were obtained in Q2 and Q3, but only Q2 achieves an average score. In Fig. 1, we can see the results obtained in both countries.

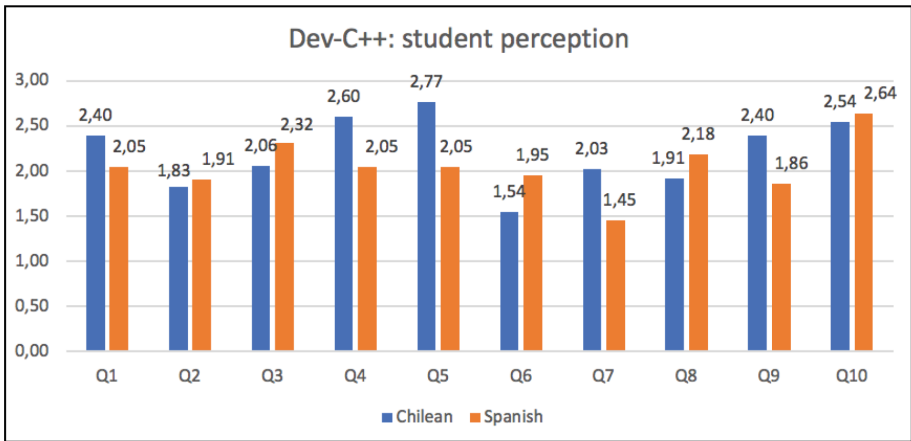


Fig. 1. Dev-C++: results (scale 0–4).

In both results we can see that the perception of students about the IDE is quite negative, especially in the case of Spanish students. Whereas Dev-C++ is complex, difficult to use and to learn. All the above indicates that the satisfaction of the students is very low.

4.2 Eclipse

Eclipse is an open source platform with a worldwide known IDE that allows to work with Java programming language and can be extended to other languages, such as the programming languages C/C++ and Python. Its latest version is from 2018 [31].

Eclipse was evaluated by 36 students in Chile in the 2nd, 4th and 5th years. The result obtained is of 60.9, which places it in grade D, so it has important aspects to

improve. The lowest scores were obtained in Q4 and Q10, being perceived by students as difficult to use and difficult to work. Questions Q1 and Q5 obtained the highest scores, but neither of them reached a minimum average score, so they are relevant to highlight as positive aspects perceived by the students.

In total 27 students from Spain belonging to 2nd and 4th year answered the Eclipse survey. The score obtained is 67.8 so the IDE is considered as usable by the students. The lowest scores are in Q4 and Q10, both reach an average score, this means that students consider moderately difficult to use and complex to work. The highest scores were obtained in Q1 and Q6. Only Q6 reaches a good score. So, the students consider that Eclipse is a consistent environment.

The results obtained show that Spanish students perceive that Eclipse is a usable IDE unlike Chilean students. In both cases the lowest scores were obtained in questions Q4 and Q10, with Chilean students evaluating it with the lowest results. In Fig. 2, we can see the results obtained.

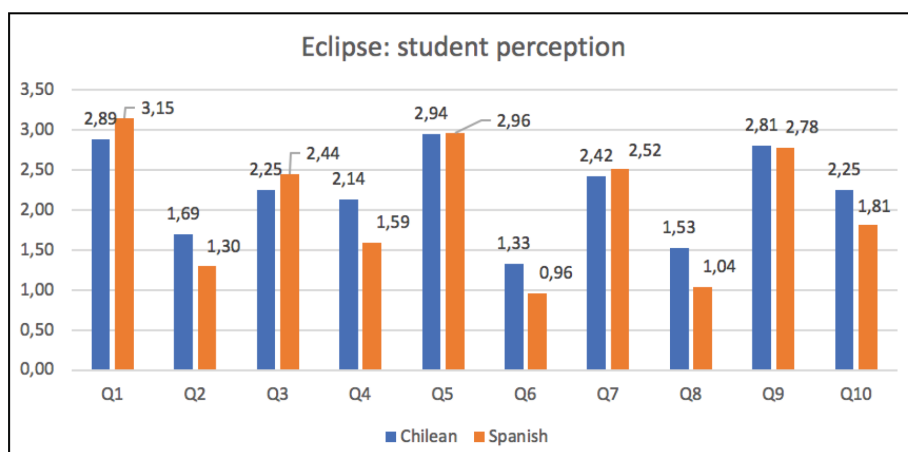


Fig. 2. Eclipse: results (scale 0–4).

4.3 NetBeans

NetBeans is developed by Apache Software Foundation and it is a free multiplatform IDE programmed in Java. Mainly for Java language developments, it can be extended to other languages. Its latest available version is from December 2018 [32].

NetBeans was only evaluated by 20 Chilean students in 3rd year. Spanish students have a transition from Dev-C++ to Eclipse directly, without using NetBeans, as is the case of Chilean students who follow the sequence, Dev-C++, NetBeans, and then Eclipse.

The results show that the total score obtained is 59.6. This result places NetBeans in a D grade, that is, it has aspects to improve to be more usable. The questions that have lower scores are Q4 and Q10, which means that students consider the complex to use and to work. The highest score was obtained in Q5, however it is not enough to obtain

an average score, so it could not be considered as a positive aspect in the students' perception. In Fig. 3, we can see the results.

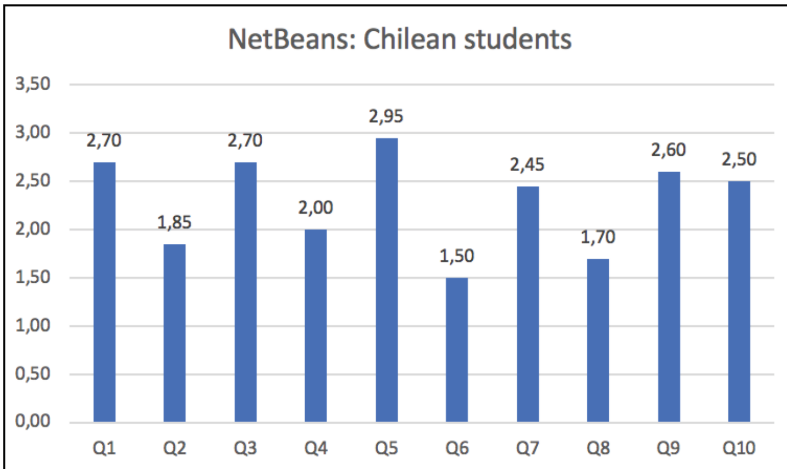


Fig. 3. NetBeans: results (scale 0–4).

4.4 Interviews

The analysis of the interviews made to the professionals indicates the following relevant results:

What IDE have you used? All of them had used NetBeans, Eclipse and Dev-C++. 2 of 3 had used Visual Studio whereas half of them Android Studio.

What functionalities of the IDEs have you used? The most used functionality is the debugging of code in IDEs (66% of respondents), followed by the function of auto fill code (33% of respondents), a functionality mentioned by one of the respondents in the search for definitions inside the code.

Referring to the IDEs used: Which one do you like more and which one less? The IDEs most liked by respondents are NetBeans and Visual Studio, each one with two preferences, followed by Eclipse and Visual Studio with one preference. As for the less preferred, Dev-C++ is mentioned twice, because they consider it is a very basic IDE that is rarely used in the work they do.

Can you tell us three favorable things and three unfavorable things of the IDEs that you have used? About NetBeans, the interviewees mentioned how fast the speed to create projects, besides supporting several languages. In the case of Eclipse, ease of use and speed to find errors stand out were the more mentioned favorable aspects. Favorable aspects of Visual Studio like familiarity and ease of use were also mentioned. In case of Android Studio, interviewees mentioned the speed of compilation and debugging. No participant referred to positive aspects of Dev-C++. About unfavorable aspects of Eclipse and NetBeans IDEs, the interviewees (50% of them) stated that they use many resources of the computer which makes them slow in their

performance. Others stated that IDEs in general have non-visible functionalities and that error messages are not understandable.

If you had to recommend an IDE, what would it be and why? In this question the interviewees responded in some cases with more than one recommendation. The most recommended IDE by the interviewees was NetBeans, recommended by 50% of them, due to the easiness to create projects and availability of complements. Followed by Visual Studio, recommended twice for its ease of use and documentation. Finally Eclipse, Android Studio and Dev-C++, were recommended only by one interviewee; in the case of Dev-C++ only for the use of programming in C language.

What would you like to improve in IDEs? Although this question is quite broad and general, it seeks to identify the functionalities that the users experienced considered necessary to implement. The answers show that what they would like to improve in the IDEs is access to forums internally, without having to leave the environment to communicate with the other programmers (50% of them considered it). It was also considered important to improve the generation of more documentation and in other languages besides English for beginning programmers (this is because the native language in Chile is Spanish). These answers about the social interaction with forums and the outside had already been found in a recent work of Astromskis et al. where the behavior of the programmers is monitored [33].

5 Conclusions and Future Work

In relation to the surveys conducted, the total number of students surveyed is 140, composed of 91 Chilean students and 49 Spanish students. This is a significant number for this study. As for Dev-C++, students perceive it as a system with low usability; especially Spanish students are more critical about it. Dev-C++ was considered complex, difficult to learn and difficult to use. The previous results were obtained with students from different courses of their degrees. The lowest results of the study were obtained by Dev-C++.

In relation to NetBeans, the results show that it is an IDE with important aspects to improve, being considered by the students as complex and difficult to work. This IDE obtained the average score of the three IDEs.

Clearly, the results of Eclipse were better, obtaining the best perception of usability by students both in Chile and in Spain. Eclipse achieved a score of 67.8 and was considered as an IDE quite usable by the Spanish students. It is also considered that it has aspects to improve such as the difficulty of use and the complexity to work it.

The interviews offered interesting conclusions about the most used aspects of IDEs, such as debugging and auto-filling code functionalities. In addition to the consideration that Dev-C++ is the least used in the work that the interviewees perform. The ease of creating projects appears also as a relevant aspect of NetBeans. As well Eclipse and NetBeans are considered slow IDEs and they require too many resources of the computer. An interesting finding is the need for programmers to connect with others through the programming environment.

As future work, the segmentation of the students and the comparison of the obtained results will be considered. This study could be refined by considering

equivalent years of the degree of the participants. In Chile, surveys have been conducted between students of 5 different levels (1st, 2nd, 3rd, 4th and 5th year courses) and in the case of Spain surveys have been conducted by students of only 3 different levels (1st, 2nd and 4th year course), so it would be interesting to establish a more equitable relationship in terms of the course studied.

In the interviews arise new IDEs that programmers use in their work such as Visual Studio and Android Studio. These two IDEs will also be included in surveys of our future works. In addition, we will complement the interviews with more professionals from both Chile and Spain to find more interesting and comparable results.

Acknowledgment. We are grateful to all the students that have participated in the surveys and interviews from the Universidad Autónoma de Chile, Pontificia Universidad Católica de Valparaíso (Chile) and Universidad Miguel Hernández de Elche (Spain). Jenny Morales is a beneficiary of one INF-PUCV doctoral scholarship.

References

1. ISO 9241-11: Ergonomics of human-system interaction- Part 11: Usability: Definitions and concepts (2018)
2. Nielsen, J.: Usability Engineering. AP Professional (1993)
3. ISO 9214-210: Ergonomics of human system interaction-Part 210: Human-centred design for interactive system, Switzerland (2010)
4. Morville, P.: User experience honeycomb. http://semanticstudios.com/user_experience_design/. Accessed 14 Jan 2019
5. Hollmann, N., Hanenberg, S.: An empirical study on the readability of regular expressions: textual versus graphical. In: 2017 IEEE Working Conference on Software Visualization (VISSOFT), Shanghai, China, pp. 74–84. IEEE (2017)
6. Zhang, Y., Surisetty, S., Scaffidi, C.: Assisting comprehension of animation programs through interactive code visualization. *J. Vis. Lang. Comput.* **24**(5), 313–326 (2013)
7. Athreya, B., Scaffidi, C.: Towards aiding within-patch information foraging by end-user programmers. In: 2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), Australia, pp. 13–20. IEEE (2014)
8. Karrer, T., Krämer, J.P., Diehl, J., Hartmann, B., Borchers, J.: Stackexplorer: call graph navigation helps increasing code maintenance efficiency. In: Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, USA, pp. 217–224. ACM (2011)
9. Beelders, T.R., du Plessis, J.P.L.: Syntax highlighting as an influencing factor when reading and comprehending source code. *J. Eye Mov. Res.* **9**(1) (2015)
10. Stefik, A., Siebert, S.: An empirical investigation into programming language syntax. *ACM Trans. Comput. Educ. (TOCE)* **13**(4), 19 (2013)
11. Sedano, T.: Code readability testing, an empirical study. In: 2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET), USA, pp. 111–117. IEEE (2016)
12. Kraeling, M.: Embedded software programming and implementation guidelines. In: Software Engineering for Embedded Systems, pp. 183–204 (2013)

13. Dieste, O., et al.: Empirical evaluation of the effects of experience on code quality and programmer productivity: an exploratory study. *Empirical Softw. Eng.* **22**(5), 2457–2542 (2017)
14. Solla, M., Patel, A., Wills, C.: New metric for measuring programmer productivity. In: 2011 IEEE Symposium on Computers & Informatics (ISCI), Malaysia, pp. 177–182. IEEE (2011)
15. Vakilian, M., Chen, N., Zilouchian Moghaddam, R., Negara, S., Johnson, R.E.: A compositional paradigm of automating refactorings. In: Castagna, G. (ed.) ECOOP 2013. LNCS, vol. 7920, pp. 527–551. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39038-8_22
16. Yoon, Y., Myers, B.A.: Supporting selective undo in a code editor. In: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering (ICSE), vol. 1, pp. 223–233. IEEE (2015)
17. Hundhausen, C.D., Carter, A.S.: Supporting social interactions and awareness in educational programming environments. In: Proceedings of the 5th Workshop on Evaluation and Usability of Programming Languages and Tools, USA, pp. 55–56. ACM (2014)
18. Bravo, C., Duque, R., Gallardo, J.: A groupware system to support collaborative programming: design and experiences. *J. Syst. Softw.* **86**(7), 1759–1771 (2013)
19. Salvaneschi, G., Mezini, M.: Debugging for reactive programming. In: Proceedings of the 38th International Conference on Software Engineering, pp. 796–807. ACM (2016)
20. Humayoun, S.R., Dubinsky, Y., Catarci, T.: UEMan: a tool to manage user evaluation in development environments. In: Proceedings of the 31st International Conference on Software Engineering, pp. 551–554. IEEE Computer Society (2009)
21. Coblenz, M., Nelson, W., Aldrich, J., Myers, B., Sunshine, J.: Glacier: transitive class immutability for Java. In: 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE), Argentina, pp. 496–506. IEEE (2017)
22. Bellucci, A., Romano, M., Aedo, I., Diaz, P.: Software support for multitouch interaction: the end-user programming perspective. *IEEE Pervasive Comput.* **15**(1), 78–86 (2016)
23. Dillon, E., Anderson, M., Brown, M.: Comparing feature sets within visual and command line environments and their effect on novice programming. In: Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, USA, p. 675. ACM (2012)
24. Endrikat, S., Hanenberg, S., Robbes, R., Stefik, A.: How do API documentation and static typing affect API usability? In: Proceedings of the 36th International Conference on Software Engineering, India, pp. 632–642. ACM (2014)
25. Bastos, J.A., Afonso, L.M., de Souza, C.S.: Metacommunication between programmers through an application programming interface: a semiotic analysis of date and time APIs. In: 2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), USA, pp. 213–221. IEEE (2017)
26. Gonçalves, R., Amaris, M., Okada, T., Bruel, P., Goldman, A.: OpenMP is not as easy as it appears. In: 2016 49th Hawaii International Conference on System Sciences (HICSS), USA, pp. 5742–5751. IEEE (2016)
27. Brooke, J.: SUS-A quick and dirty usability scale. *Usability Eval. Ind.* **189**(194), 4–7 (1996)
28. Parsons, D., Susnjak, T., Mathrani, A.: Design from detail: analyzing data from a global day of coderetreat. *Inf. Softw. Technol.* **75**, 39–55 (2016)
29. Sauro, J.: Interpreting single items from the SUS. <https://measuringu.com/sus-items/>. Accessed 15 Jan 2019
30. Dev-C++ Blog. <http://orwelldevcpp.blogspot.com/>. Accessed 15 Jan 2019
31. Eclipse Foundation. <https://www.eclipse.org/>. Accessed 15 Jan 2019
32. Netbeans. <https://netbeans.org/>. Accessed 14 Jan 2019
33. Astromskis, S., Bavota, G., Janes, A., Russo, B., Di Penta, M.: Patterns of developers behaviour: a 1000-hour industrial study. *J. Syst. Softw.* **132**, 85–97 (2017)