2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)

Achievements Visualization in Programming Education

Kaisei Hanayama, Shinsuke Matsumoto, Yoshiki Higo and Shinji Kusumoto
Graduate School of Information Science and Technology, Osaka University, Japan
{k-hanaym, shinsuke, higo, kusumoto}@ist.osaka-u.ac.jp

***Abstract*—Programming education focusing on the correctness of program behavior is often conducted with automated testing. However, it is difficult to instruct students about internal program quality since automated testing confirms only the external behavior of the program. Although there are a lot of quality measuring tools, results in these tools are so detailed or complicated that students cannot handle them. Besides, these tools do not give explicit criteria to judge the quality of the given program, thus students cannot figure out how high their program quality is. As a result, using these tools does not lead to improving the program quality. In this paper, we propose an educational method in which visualize the program quality by introducing a concept called *achievement*. We also introduce an implemented prototype system, *Ave*, to realize the proposed method.**

***Index Terms*—programming education, automated testing, program quality, achievement, visualizing**

I. INTRODUCTION

There are many methods in programming education based on automated testing, in which submitted programs are built and tested automatically [1]. Evaluation in such methods is performed by whether they passed the tests teachers defined in advance. We call such methods as *test-based education* in this paper. Test-based education has a certain effect on motivating students for programming since the success or failure of their assignment can be determined automatically and immediately. It is also capable of reducing the burden on teachers because they no longer need to check all the assignments [1].

However, test-based education only focuses on the external behavior of the program. Therefore internal structure, that is, the quality of the program tends to be neglected [2].

The quality measurement tools, such as PMD or Checkstyle, have been actively developed. These tools can objectively show the program quality by static code analysis. Besides, CI tools (e.g., Jenkins and Travis CI) have a certain compatibility with these tools, and it is possible to perform program building, testing, and quality measurement in automatic.

However, the quality measurement tools are supposed to be used by practical developers [3], so the result in these tools may be over-detailed or complicated for students to utilize them. Furthermore, measurement tools do not give an criteria to judge the program quality so students cannot judge how high their-created program quality is. For the above reasons, we assumed that applying such quality measurement tools may be an ineffective approach in test-based education.

In this paper, we propose to introduce a concept called *achievement* into test-based education. It aims to improve the quality of student-created programs and reform awareness of students for program quality. In the proposed method the quality of the program which passed the tests is measured. By processing measurement results to achievements and presenting it to the student, the quality of submitted program can be shown in a concise way. In addition, these flows can easily be integrated into test-based education because all of them are performed automatically. We also implemented *Ave* as a prototype system to realize the proposed method.

II. PROPOSED METHOD

A. Introduce achievement into test-based education

The purpose of this study is to take up the program quality at test-based education in order to improve the quality of student-created programs and reform awareness of students for program quality. To achieve this goal we propose to introduce *achievement* into test-based education. Achievement is a concept which is used in video games, and it shows evidence or results obtained by satisfying certain scores. There are many studies considering about introducing the elements of gamification[1] into programming education [5]. These studies have shown that it has a certain effect on motivating students for class and raising grades, so we assume that introducing achievement into test-based education will also be effective.

Some items which characterize the program quality are set as achievements and introduced into the test-based education in this study. Adopting achievements makes it possible to set evaluation criteria for the program quality. It is also capable of indicating students what does it mean that program quality is high and how to improve the quality of their program in an easy-to-understand way. Moreover, awareness for improving program quality and motivation for programming can be encouraged by introducing elements of gamification.

B. Types of achievements

In formulating the achievements, the program quality covered in this study was classified into two types:

- *metrics-based achievements*: achievements based on quantitative measurement items such as “maximum lines of code per method is under 30” or “average code coverage of each method is above 80%”

[1]application of game-design elements and game principles in non-game contexts. [4]

978-1-7281-2607-4/19/$31.00 ©2019 IEEE
DOI 10.1109/COMPSAC.2019.00144

IEEE computer society

- *rule-based achievements*: achievements based on binary measurement items such as "there is no empty `if` block" or "unnecessary variables do not exist"

For metrics-based achievements, three stages of difficulty (Lv.1 to Lv.3) with different thresholds are set, which enables to establish criteria for program quality and help students to comprehend. Also, setting Lv.1 to relatively low difficulty makes achievements easier to accomplish and motivates students to think about program quality during implementation.

C. Visualization and comparison of achievements

The proposed method employs visualizing progress of achievements and comparing them with other students. In the test-based education, the task to be achieved is strictly stipulated by teachers as test cases. Each student implements their own program which will accomplish that task, so their way of implementation may differ. Therefore, visualizing the status of achievements of their program makes it possible to briefly show the quality of them and enable students to judge quality level at a glance. Competitiveness of students and motivation for programming also can be improved by comparing the accomplished achievements with other students taking the same class.

III. IMPLEMENTED SYSTEM

We implemented *Ave* as a prototype system to realize the proposed method. Ave is an abbreviation for *Achievements Visualization in programming Education*. Fig. 1 shows one of the Ave views, the confirmation view of achievement status. This view shows the status and the description of each achievement, and the number of accomplished achievements. Buttons marked with a check mark ("✓") indicate accomplished achievements and with an exclamation mark ("!") indicate unaccomplished achievements. A summary of achievement is displayed when hovering the cursor on the button. Clicking the button jumps to API documentation, in which it is able to check more detailed content such as code examples, benefits of accomplishing achievements, columns related to programming, etc. We wrote this API documentation concurrently with the implementation of Ave. The program quality can be shown briefly by visualizing status of achievements. Also, it is possible to provide various knowledge and motivate students to think about the program quality by API documentation.

Ave is being introduced to the class for undergraduate students as an application experiment. Fig. 2 shows the flow of the class of application experiment. As shown in Fig. 2, this class is test-based education, and employs some external tools; Jenkins, PMD, Checkstyle, etc. The blue arrows represent the flow of the conventional method and the orange arrows represent the proposed method. In the proposed method, measurement of program quality and processing thereof are also performed automatically, in addition to the flow of conventional method. Therefore, we are going to confirm whether the introduction of Ave is effective to improve the quality of student-created programs and awareness of students for program quality.

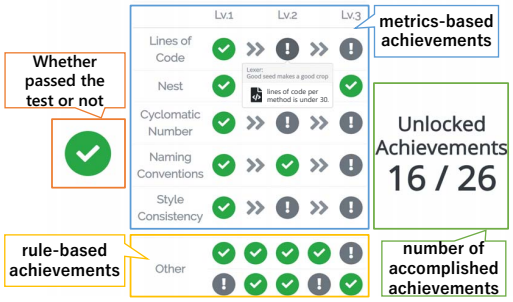

Fig. 1: The confirmation view of achievement status in Ave

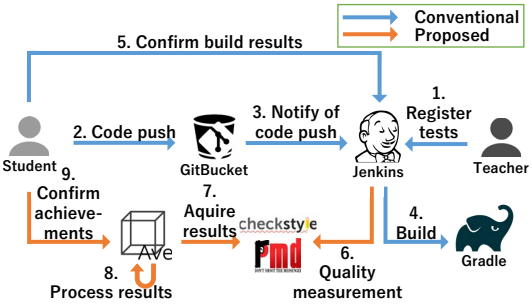

Fig. 2: The flow of the class of application experiment

IV. CONCLUSION

In this paper, we proposed the method for programming education based on automated testing, in which visualize the program quality by introducing a concept called *achievement*. We also introduced an implemented prototype system, *Ave*, to realize the proposed method.

As a future work, we will study appropriate achievement items and thresholds using the data obtained in application experiment. We are also going to make Ave more flexible and proper. At present Ave is specialized for the class of application experiment, but a similar system can be constructed in other lessons if it is test-based education. Therefore, in order to be applicable to other classes, we would like to generalize the specialized parts in Ave and to improve flexibility.

ACKNOWLEDGMENT

This research was partially supported by JSPS Grant-in-Aid for Scientific Research (JP25220003, JP26730155).

REFERENCES

[1] J. A. Harris, E. S. Adams, and N. L. Harris, "Making program grading easier: but not totally automatic," *Journal of Computing Sciences in Colleges*, vol. 20, no. 1, pp. 248–261, 2004.

[2] S. H. Edwards, "Rethinking computer science education from a test–first perspective," in *SIGPLAN Conference on Object–Oriented Programming, Systems, Languages, and Applications*, 2003, pp. 148–155.

[3] P. Tomas, M. J. Escalona, and M. Mejias, "Open source tools for measuring the internal quality of java software products. a survey," *Computer Standards and Interfaces*, vol. 36, no. 1, pp. 244–255, 2013.

[4] K. Huotari and J. Hamari, "Defining gamification: A service marketing perspective," in *International Academic MindTrek Conference*, 2012, pp. 17–22.

[5] G. Barata, S. Gama, J. Jorge, and D. Goncalves, "Engaging engeneering students with gamification," in *International Conference on Games and Virtual Worlds for Serious Applications*, 2013, pp. 1–8.