Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 15, no. 11, November 2019, pp. 2944-2951
DOI: 10.23940/ijpe.19.11.p13.29442951

Crowdsourced Testing Ability for Mobile Apps: A Study on Mooctest

Peng Yang[a], Jin Xu[b,c], Hongyu Sheng[d,*], Yong Huang[b], and Jianfeng Xu[b]

[a]*School of Information Engineering, Guangzhou Panyu Polytechnic, Guangzhou, 511483, China*
[b]*Mooctest Inc., Nanjing, 210000, China*
[c]*Software Testing Engineering Laboratory of Jiangsu Province, Nanjing, 210000, China*
[d]*Beijing Key Lab of Information Service Engineering, Beijing Union University, Beijing, 100101, China*

Abstract

As an emerging trend of software testing, crowdsourced testing is increasingly attracting attention from various fields, such as education and industry. In this paper, we studied the ability of crowdsourced testing for teaching. We mainly aimed to find the relationship between students' crowdsourcing ability and their performance in software testing courses, so as to help schools devise a better teaching model and further improve teaching quality. Our study included two parts. First, we published crowdsourced tasks of a mobile application to students on a well-known crowdsourced testing platform. Then, we conducted a follow-up survey to collect more information about the students. We surprisingly found that the superiority of a student's grades can be reflected with higher crowdsourced testing skills to an extent, but not as an absolute measure. More importantly, practical crowdsourced testing is naturally complementary to purely theoretical teaching.

Keywords: crowdsourced testing; educational project; testing skills; grades

(Submitted on August 10, 2019; Revised on October 10, 2019; Accepted on November 15, 2019)

1. Introduction

In the era of rapid development of mobile Internet, the life cycle of software products has been compressed dramatically. The traditional software testing cycle is too long, and most of the automated scripts are used for testing traversal in the testing process, which cannot simulate real use scenarios of the user. How to quickly get the real feedback from a large number of users to complete the real test task with lower cost and higher efficiency has become a difficult problem. Crowdsourced technology can solve the above problems to a certain extent. Crowdsourcing is a distributed problem solving model. Enterprises use the Internet platform to release tasks, and through the economy or experience, they motivate crowdsourced participants to share time, ideas and knowledge in a free and voluntary manner.

Crowdsourced testing is a branch of crowdsourced software engineering. Compared with crowdsourced software engineering, there is little correlation between testing tasks and these tasks can be executed concurrently. Crowdsourced testing can simultaneously invite a large number of users to participate in the test task online. The real scene and user operation needed in practical application can be simulated completely. Usually, the whole test cycle is short and the cost is low. What's painful is that, unlike traditional testing methods [1-2], crowdsourced testing [3-4] is carried out by many different volunteers from different places rather than by hired consultants and professionals. These volunteers generally are less experienced and tend to lack adequate training. This makes it difficult for task owners to control the testing process and testing quality, especially for professional software tests [5]. Crowdsourcing companies find it difficult to employ a large number of professional testing volunteers, which makes crowdsourced testing only a supplement to software testing. However, students in colleges and universities majoring in software engineering have a chance to participate in crowdsourced testing as volunteers; this, to a great extent, makes up for the weakness of volunteer technology. Thus, it can be combined with the teaching process to improve teaching quality.

Chen et al. proposed "quasi-crowdsourced testing" [6] and tried to combine educational projects with crowdsourced testing. Students who had a certain social relationship and did not request high payment could participate in the enterprise

* Corresponding author.
E-mail address: jzw@buu.edu.cn

software project as testers, which achieved an effective link between teaching theory and project practice. Their research focused on the superiority of crowdsourced testing to the cultivation of students' comprehensive testing ability compared with other testing projects. In this paper, based on the research results of Chen et al., we will further analyze the potential links between students' grades and the differences in crowdsourced testing ability among different students by analyzing specific test skills required for crowdsourced testing.

Our study included the following steps. First, we studied students' scores and their performance in a test project related to crowdsourced testing (i.e., mobile application testing). This aimed to better understand the relationship between students' crowdsourced testing ability and their course performance as well as determine students' attitude towards crowdsourced testing that is potentially related to their course scores. Then, we analyzed the effects of crowdsourced testing on students' specific testing ability in different scenarios and the significant differences in students' scores. After that, we conducted a follow-up survey [7] to collect and record the scores of these students in software testing courses and explored the efficiency of students' basic knowledge of software testing in crowdsourced testing practice. Finally, we asked them to briefly evaluate the crowdsourced testing in order to better clarify their attitudes and viewpoints. Our major contributions are as follows:

- We evaluated the testing abilities of undergraduate and junior college students by analyzing their performance in a test project.
- We reviewed test reports submitted by students and conducted a questionnaire survey on the students participating in the crowdsourced testing to determine the relationship between students' testing ability and their grades.
- We emphasized that the grades of the curriculum cannot be used as a sole criterion for measuring the true test level of students and gave some suggestions for the future teaching work of colleges and universities.

2. Test Project

2.1. Crowdsourced Testing Platform

The crowdsourced testing platform selected in this paper integrates the recommendation system for collaborative crowdsourced testing, the quality control system for collaborative crowdsourced testing, and the crowdsourced review and delivery system based on the test report summary.

The recommendation system for collaborative crowdsourced testing will complete the information sharing and task allocation in the whole process of the test task. Crowdsourced workers are both submitters (submitting test cases and test reports) and reviewers (reviewing other workers' test reports) throughout the manual testing. We take full advantage of collaboration among workers and the guidance recommended by the platform to improve the quality of reports. During the period, workers can choose to like or dislike the recommended reports. The public test platform in the report integration stage can refer to this information to make a preliminary evaluation of the quality of the report. At the same time, workers can also turn to co-edit, that is, Fork, to modify the recommended reports. The testing method of the crowdsourced workers mentioned above can be seen in Figure 1.

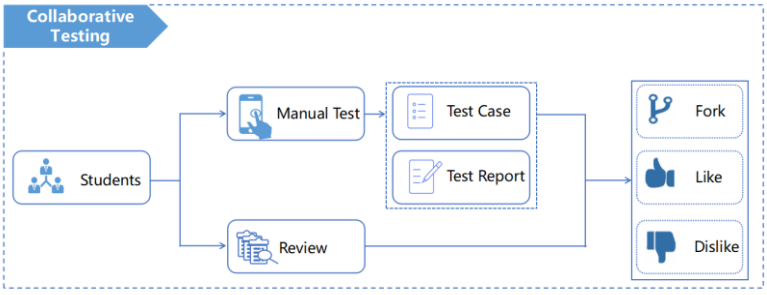

Figure 1. Testing method of the crowdsourced workers

The main functions of the quality control system are to raise the awareness of crowdsourced workers about test quality, to evaluate and ensure the quality of test reports, and to identify and eliminate malicious crowdsourced workers in advance by taking appropriate test quality control measures, testing the effectiveness of output, evaluating automatically, monitoring feedback and auditing. What's more, the crowdsourced review and delivery system is divided into review and delivery modules, text process modules, image process modules, aggregation modules and summary modules. This system has

effectively aggregated similar reports and reasonably organized redundant information, which enables the reviewers to identify the helpful information more quickly and accurately. Furthermore, it improves the efficiency of the report review, guarantees the quality of the report delivery, and provides a better delivery of the test product to the task initiators.

On the other hand, crowdsourced testing mainly includes three parts: a task initiator, a crowdsourced testing platform, and students (crowdsourced workers). We can see from Figure 2 that the general process of crowdsourced testing consists of three phases: task preparation, task execution, and test report integration [8]. In crowdsourced testing, all test reports are visible to the students, who complete the test reports in a form of Wiki1 (a multi-person collaborative writing system). Students can not only submit test reports, but also review test reports submitted by others. This can help crowdsourced testing platforms identify high-quality test reports. Such crowdsourced testing feeds students with various kinds of tasks by providing specific ways of collaboration, which gives students a stronger sense of participation and achievement.

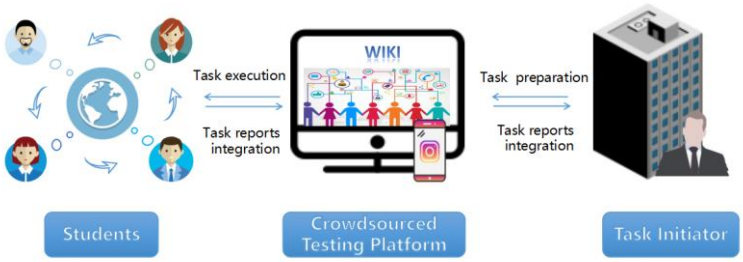

Figure 2. The general process of crowdsourced testing

2.2. Mobile Application Testing

Mobile application testing [9] is a complex crowdsourced testing task that tested students' software testing ability in many aspects. We chose a comic reading software (an Android mobile app) called "ComicsIsland". Students needed to manually test the software for functionality, covering as many paths as required by specific test requirements [10]. Students were allowed to create test cases and filled out test reports (detailed descriptions and screenshots of specific questions) after a bug was discovered during testing.

In order to better analyze the test efficiency of students under the condition of this simulation test, we fully considered the impact of different educational environments on students' test abilities and set up a group of comparative experiments. Crowdsourced workers were made up of undergraduates and junior college students, and the proportion was close to one to one. In the case of a consistent test environment, the same test tasks and test time, both experimental groups carried out a crowdsourced test based on the mobile application (ComicsIsland) mentioned above. Throughout the test, we monitored and recorded their tests in real time.

2.3. Test Analysis

First, we studies and analyzed the whole testing process. The crowdsourced test lasted four hours. We counted the total number of bugs found by these two types of students every half hour in order to determine whether there were any differences in the ability of students with different academic backgrounds in the practice of crowdsourced testing. The statistical details are shown in Figure 3.

From Figure 3, we can see that the red curve (i.e., the number of bugs found by junior college students) generally increases steadily, while the blue curve (i.e., the number of bugs found by undergraduates) tends to be flat three hours after the beginning of the test; it can almost be seen as part of a parabola.

Without considering the effectiveness of the bugs that have been discovered, it is easy to conclude that the ability of junior college students to identify defects is significantly stronger than that of undergraduates. At the beginning of the test, the number of bugs found by junior college students was slightly higher than that of undergraduates. As time passed, the gap between the number of bugs found by the two types of students grew. We further explore the differences between the two

[1] https://en.wikipedia.org/

types of students' testing ability in each test phase. We divided the entire testing process into early test and late test, with the third hour of the test acting as the dividing line. The early test mainly focused on the ability of students to find obvious or common defects, which is the most basic testing ability of software engineering students. During this period, junior college students performed slightly better than undergraduates. In the late test, students needed to mine hidden bugs and associated bugs (one defect may lead to other defects), which tests students' ability to find hidden dangers and root causes of defects. During this period, undergraduates were clearly at a disadvantage.

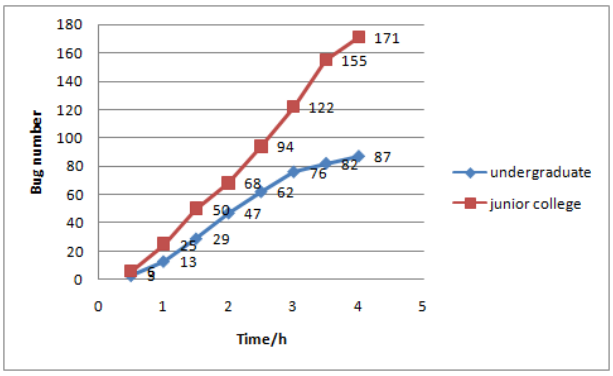

Figure 3. The total number of bugs

The assessment of test progress is only a small part of our work of measuring differences in testing abilities among students, and the main focus is to check the quality of test reports submitted by students. The analytical work of crowdsourced testing is mainly divided into three parts: the review of use case design, the quality check of bugs that have been found, and the evaluation of mutual reviews among students. The first and second parts are individual report scores for each student, while the third part is the score for mutual review among students. The distribution of the final comprehensive score is shown in Figure 4.

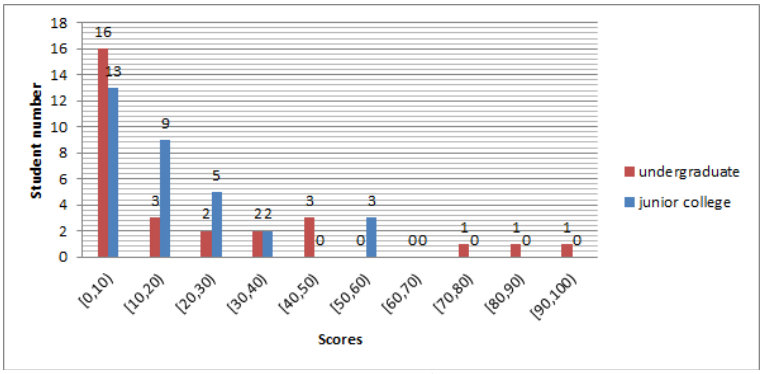

Figure 4. The distribution of students' scores

From Figure 4, we can observe that the overall score of students is somewhat poor: some students received more than 90 points, while many students only received a few points. The number of students in the middle score was also very sparse. The number of students who passed the test (with a score of more than 60) was only three. The number of students with a score of less than 20 accounted for the vast majority of all students who submitted reports (only students who submitted reports are scored). In other words, most students did not meet the basic standards set by enterprises in industry. It is even more surprising that only three people are undergraduates, and none of the junior college students scored more than 60. This is inconsistent with our predictions. The data in Figure 3 shows that the number of bugs that junior students found was 171, which is nearly twice the number of bugs that undergraduates found (87). From a quantitative point of view, the testing ability of junior college students far exceeds that of undergraduates, and in terms of the quality of test reports, the ability of junior college students to find effective bugs needs to be strengthened. In other words, their testing efficiency is weak. Taking into account the multi-angle analysis and thinking, there is still a certain gap between the testing ability of junior college students and the testing ability of undergraduates. Such results are within expectations given that the two types of students receive different teaching and training.

In addition, according to our statistics, the total number of students participating in the education program is as high as 168, and the final number of students' test reports is only 61, of which only 42 are valid. The remaining 19 invalid test

reports are either empty, or the proposed bug is invalid. In other words, these bugs are not real bugs, which means that the bug ability and defect analysis and judgment ability of many students are weak. However, these abilities are positive and important for them to participate in test-related work in the future. Moreover, we noticed several characteristics when looking through these test reports:

1) Generally speaking, the higher a student's final score is, the more test requirements were covered by the student's test cases and the more effective bugs he/she could find, and these bugs were more clearly and logically described.
2) There are many students who can find effective bugs but cannot correctly describe those bugs, which results in low scores.
3) Many students have weak collaborative abilities and do repetitive work, which leads to the emergence of a large number of duplicate test reports.
4) There is also a very special case in which a very small number of students find that the number of bugs is rare and the score is not high; most of these are valuable bugs that are very important but are not found by other students. For example, some students take into account the particularity of this crowdsourced testing (mobile application testing) and test the operation of the "ComicsIsland" app in different network environments to find the existence of bugs.
5) There is also an extreme situation in which a very small number of students have very poor quality of test reports: using a short sentence such as "I feel defective" and "yes" to describe bugs.

For the first four phenomena, we believe that assessing a student's crowdsourced testing ability needs to be considered from multiple perspectives, such as basic testing skills, writing expression skills, collaboration skills, and defect insight ability (hidden bug discovery). This is undoubtedly a good inspiration for educators; it is not limited to the teaching of theory, but also the multi-dimensional quality training to improve students' comprehensive testing ability. The last situation demonstrates that some students' attitudes towards crowdsourced testing are not serious or correct. We can see that this kind of student is not interested in crowdsourced testing such that the writing of test reports is simple and even perfunctory. Through the above content, our intuitive feeling is that the ability differentiate students in crowdsourced testing is reflected in the attitude of individual students. Only on the premise that students take the task of testing seriously can our study be effective and meaningful.

3. Student Survey and Feedback

After the completion of the test project, we conducted a real-name online questionnaire survey on the students who had submitted test reports in order to understand students' personal thoughts and their actual course scores in school. We received 38 survey responses and filtered out some invalid responses (e.g., a questionnaire without the real name is considered invalid as we do not know the student's grades). 28 valid survey reports remained after filtering. Due to the limitations of the paper, we only select some of the questions that are closely related to this paper to illustrate our research results.

- *Q1: What is your attitude towards crowdsourced testing?*
- *Q2*: *What is the purpose of your crowdsourced testing*?

According to Figure 5, although only 10.71% of the students did not want to participate in the crowdsourced testing at all, up to 64.29% of all students were not very interested in the crowdsourced testing. In the process of learning, students generally feel the content of software testing courses is abstract and boring, and their interest in learning is not high. The teaching of software testing courses mostly shows all kinds of testing skills by means of manual testing, and the teaching cases usually come from textbooks or mini programs, which were developed by teachers in the past. Students cannot fully understand the functional requirements of the test cases, but often do the test with the mentality of completing the exercise. Their lack of interest in learning ultimately leads to poor learning results. Now, there is a big problem, that is, after taking software development and software testing courses, students do not have a clear understanding of the complete development process of the product. This shows that there are still great shortcomings in our education system.

Even though there has been a significant increase in the number of students participating in crowdsourced testing compared with other traditional tests in the past, crowdsourced testing is still not very attractive to students. As for the reasons why students participate in crowdsourced testing, according to Table 1, the main purpose of most students was to improve their academic performance, improve their personal testing ability, and find better employment in the future. Very few students were under pressure from teachers or blindly followed others. After checking the course scores of these 28 students, and we find that those who were not interested in crowdsourced testing did not perform well in software testing-related courses. The students who participated in crowdsourced testing for academic and personal development usually achieved good or excellent grades. 85.71% of the students who were strongly interested in crowdsourced testing did very

well in the course, and one of the main reasons they took part in the testing was to boost their self-confidence. This shows that crowdsourced testing plays a positive role in encouraging students to an extent.

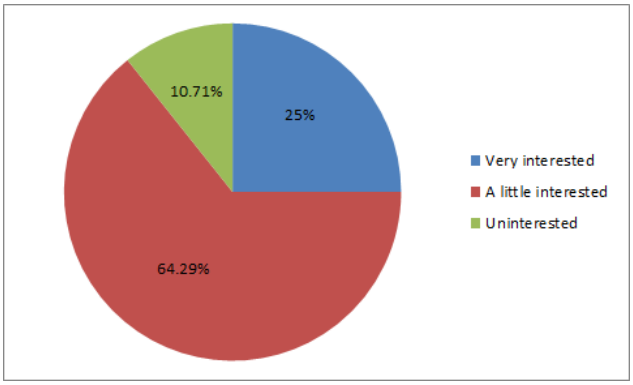

Figure 5. Students' attitudes

Table 1. Purpose of students' participation in crowdsourced testing

No.	Answer	# of Answers	% of Answers
A	Consolidate the knowledge base and get high scores in the course	22	78.57%
B	Interest	8	28.57%
C	This is a good opportunity to get in touch with society in advance	16	57.14%
D	Boost self-confidence	7	25%
E	Prize, bonus	8	28.57%
F	Improve test skills	14	50%
G	Asked by the teacher because of professional relevance	3	10.71%
H	Congregational psychology	2	7.14%
I	Others	0	0%

- *Q3: What is your gender?*
- *Q4*: *How close/relevant do you think the software testing knowledge you learned is to test projects*?

Through the students' course scores, we find that female students' performances were generally higher than that of male students. However, in the practice test project, nearly 80% of the bugs found by female students were invalid or incorrect, resulting in the final test scores being extremely low. In contrast, male students performed much better in finding valid bugs. There are many male students who can quickly find bugs and accurately describe them. However, the statistics in Figure 6 show that the vast majority of students, regardless of gender, believe that the content of the teaching is very relevant to the actual test project, and there is not much derailment. Therefore, we can believe that a student's academic performance is not the only criterion for measuring his or her true test ability. Maybe, we can find some relevant reasons from the teaching. There are many theoretical contents in the software course, and the knowledge points are too trivial. It is difficult for students to construct a more complete knowledge system structure after learning. Software testing is a very important course in the course system of software engineering, which is rich in theoretical content and practical. Software testing courses involve a lot of professional terms, test methods and skills, which require students to have a higher level of logical thinking ability than hands-on ability. Simple knowledge point descriptions and independent test technique exercises make it easy for students to fall into the whirlpool of knowledge, and so it is difficult for students to construct a sound and orderly knowledge structure in their minds.

More importantly, the fact that a student's test knowledge is rich but the test ability is very weak reflects the lack of teaching practice to an extent. Judging from the current teaching conditions of software testing in colleges and universities, it is difficult for both the testing environment and the testing equipment to meet the practical teaching needs of software testing courses. Furthermore, there is a discrepancy between the testing requirements and the testing tools used in the daily work of enterprises. This has a great impact on the training of high-quality and high-level professional testers.

What's worse is that there is a lack of coherent case teaching in class. Because not all software testing teachers have rich experiences in software development, most of the cases used in teaching are scattered mini programs. Students need to re-understand the functional requirements of the software every time they practice, instead of focusing on the study and comparative analysis of test methods, which weakens the teaching effect. Coherent teaching cases are very important for students to master testing processes and testing technology.

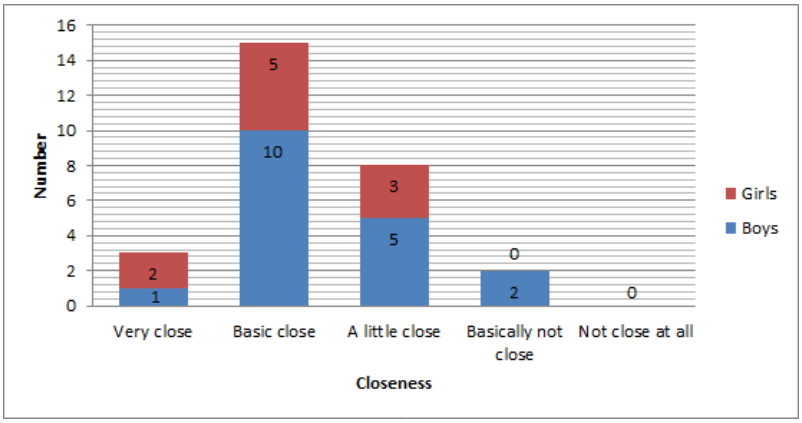

Figure 6. Personal views of male and female students

4. Conclusions and Suggestions

In this paper, we studied the differences between students' testing ability in crowdsourcing projects and tried to explore the potential relationship between the differences of testing ability and students' performance in software testing courses. First, we compared the differences in the ability of undergraduate and junior college students to find bugs in both quantitative and qualitative ways. Then, we focused on assessing the quality of test reports submitted by those students and found differences in the ability of different students to find bugs, write expressions, and judge the effectiveness of bugs. Combining the results of students' software test courses and their feedback (collected through survey), we found that students with better grades generally had some interest in crowdsourced testing, while students with lower grades tended to have a sense of rejection and aversion. This shows that students' understanding of crowdsourced testing is not comprehensive enough. At the same time, students' rich knowledge of testing theory does not indicates a strong testing ability. In other words, students' ability and performance are not equivalent. However, testing theory can provide professional knowledge for testing practice, and testing practice could supply and expand testing theory. This can help students get more exposure to advanced testing technology, which requires colleges and universities to strengthen teaching practice. Some useful suggestions are listed as follows:

- We suggest that the outdated content of the currently used textbooks can be deleted to make the teaching content more refined. We should also make full use of the network resources to find out more about the latest technology, development trends, and even the demand for talent in the industry. These will be used as a basis for adding new knowledge points. In addition, it is necessary to sort out and refine the scientific research practice and teaching experience of software testing as a supplement to the teaching content. Teachers with practical work experience can strongly introduce some auxiliary working methods, tools and related industry knowledge used in practical testing work or projects in order to improve the teaching content and enhance students' interest in learning.
- It is necessary to guide students in testing correctly. Teachers should first guide students to correctly view software testing courses, so as to improve students' enthusiasm to learn software testing courses in a well-rounded manner. Teachers need to help students understand that software testing is the quality assurance of software development, which runs through the whole software development process. Only by ensuring the accuracy of software testing can we ensure that the quality of software development meets the relevant requirements.
- More attention should be paid to practical teaching. For the teaching of software testing courses, it is important to cultivate students' practical ability. Therefore, we must pay more attention to practical teaching in order to improve students' practical testing ability. Equally important, schools should also attach importance in cooperation with software enterprises so that students can graduate, enter society, and apply theory in practice. This can help students find their own shortcomings and lay a good foundation for the comprehensive development of education.
- Finally, schools should increase investments and buy software testing tools that are popular in some small and medium-sized software enterprises to meet the needs of the curriculum in hardware. On this basis, educators can combine real data and software code provided by the enterprise, build a simulation working environment, and decompose the project tasks to complete the reproduction of the task scene in the enterprise. Students can be assigned to groups, assigned roles, and motivated to actively collaborate on project tasks. This kind of practical teaching method not only strengthens the students' practical ability, but also trains and cultivates their communication skills and teamwork ability.

Acknowledgements

This work is partly supported by the National Key Research and Development Program of China (No. 2018YFB1403400) and the National Natural Science Foundation of China (No. 61690201, 61772014). This work was also supported in part by the Science and Technology Project of Guangzhou (No. 201904010220) and by the Project of Technology Development Foundation of Guangdong（No. 706049150203). The authors would like to thank the students of Guangzhou Panyu Polytechnic, who provided assistance during the experimentation process.

References

1. P. C. Jorgensen, "Software Testing: A Craftsman's Approach," Auerbach Publications, 2013
2. W. E. Perry, "Effective Methods for Software Testing: Includes Complete Guidelines, Checklists, and Templates," John Wiley and Sons, 2007
3. D. Liu, M. Lease, R. Kuipers, and R. G. Bias, "Crowdsourcing for Usability Testing," *Journal of Proceedings of the American Society for Information Science and Technology*, Vol. 49, No. 1, pp. 1-10, 2012
4. T. Zhang, J. Gao, and J. Cheng, "Crowdsourced Testing Services for Mobile Apps," in *Proceedings of 2017 IEEE Symposium on Service-Oriented System Engineering* (*SOSE*), 2017
5. L. Riungu-Kalliosaari, O. Taipale, and K. Smolander, "Research Issues for Software Testing in the Cloud," in *Proceedings of 2010 IEEE Second International Conference on Cloud Computing Technology and Science*, 2010
6. Z. Y. Chen and B. Luo, "Quasi-Crowdsourcing Testing for Educational Projects," in *Proceedings of the 36th International Conference on Software Engineering*, 2014
7. M. McMullan, "Students Perceptions on the use of Portfolios in Pre-Registration Nursing Education: A Questionnaire Survey," *International Journal of Nursing Studies*, Vol. 43, No. 3, pp. 333-343, 2006
8. Y. Feng, Z. Chen, J. A. Jones, C. Fang, and B. Xu, "Test Report Prioritization to Assist Crowdsourced Testing," in *Proceedings of the ESEC/SIGSOFT FSE*, 2015
9. B. Kirubakaran and V. Karthikeyani, "Mobile Application Testing—Challenges and Solution Approach Through Automation," in *Proceedings of 2013 International Conference on Pattern Recognition*, *Informatics and Mobile Engineering*, IEEE, 2013
10. J. Wang, S. Wang, Q. Cui, and Q. Wang, "Local-based Active Classification of Test Report to Assist Crowdsourced Testing," in *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, pp. 190-201, 2016