Curriculum Reform on Software Testing Courses to Strengthen Abilities of Engineering Practice

Rui Mao [a], Meili Wang [b], Xiangtao Wang [c] and Yang Zhang [d,*]

College of Information Engineering, Northwest A&F University, Shaanxi 712100, China.

[a] maorui@nwafu.edu.cn, [b] wml@nwafu.edu.cn, [c] wxt@nwafu.edu.cn, [d] zhangyang@nwafu.edu.cn

Abstract

Software testing is the key to ensuring a successful and reliable software product or service. In Chinese market, the demand for software testing engineers exceeds the supply. However, the deficiency of engineering practical ability prevents the access to sites of test requirements for university graduates cultivated in the conventional software testing courses. In order to enhance the ability of engineering practice, we explored a set of reformation schemes in the course teaching of software testing. The reformation schemes include adding automated software testing to enrich teaching contents, implementing the case method of reverse design to improve teaching methods, and proposing a multivariate process assessment to better education learning results.

Keywords

Engineering Practical Ability, Automated Software Testing, the Case Method of Reverse Design.

1. Introduction

As computer software continues to develop and prevail in national economy and social life, users have increasingly focused on high-quality standards of software products. The software testing is a necessary stage in delivering reliable software to users and a critical process to improve and ensure the quality of software product. Therefore, software testing has inhabited a larger share in the software development. According to relevant data, China's software industry provides around 200,000 employment opportunities for testers while the existing testing talents are less than a tenth of demand, leaving a great gap that has become a bottleneck preventing the development of China's software industry[1].

Universities, as the important platform for cultivating software talents, play a critical role to cultivate testing talents that could cater to demand of testing jobs through reforms of software testing curriculum. The conventional software testing curriculum used to pay too much attention to theories of software testing. As a result, teaching methods mainly focus on imparting knowledge with the teacher-dominated direction. Meanwhile, there is only one standard of assessing students' performance, employing theory examinations. The testing talents that are cultivated through attending conventional courses cannot meet the demands of the testing market and not competent to engineering practices. Therefore, our study will explore reforms of the contents of software testing curriculum, teaching methods and assessment system to improve students' ability of engineering practices

2. Reform Explorations of Software Testing Curriculum

2.1 Focus on Abilities of Practice & Combine Automated Software Testing with Manual Software Testing

The software testing curriculum mainly covers the basic theories, methods and technologies of testing. The conventional stage of testing practice is always included in the practice of software development. Firstly, compared with software design and coding, testing has always not been highlighted and the quality of testing cannot be ensured. Secondly, the practice of testing is divorced from testing curriculum and students are unable to effectively operate and enhance their understanding of acquired knowledge and skills. Practice test theories and spark students' aspiration for theories[2]. On the basis of that, in order to improve students' performances in engineering practices, the curriculum design of dividing practice and theory teaching would be changed, theory study and practice be combined, objects and methods of testing be chosen independently, some testing tools be used and testing results be confirmed through designing testing cases.

Automated testing is a process of transforming man-driven testing into a machine-operated one, which can be seen as a necessary supplement to manual testing[3]. It can expand the coverage rate of testing and reduce the time that it takes to finish repeated manual testing. Automated software testing is especially necessary when high-quality software products need to be delivered quickly[4]. However, conventional software testing curriculum is shortage of the systematic contents related to theories, methods and tools of automated testing, which causes students to design testing plans using a one-sided way from the manual testing perspective and leads to limits for the testing of multi-user concurrency and system reliability. Meanwhile, that method cannot ensure the efficiency of testing. Therefore, we should add some theories and methods of the automated testing, such as the code analysis, capture and playback, automation script and comparison, on the basis of which the automated testing and management tools, like Pc-Lint, Junit, Eclemma, Ant, QTP, LoadRunner, are combined. Then some practices, such as static code detection, unit test, function test and performance test, are designed to develop students' abilities of testing practices comprehensively in order to meet the demand of testing skills in software industry.

2.2 The Case Method of Reverse Design

The software testing curriculum is strongly practical. Bald explaining and imparting testing theories, methods and tools would curb students' initiative and turn them into silent learners, cause adverse impacts on the results of teaching. "Teachers are playing a much smaller role in imparting knowledge while larger role in encouraging thinking. Apart from their formal responsibilities, they would be a counselor, a participant in exchanging ideas and an assistant in finding contradictory points but not a provider presenting existing axioms." UNESCO has already pointed out in 1990s[5]. At the moment, there are various methods and tools of testing, which are closely related with development language, application and testing contents[6]. If the teaching method of "theories before practices" is strictly applied, students would loathe learning software testing due to the abstractness of methods and tools. However, if curriculum is designed reversely, practices before theories and study objectives before teaching practices, students could independently choose testing knowledge and skills that they need through network platforms on the basis of knowledge background and features of software. As a result, study objectives can be clear and students can be driven to study testing knowledge.

The reverse design mentioned above can encourage students to comprehensively take the fact into consideration, to learn different test schemes and methods based on different technology framework of software development. Take the unit test of the software as an example. When students are doing the unit test of the Java environment, they choose Junit while doing that of C/C++ language development, they choose C++Test. In this way, when students complete the practice, they can utilize open cooperative dialogue, which means that "the case method" approach could enable students to further the theoretical study. In the course of practice, according to different test schemes and methods obtained, students are free to form case groups, through cooperative discussions to find common problems and the views on the problems. At this point, teachers can not only participate in the discussions as members, but also as service-providers, providing a strong guarantee for teachers and students to start a dialogue with different views[7]. Through " the case method of reverse design" approach, teachers can be helped to guide students to choose test objects out of their interests, find problems, learn from information, explore individual ways to resolve the problems, and realize the dynamic combination of theories and practice.

2.3 Diversification of Assessment

The examination of a single form is mostly utilized in conventional assessment method and it lays more stress on theory and memory ability, while ignores the assessment of student’s ability of cognition and practice over their study period[8]. In an effort to objectively evaluate the comprehensive quality of students' cognition and practical ability in an all-round way, it is one of the contents of the reform for software testing course to explore the diversification of process assessment. Software testing course is very practical. In the reform of teaching methods, we explored "the case method of reverse design" approach, increased the assessment of their learning effects accordingly. In the learning process of "the case method of reverse design", students can select the learning contents and teams of case project according to their own situations, and they can deepen understanding and improve research ability through cooperation. Therefore, the assessment mainly depends on the completion of learning objectives, attendance in group discussions, preparation for test cases, and participation in problems feedback. The students will be scored by team leaders and teachers, and the results will be released right after the teaching classes and questioned by students, so as to make the next step work better. Combined with the conventional final examinations and practice reports, the assessment of "the case method of reverse design " can consolidate students' understanding of the basic theory, improve students' engineering and practical abilities, and help students cooperate better.

3. Conclusion

Software testing is an emerging industry and many businesses are in need of it, which brings a promising future to software engineering graduates. Software testing course reform has experienced a six-year-long exploration in our school, and made some progress in breaking the conventional software testing course that has too many theories and single way of teaching and assessing. In the content, the manual test method will be combined with the automated one to pay attention to engineering practical ability; in the teaching, we utilize "the case method of reverse design" approach; and in the evaluation, we enhance the diversification process assessment. These course reform methods help us to improve problem-solving abilities of using methods, techniques and tools of software testing for students. However, to cultivate the qualified talents of software test meeting the needs of the software industry, it is still a long-term task of software testing course reform and development.

Acknowledgements

This work was supported by

1. The Fundamental Research Funds for the Central Universities (No. 2452016080).
2. The Program of Northwest A&F University to Sponsor Key Graduate Subject Construction.

(No. ALK201520).

References

[1] Information on http://www.chinajob.gov.cn/.
[2] Coburn, Cynthia E., and William R. Penuel. "Research–practice partnerships in education: Outcomes, dynamics, and open questions." Educational Researcher 45.1 (2016): 48-54.
[3] Garousi, Vahid, and Mika V. Mäntylä. "When and what to automate in software testing? A multi-vocal literature review." Information and Software Technology 76 (2016): 92-117.
[4] Böhme, Marcel, and Soumya Paul. "A probabilistic analysis of the efficiency of automated software testing." IEEE Transactions on Software Engineering 42.4 (2016): 345-360.
[5] McGreal R, Kinuthia W, Marshall S, et al. Open educational resources: Innovation, research and practice[M]. Commonwealth of Learning (COL), 2013.
[6] Jan, Syed Roohullah, et al. "An Innovative Approach to Investigate Various Software Testing Techniques and Strategies." International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Print ISSN (2016): 2395-1990.

[7] Yuan, Li, Stephen Powell, and JISC CETIS. "MOOCs and open education: Implications for higher education." (2013).

[8] Sadler, D. Royce. "Three in-course assessment reforms to improve higher education learning outcomes." Assessment & Evaluation in Higher Education 41.7 (2016): 1081-1099.