Combining Research and Education of Software Testing: A Preliminary Study

Zhenyu Chen
Software Institute
Nanjing University
Nanjing, China
zychen@software.nju.edu.cn

Atif Memon
Dept. of Computer Science
University of Maryland
College Park, MD, USA
atif@cs.umd.edu

Bin Luo
Software Institute
Nanjing University
Nanjing, China
luobin@software.nju.edu.cn

ABSTRACT
This paper reports a preliminary study on combining research and education of software testing. We introduce some industrial-strength programs, from the open-source projects for research, into the assignments of system testing. Research assistants and teaching assistants work together to establish and evaluate the assignments of system testing. Our preliminary results show that research and education of software testing can benefit each other in this way.

Keywords
Software testing education, undergraduate teaching, assignment design

1. INTRODUCTION
In software testing education, open-source software is always introduced into educational projects to achieve industrial-strength training [1]. Open-source software always lacks of documents, such that they are difficult to understand, install and use by students. In order to establish suitable projects based on large open-source programs for students, we need a lot of manual labors. On the other hand, software engineering research community pays more and more attention to empirical studies in recent years [2,3]. The empirical study on industrial-strength software is always a non-trivial task, which may require a lot of human interventions and domain knowledge. Moreover, some empirical studies even need several independent teams working on the same task for comparison. This is impossible in industry for expensive human resources [4]. But at universities, it may be feasible if we can find and organize some well-motivated students.

This paper reports a preliminary study on combining research and education of software testing. The initial motivation is to overcome the problems of our research project on software testing. The project requires a large number of test scripts created manually. Another motivation is that the programs used in our previous course of software testing are simple [5]. These toy programs are difficult to stimulate students’ interests and enthusiasms. Therefore, we need some large and complex programs to improve the quality of students’ projects, as well as transfer of learning [6].

2. ASSIGNMENTS OF SYSTEM TESTING
Random Testing (R.T.) and Functional Testing (F.T.) are two widely used methods of system testing. Two assignments, R.T. and F.T., with three open-source programs are designed and distributed to students. In the assignment of R.T., students are required to test Applications Under Test (AUTs) randomly and manually. In the assignment of F.T., students are required to test AUTs based on the functionality sets, which are provided by Teaching Assistants (TAs).

TAs and RAs: Four TAs in this course are familiar with software testing methods and the related tools. Before distributing the assignments, TAs should analyze the applications under test carefully and try some test scripts running on these applications. However, these open-source programs lack of documents. We need to construct testing requirements by ourselves. In order to improve the quality of documents, we ask Research Assistants (RAs) to do this task, because RAs are more familiar with these programs, which have been used in their research projects. A more important reason is that the quality of test scripts are key for their research projects. This also inspires RAs to assist the design and evaluation of assignments.

Assignment Design: Random testing may be the most simple and popular testing method. It is widely used in unit testing and system testing. A new user will operate software systems randomly in practice. The students are unaccustomed with these applications before creating test scripts. To ensure the quality of test scripts, this assignment requires that all GUI widgets of applications under test should be covered as many as possible and students should do it randomly. Through this assignment, students are familiar with QTP, test scripts and applications under test. It is not surprising that random testing is easy to master by students.

Functional testing is also widely used in system testing. Testers always divide functionalities into several sets and then create test scripts to satisfy them. Functional testing is used to show whether the functionalities are fulfilled correctly. For comparison, we need several independent teams working on the same task, because RAs are more familiar with these programs, which have been used in their research projects. A more important reason is that the quality of test scripts are key for their research projects. This also inspires RAs to assist the design and evaluation of assignments.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 2014 ACM 978-1-4503-2469-4/14/03 ...$15.00.
signed a definite and detailed specification to guide all participants to divide each AUT into several functionality sets. Through this assignment, students can master how to design test cases and create test scripts to fulfill functionalities of applications.

**Assignment Evaluation:** There are 234 regular junior students in our teaching in 2011. It is challenging to evaluate thousands of assignments within limited time. Hence, we ask RAs to help automate the evaluation process. It is not difficult to evaluate the assignments on random testing by collecting all executed widgets and calculating the percent of coverage as a score. For the assignments of functional testing, we define a chip of events as evaluation criteria. However, some test scripts may fail in QTP and it will interrupt the automation process. This may involve some human interventions. RAs will improve the automation framework to reduce human interventions. TAs do the evaluation task: run test scripts, collect the data, and analyze the data as scores. Their cooperation of TAs and RAs is the key point to our success.

3. **SUCCESSFUL EXPERIENCES**

**Motivating Undergraduate Students:** One of the main teaching objectives is to let students recognize the advantages and disadvantages of different testing methods in practice. Although we have many discussions on comparing testing methods in our teaching [7], some questions are still remained on comparing random testing and functional testing after they finish the assignments. Some junior students are very interesting in this topic. I encourage them, with the RAs, to set up a study group. The main purpose is to conduct an empirical study to compare random testing and functional testing for GUI applications. The experimental results should be analyzed in detail and some practice guides of these testing strategies are desirable.

Benefiting from our course, students obtain in-depth understanding on these two testing strategies and can skillfully use QTP and test scripts. The test suites and subject programs are available from our course. However, there are still some challenges for a successful study on comparing testing methods. (1) The experiment should be designed carefully, especially for the evaluation metrics. We should write some scripts to repeat our experiments automatically many times. (2) Bugs (Faults) are not available in our course. But bugs are necessary for comparing testing methods. So the information of bug detection should be collected for evaluation. (3) Many related papers should be studied carefully, which is necessary to a research paper.

After discussing with teacher and TAs, the undergraduate students conduct the empirical study and completed a paper [8]. We study three aspects on these two testing strategies: effectiveness, complementarity and the impact of test case length. Some useful observations of the empirical study are: (1) Random testing is more effective in the early stage of testing on small applications and functional testing is more scalable on large applications. (2) Random testing and functional testing exhibit some complementarity in our experiment. (3) Short test cases can reveal some faults more quickly and long test cases can reveal more faults lastingly.

**Industrial Strength Training:** The main purpose of our software testing course is to let students recognize the advantages and disadvantages of various testing methods in practice. It requires that applications under test should be large and complex enough. For example, in the previous section, random testing is more effective at the early stage of testing on small applications and functional testing is more scalable on large applications. Through the training with the industrial-strength software, the students are able to understand the necessity of functional testing, especially when the application under test is large and complex. It also deepens the understanding of random testing and functional testing, such that they can select suitable testing methods for different applications in practice.

For junior students, it is easy to install and use the well-done commercial testing tools, such as QTP. Hence they may ignore the cost of establish testing environments, if the applications under test are small and simple. In the assignments, it is a non-trivial task to establish the open-source programs and the automatic testing framework by the students. Many technical problems should be solved before a set of success test scripts is built. Through the training with industrial-strength software, the students understand that software testing is a labor-sensitive task. It will inspire the students to develop automatic tools to reduce the costs.

An important fact that may be ignored in the traditional software testing course is the challenges of testing evolving software. The students are required to test a fixed version of software. However, the software is always evolving due to the bug fixing, performance tuning and additional functionalities. In our course, the new versions of programs are provided to students after they generate test scripts on the old versions of programs. The test scripts are run on two versions to demonstrate why and how many test scripts become unusable. This will inspire students to think how to maintain test scripts for evolving software. This is a key point to train students’ testing skills, because software is always evolving in practice.

4. **REFERENCES**


http://dx.doi.org/10.1016/j.jss.2013.09.043