Advanced Science and Technology Letters
Vol.63 (NGCIT 2014), pp.147-150
http://dx.doi.org/10.14257/astl.2014.63.31

Test case Expansion Method from Experience-based Techniques to Specification-based Technique

Dea-Kwang Kim, Lee-Sub Lee *

Department of Computer Engineering, Kumoh National Institue of Technology University,
61, Daehak-ro, Gumi-si, Gyeongsangbuk-do, 730-701 South Korea
{aav7co, eesub}@kumoh.ac.kr

Abstract. The verification of mobile application software requires a lot of black -box testing. Frequent release schedule and the lack of testing experts cause many difficulties for applying Specification-based testing. For this reason generally Experience-based testing has been used in practice. Hand-written test scripts are not sufficient for providing enough coverage due to its cost. In this paper, we propose an automatic test case generation method from Experience-based testing. We expect to provide extending test coverage from Experience-based test with less cost.

Keywords: Experience-based testing, Specification-based testing

1 Introduction

GUI is a very important method for interaction method between the user and the system. Because it is the last resort to verify the system under test, GUI testing takes a very important role of acceptance testing. Fig.1 depicts a classification or taxonomy of test design techniques. As shown in Fig.1, Specification-based techniques and Experience-based techniques are mainly used for GUI testing.

Specification-based techniques are methods to generate a set of test cases from specification documents such as a formal requirements specification [1][2]. Experience-based testing method unites less formal testing techniques, but some of them are still very powerful when are used by professionals. These techniques are Checklist-based Testing, Exploratory Testing, Error Guessing, and Ad-hock Testing. [3][4]

The advantage of the Specification-based techniques is that if it is written appropriately, it provides high test coverage. However, Specification-based techniques require high level professionals and a lot of efforts for creating the specifications. Furthermore, considering the competitive market environment, to satisfy a very strict timeline for time to market, developers and tester should choose Experience-based techniques for urgent development schedule. Therefore, in most cases, test cases are manually written based on Experience-based techniques using related tools such as Record/Playback, Data-driven, and Keyword-driven tools.

ISSN: 2287-1233 ASTL

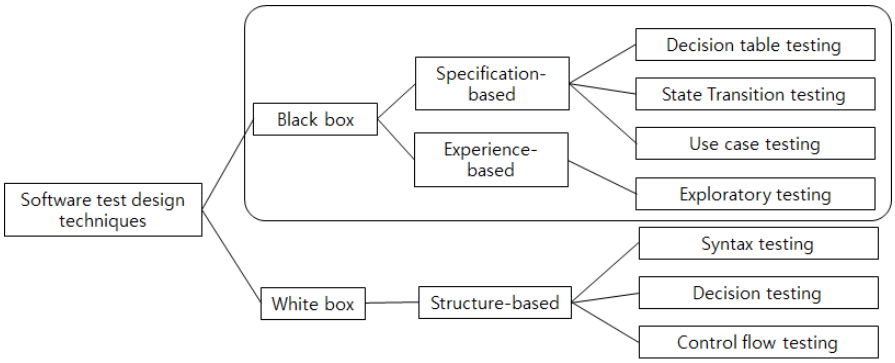


Fig. 1. Classification of Test Design Techniques

However, Experience-based testing could not provide sufficient test coverage. As a result GUI testing cannot provide enough quality of the product. Moreover, it is very labor-intensive work for expanding even a little more coverage. It requires too much cost that development and testing team cannot afford to.

In this paper, to take advantages of Specification-based techniques we proposed a method of applying a reverse engineering concept which automatically generates formal specifications from manually written test cases. The paper also proposed a method to expand uncovered test paths. The generated specification could contribute to enhancing the quality of software with little additional manual tasks.

2 Software testing is applied to reverse engineering

Reverse engineering is the "reverse progression" implementation of forward engineering. In order to understand the activities of reverse engineering, we should first understand forward engineering. Fig. 2 shows the phases of the 'waterfall' model, for forward engineering, which is a typical process used to the construction of legacy systems.[5] Reverse Engineering is focused on the challenging task of understanding legacy program code without having suitable documentation.[6] In this paper, to obtain high coverage with less cost we are trying to apply and enhance existing reverse engineering techniques in the software development lifecycle.

In order to understand the activities of reverse engineering in test process, we should first understand forward engineering in test process. Fig.3 shows forward engineering in model based testing as an example of specification based techniques. [7]

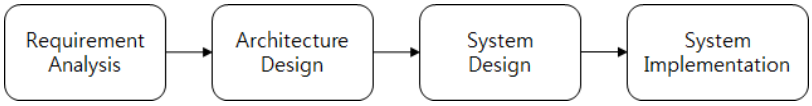
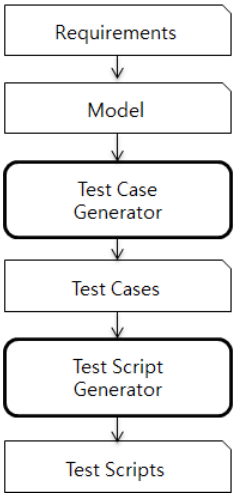


Fig. 2. The Waterfall Model of Forward Engineering



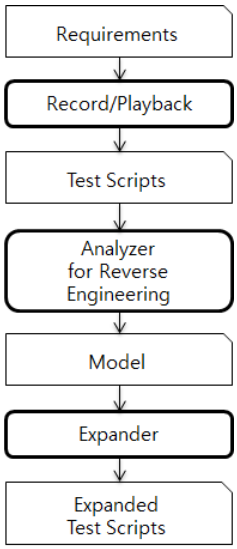


Fig. 3. Model-based Testing Process **Fig. 4.** Testing Process using Reverse Engineering

We propose a suitable reverse engineering process from experience based techniques to Specification-based techniques in fig. 4. The purpose of this process is by reusing the written test cases from Experience-based testing we can easily extract test cases with high coverage.

From requirements testers write test scripts with Record/Playback tools such as Sikuli Tool. [8][9] It is a representational tool for traditional Experience-based technique. From the written test scripts an analyzer for reverse engineering extracts a model or specification that has the form of Test Flow Graph(TFG).[10] We use TFG rather than Extended Finite State Machine (EFSM). Due to the variety form of test scripts, it is very hard to get enough information for analyzing exact and perfect EFSM. This task is also very hard work for even Specification-based testing professionals. During the analysis analyzer identifies state, event, guarding condition and duplicated test paths. With the generated model expander analyze which path is missing in terms of evident testing coverage method. Then it also checks the missing test data according to trivial testing theory such as Boundary testing etc. Even though, it could not generate a perfect formal model. It is very helpful and practical for testers to check whether testers miss evident test paths and related data.

3 Conclusion

Although Specification-based Testing could generate high quality of test cases, in most cases due to its heavy cost it is not suitable method in industry domain practically. The most popular adapted method is Experience-based Testing. Even though in the early stage of the Experience-based testing the cost is relatively low. As

the verification process continues even for a little higher coverage, it requires very high labor cost.

To solve the problem, this paper suggests a modified version of reverse engineering for figuring out specification model. With this method the cost of verification could be reduced without high labor cost.

The future work will include more detail algorithm for unifying related test paths and extracting meaningful test data. The work will also include more enhanced expanding method for various test paths and data.

References

1. Nicha Kosindrdecha and Jirapun Daengdej, "A Test Case Generation Process and Technique," Journal of Software Engineering, vol.4, no.2, pp. 265-287, 2010.
2. J. Srinivasan, and N.Leveson, "Automated Testing from Specifications", Digital Avionics Systems Conference 2002. Proceedings. vol. 1, pp. 6A2-1-6A2-8, 2002.
3. Nicha Kosindrdecha and Jirapun Daengdej, "A Test Generation Method Based on State Diagram," Journal of Theoretical and Applied Information Technology, 2005~2010. pp. 28-44.
4. J. Edvardsson, "A Survey on Automatic Test Data Generation," In Proceedings of the 2nd Conference on Computer Science and Engineering, Linkoping, pp. 21–28, October 1999.
5. Chih-Wei Lu, William C. Chu, Chih-Hung Chang, Yeh-Ching Chung, Xiaodong Liu, and Hongji Yang, "Reverse Engineering," Handbook of Software Engineering and Knowledge Engineering, Vol. 2
6. Ramandeep Singh, "A Review of Reverse Engineering Theories and Tools", International Journal of Engineering and Science Invention, vol. 2 no. 1, pp. 35-38., January 2013
7. M. Utting and B. Legeard, "Practical Model-Based Testing - A Tools Approach," Morgan and Kaufmann, 2006
8. T. Yeh, T.-H. Chang, and R. C. Miller. Sikuli: Using GUI screenshots for search and automation," In UIST ’09, pages 183–192. ACM, 2009.
9. T.-H. Chang, T. Yeh, and R.C. Miller, "GUI Testing User Computer Vision," CHI, 1535-1544, 2010.
10. Ryan Voigt, Kareem Fazal, and Hassan Reza, "Specification-based Testing Method Using Test Flow Graphs," ICSEA 2007. pp. 48-58, 25-31, Auguest 2007.

