Smart Innovation, Systems and Technologies 79

Arun K. Somani
Sumit Srivastava
Ankit Mundra
Sanyog Rawat *Editors*

Proceedings of First International Conference on Smart System, Innovations and Computing

SSIC 2017, Jaipur, India

KES International

Springer

User Feedback Based Test Suite Management: A Research Framework

Akhil Pillai and Varun Gupta

Abstract Delivering better product value on each revision is the principal task for an organization's existence in a competitive market. The customer's base needs to be continuously increased to earn more revenue. Feedback is collected from customers to decide functionalities that need to be included in the future versions of the software. A software company gets ooded with feedback, which makes it dif cult to analyze, process, and prioritize. Feedback may result in addition of new code in order to implement new functionality or to delete/modify existing one. Each task which alters the source code to implement a feedback requires modifying test suites which enhance the testing effort thereby impacting project schedules, budget, and plans. In this paper, we highlight a research framework that attempts to lower the testing effort by reducing the number of test cases by compounding the feedback which is similar to existing requirements of same or similar projects.

Keywords Test cases Customer feedback Requirements

1 Introduction

A software company delivers software incrementally which gives its users enough time to use a particular increment and record their expectations as feedback, which shapes the next increment. This feedback oriented development is followed till the software is taken off the market. Thus, feedback is used by rms to make decisions about the functionality of next increment and various improvements/modi cation.

Each requirement is mapped to certain test case/s, which are always increasing. In other words, requirements expressed implicitly or explicitly through user feedback increases the size of test suite. Feedback is an integral part of the development

A. Pillai) V. Gupta
Amity School of Engineering and Technology, Amity University, Noida, Uttar Pradesh, India
e-mail: akhilpillai18@gmail.com

V. Gupta
e-mail: vgupta7@amity.edu

A.K. Somani et al. eds.), *Proceedings of First International Conference on Smart System Innovations and Computing*, Smart Innovation, Systems and Technologies 79, https://doi.org/10.1007/978-981-10-5828-8_78

of evolutionary software, it also aims at reducing testing effort as it is one of the objectives of a software rm.

This paper highlights the motivation and rationale behind the research, i.e., focusing on building a test suite on the basis of user feedback for the current version. Once some feedback is obtained from customers, it is analyzed to check whether it requests for new requirement/s in which case new test case/s need to be designed or whether it requests for deletion of feature/s, accordingly test case/s are required to be removed completely. Feedback received can also indicate modi - cation of an existing requirement, thus either a change is made in the existing test cases or new test cases are created. Minimization of the test suite is challenging in mass market development, because in such development the number of requirements feedbacks) is always increasing [1].

2 Research Framework

Let R_i be a requirement in set R with T_i, T_j test cases associated with it. Let F_i be feedback relaying either new functionality, deletion request or modi cation request. F_i is associated with a single requirement, R_{Fi} and T_{Fi}, T_{Fj}, …. T_{Fn} test cases. For new functionality, the test cases associated with F_i say T_f prompts an increase in the size of the test suite. Whereas for deletion, the test cases associated with F_i say T_f will decrease the size of test suite by maximum value of N where N is the number of test cases associated with feedback F_i), it is important to note that not all test cases associated with feedback F_i be removed, since some test cases may be required to be repeated as they might be associated with requirements from set R. While for modi cation, the cardinality of test suite will increase since few test cases will be modi ed and few more might be added Fig. 1).

For the rst increment, effort is given by

$$E_1 = E_{R1} + E_{R2} + \cdots + E_{Rn} \quad (1$$

For the second increment, effort is given by feedback

$$E_2 = E_{regression} + E_{Rn1} + \cdots + E_{Rmod1} + E_{Rmod2} + \cdots + E_{Rmodn} \quad (2$$

Now $E_2 > E_1$, E_2 is too high.

The objective of the research is to handle feedback in such a manner that E_2 is minimized. This is possible if we can consider high priority feedbacks and attempt to minimize the number of test cases associated with these feedbacks.

Thus,

E is directly proportional to $E_{Feedback}$
E is directly proportional to $E_{Regression\ Testing}$
$E_{Feedback}$ is directly proportional to the number of feedback
The number of test cases is directly proportional to the number of feedback.

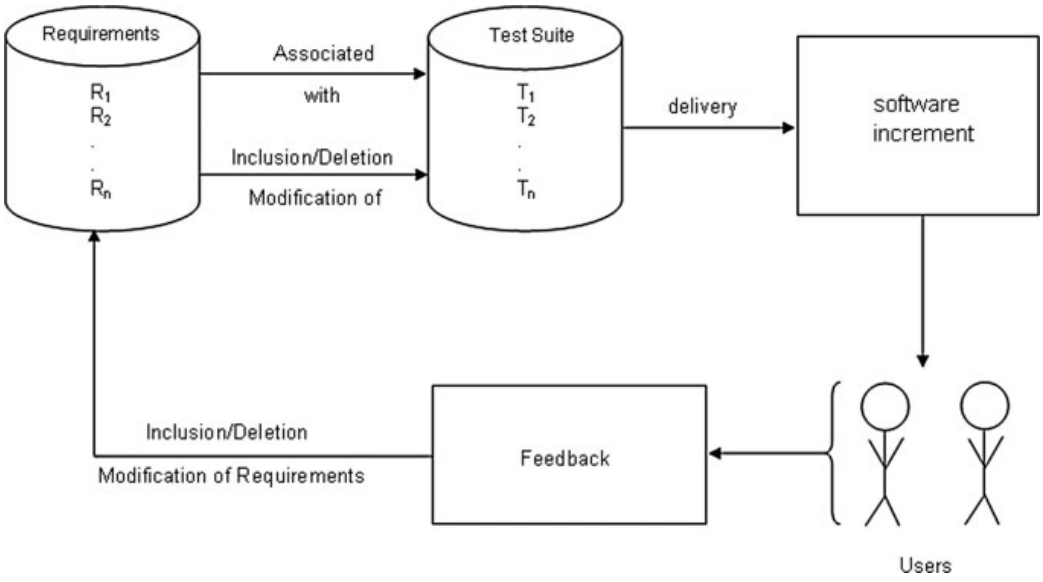

Fig. 1 Suggested framework for user feedback based software testing

So by reducing the number of feedback, the number of test cases are reduced, which in turn reduces the effort E) required. Reduction of feedback is possible by careful prioritization of feedback and reduction of test cases by analysis of the similarity of new feedback with existing requirement of the same project.

2 1 Impact of Feedback

During the initial phase of a development cycle, test cases are created on the basis of requirements. Each requirement or a group of requirements are referred to create a test case. After the rst iterative release of the software product, feedback is collected from customers. It is important to note that the feedback received from the customer might contradict an existing requirement, complement an existing requirement or have no effect on the existing requirement. The nature of feedback is pivotal in ensuring elimination of redundancy in test cases as well as minimization of test cases [2–4].

The feedback received from a customer might request a new feature, change in some existing feature or even removal of a feature. This nature of feedback has an impact on the existing set of requirements. This impact is to be measured before creating a new set of test cases based on the feedback and starting the development process [5].

If the nature of feedback contradicts the existing set of requirements, then it has the highest impact since it indicates a new set of requirements which would require the creation of completely new test cases. For instance, the username eld in a login webpage might initially allow numeric characters but after the rst increment is

Table 1 Measuring impact of feedback on existing requirement

	R_1	R_2	R_n
F_1	0	2	1
…	.	.	.
Fn	2	1	0

delivered the feedback collected from the client might specify for alphanumeric characters in the username eld.

If the nature of feedback complements the existing set of requirements, it has a moderate impact on the existing set of requirements since the previously created test cases can be used or modi ed. For instance, the username eld in a login web page might initially allow a length of 4 numeric characters but after the rst increment is delivered the feedback collected from the client might specify to allow a length of 32 numeric characters in the username eld. The least impact is caused by the feedback which is similar to the existing set of requirements, in which case we only have to reuse the existing set of test cases.

The impact values of the feedback are numerically graded, least impact as 0, moderate impact as 1 and one with the highest impact as 2. Table 1 can be created where columns correspond to requirements and rows correspond to feedback obtained from customers.

3 Proposed Solution

In this section, we propose an algorithm that attempts to reduce the number of test cases on the basis of user feedback. The feedback generated is input to the algorithm. Feedbacks are generated in natural language. It is dif cult to process natural language, since the processing and conversion of natural language is out of scope of the presented research. It is assumed that the feedback provided will be in a structured manner such that it matches the input conditions of the algorithm. Feedback is generally about requirements the user wants to add, remove or edit. So the feedback should be represented as requirements are represented.

3 1 Proposed Algorithm

Step 1. Procure feedback from the team, and process and format the feedback similar to that of the requirement form.
Step 2. Choose one feedback from the list and compare it with the existing requirements.
Step 3. Determine whether this feedback is contradictory to the original requirements and assign a maximum weight 02) per each of the requirement it contradicts, sum all the weights $W_{contradictory}$.

Let us suppose that feedback F_1 contradicts with R_1 to R_N.

$$W_{contradictory} = \sum N*2 \quad (3$$

Step 4. Determine whether this feedback supplements the existing requirements and assign moderate weight 01) to it, sum all the weights $W_{supplement}$. Let us suppose that feedback F_1 supplements with R_1 to R_M.

$$W_{supplement} = \sum M*1. \quad (4$$

Step 5. Determine the similarity of the feedback to the existing requirements and assign minimum weight 0), if it is almost similar to the existing feedback. Sum all the weights, $W_{similar}$.
Step 6. Find the net weight of the feedback using Eq. 5)

$$W_{net} = W_{contradictory} - W_{supplement} \quad (5$$

Step 7. Repeat from Step 2 to step 6 for all feedbacks.
Step 8. The feedback with the maximum net weight W_{net}) is considered of high priority and all test cases new or old) are considered of high priority. However, these test cases could also be further prioritized.

4 Conclusion and Future Work

The feedback given by users may cause an increase in the number of test cases in the test suite, which ultimately increases the total effort. If the feedback is analyzed and prioritized, it can be linked with the existing test cases which enable a rm to alter cardinality of the test suite.

In future, techniques addressing feedback oriented test suite management, i.e., based on the presented research framework will help a software company to test complete software with minimal effort. Analysis of requirements stated through customer feedback, will be used for reduction of the test suite. The proposed algorithm needs to be validated on a live case study. The number of the test cases associated with high priority feedback could also be further reduced, which is considered as future work.

References

1. Karlsson, L., Dahlstedt, Å. and Dag, J.: Challenges in market-driven requirements engineering–an industrial interview study. In: Eighth International Workshop on Requirements Engineering: Foundation for Software Quality 2002)

2. Karambir and Kuldeep Kaur: Survey of Software Test Case Generation Techniques. In: International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 6, June 2013)
3. Kohsuke Yatoh, Kazunori Sakamoto, Fuyuki Ishikawa, Shinichi Honiden: Feedback-Controlled Random Test Generation. In: Proceedings of the International Symposium on Software Testing and Analysis, Pages 316–326, 2015)
4. Carlos Pacheco, Shuvendu K. Lahiri, Michael D. Ernst, and Thomas Ball: Feedback directed Random Test Generation. In: Proceedings of the 29th International Conference on Software Engineering, ICSE'07, pages 75–84 2007)
5. Helena Holmström Olsson and Jan Bosch From Requirements To Continuous Re-prioritization Of Hypotheses. In: International Workshop on Continuous Software Evolution and Delivery, pages 63–69 2016)