Quest for Research Excellence On Computing, Mathematics and Statistics

Editors
Kor Liew Kee
Kamarul Ariffin Mansor
Asmahani Nayan
Shahida Farhan Zakaria
Zanariah Idrus

Faculty of Computer and Mathematical Sciences

Quest for Research Excellence on Computing, Mathematics and Statistics

Chapters in Book

The 2nd International Conference on Computing, Mathematics and Statistics (iCMS2015)

Editors:

Kor Liew Lee
Kamarul Ariffin Mansor
Asmahani Nayan
Shahida Farhan Zakaria
Zanariah Idrus

Quest for Research Excellence on Computing, Mathematics and Statistics

Chapters in Book

The 2nd International Conference on Computing, Mathematics and Statistics

(iCMS2015)

4-5 November 2015
Langkawi Lagoon Resort
Langkawi Island, Kedah
Malaysia

Publication by
Faculty of Computer & Mathematical Sciences
UiTM Kedah

ISBN 978-967-0314-26-6

Content

CHAPTER 2
Analyses of Software Testing Problems in Small and Medium Software Enterprises (SME's) and a Proposed Framework on Exploratory Testing

Murugan Thangiah and Shuib Basri

Abstract. In the field of IT, the software industry recognizes the value of the small and medium software (SME's) enterprises in contributing valuable products and services to the economy. In SME's the quality of the software is increasingly become a concerned subject. The suitable use of testing standards was not written for the development organizations having less than 25 employees. Currently, most of the SME's do not follow any standards and do not adopt any well structured testing methods. In this research paper, the challenges faced by the SME's towards software testing in order to produce quality products are being addressed. Various testing processes are analysed and discussed in the literature review; to identify which methods are best suitable for the SMEs and to come out with a research question. Further study on the research question will lead to a solution by proposing an acceptable framework of the testing process called Exploratory Testing.

Keywords: Exploratory Testing, Framework, SME's, Software Quality.

Murugan Thangiah (✉) • Shuib Basri
Department of Computer and Information Science,
Universiti Teknolgi PETRONAS, Tronoh, Perak, Malaysia
e-mail: tm_gun@hotmail.com, shuib_basri@petronas.com.my

1 Introduction

Software applications are widely used in complex environment ranging from single user applications to multi-client server applications running on different platforms. Applications are written using different programming languages with variety of databases and data structure [1]. All these components are connected with different networks. When a software product is used and implemented in any applications, the developer must ensure that the product is free from errors (bug), and it should work without any interruptions under various situations. One of the major activities in software testing is to identify bugs [18]. Most of the larger organizations used more than 40% of the project development time in such activity.

At present, the majority of the software companies are small and medium sized companies with 10-50 employees. The small and medium size enterprises face many challenges to develop the quality software to a reasonable and acceptable standard. The other major challenges that is quite common in small organizations such as project management, providing training to a new employee, conducting peer reviews and measurement [2, 3]. Currently, testing activities in SME's are done mainly without any proper structure [2]. After writing a piece of code, the developer tests the module within themselves. When all the modules are completed, one of the developers will integrate those modules. Then the software is being tested by a second person, probably the head of the project team, however, this is not enforced by any procedure [3]. Incidentally, there are no testing scripts made available and no testware is developed, because most of the SMEs do not apply any formal testing techniques. At the same time, the test data and testing methods are never documented.

This research work is to address the various issues surfaced in the SME's related to testing. Then to critically assess the software testing process implemented in SME's, then to propose a framework on improved testing process based on the software engineering literature, as well as the insight knowledge gained from this study.

2 Current Challenges in SME's

Across the world, majority of the software industries are comprises of SME's. In order to sustain and to survive in the highly competitive marketplace, these small and medium size software companies must produce high quality software that will ensure sustainable business model. SME's face many challenges along the way to create quality software in order to maintain the survival in the market place [4]. Firstly, they do not believe in the process and methods followed by the larger software companies. Secondly, their organizational structures and processes are quite informal and it leads to a

chaotic environment. Thirdly, the challenges are lacking of resources, experiences, skills, cannot afford to employ experienced software developers and the cost involved towards testing [5, 6].

Although several test case methodologies existed, all those existing testing methods are unable to detect the bug when the product is completed. There are situations where some bugs are identified by the customers themselves [7] and this is quite common when a product is developed and implemented by SME's. To understand on where to focus on the improvement efforts, a starting place for a test system team should start with the awareness of the different types of problems that can escape during the development of the life cycle due to the challenges in software testing. The real fact is "Complete testing is impossible" [8].

No structured or standard approach for testing

There are no proper procedures or methods adopted for the software testing. Since the SME's main objective is to satisfy its stakeholders [3], this constraint resulted in giving too little opportunity for the developers to test the task. Besides that, last minute addition of functionality can also harm the development process. This factor causes additional obstacles which may either result in the delay of the product delivery or the product has many errors [2].

Project delay

There is always a delay in starting the projects. The change of the requirements request by the stakeholder is one of the reasons of the delayed. The stakeholder does not know exactly what they need initially [4]. User can propose requirement changes at any stage during System Development Lifecycle (SDLC) process. The worst part is, sometimes the stakeholders can see what can be done with the application only after the application is completed and ready to be used. SME's organizational structures and processes are quite informal and it leads to a chaotic environment which may also causes the delay of the project.

Lack of objectives and overviews among the developers

The importance of software testing is increasing due to the increasing complexity and proliferation of the software. During the development process, the overview of the project is missing within the team members since the manager will explain only certain parts of the module to each team members. This will lead to lack of awareness on the objectives and overview of the

ongoing project. The managers and developers are standing too far apart from each other.

There are number of reasons on why software testing is required:-

- To identify the bugs,
- To measure quality,
- To reduce / minimize the risks,
- To ensure the product serves its purpose,
- To provide information to the stakeholders.

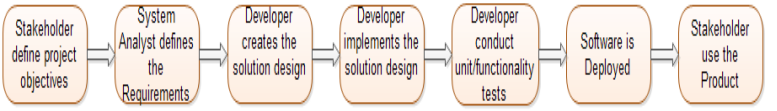

Fig. 1. Current Workflow in SME's

The above figure shows the basic workflow currently used in most of the SME's.

According to the Fig.1, many SME's do not follow any of the standard testing methods or procedures. They even do not have any separate Quality Assurance (QA) team to do the testing. Besides that, the SME's do not have proper documentation methods to record their testing process. This would result in repetition of the same testing process for much functionality which is considered as time consuming process. In addition to that, it is not affordable to use any commercial automated testing tools to test the system. Although, automated testing is recommended for software testing in SME's, but it did mention that only organizations with stable test processes should embark on an automation strategy [4].

3. Literature Review

Software testing is a process of finding the defect (bug) in a product to ensure that the intended product works well within its scope and boundary and also to meet the stakeholder's requirement [9]. In other words, software testing is a process of checking that the software product behaves well and provides

information to the stakeholders about the quality of the product. Hence, it is essential to have good quality software.

Complexities in Testing Methods

Andrew Austin and Laurie Williams emphasized on the importance of discovering the vulnerabilities in the software [10]. The reason is security vulnerability is very expensive if it was not discovered during the development life cycle [11]. According to Gary McGraw, he wrote in his book, *Software Security: Building Security In*, claimed that "Security problem evolve, grow, and mutate, just like species on a continent. No one technique or set of rules will ever perfectly detect all security vulnerabilities" [12]. Therefore, software developers should strive to discover the vulnerabilities as soon as possible.

Rodrigo Souza and Christina Chavez have done an experiment to characterize the verification of bugs found in two open sources Integrated Development Environment (IDE's) using the data available from bug the repositories. However, those bug reports are found to be inaccurate, inconsistency, misleading and ineffective [13]. It also affects the quality of the software products. It is found that the verification of the code was done through static analysis tool. This tool is not the favorable testing methods for any software projects.

Sarah Al-Azzani and Rami Bahsoon have presented a novel on the approach of using implied scenario detection for security testing with LTSA tool to test the system. In their model, they managed to detect few security threats which are closely related to one another. However, it is not clear whether this tool will detect all the security threats due to the dynamic of nature in any attacks [14]. The limitations imposed on this approach are the scalability of the LTSA tool used to detect implied scenarios.

Samer Al-Zain et al. used an automation tool called Test Complete, to conduct testing for web application [15]. This tool helps to create, manage and run automated tests for any Windows, Web or Rich Client software. Using this tool, test engineers can perform several types of automated tests~~,~~ such as functional Graphical User Interface (GUI) tests, regression tests, load and stress tests, unit tests and many more. However, the test recorder tool shows some weaknesses when using GUI test recording~~,~~ for dynamic web applications. Besides, the tester must have the knowledge and must provide the algorithm for writing robust test and successful test scripts.

Vesa Kettunen et al. have studied the differences of testing activities between software organizations which apply to the agile development methods and organizations which take~~s~~ the traditional plan-driven approach by using qualitative grounded theory method [16]. Agile methods can improve the testing position through early involvement of testing activities in

the development. It also has a positive influence on end-product satisfaction. However, most of the agile methods do not give enough guidance on how testing should be aligned parallel with development work.

Haddad et al. explored the challenges and benefits of using software metrics in small organizations. They outline a practical framework based on the Goal / Question / Metric paradigm for instituting metrics programs in small software organizations [17]. During the process, project, and product metrics shared a common goal of contributing to software quality and reliability, nevertheless, the utilization of these metrics has been minimal or not adopted at all by majority of the SME's. This is because the managers and developers understand the high-level goals of improvement on software process and metrics programs, but it is often unwilling to expand the necessary effort for various reasons [17]. Effective communication between the managers and developers is the only way to have a successful software process improvement and metrics adoption for SME's.

Ghazia Zaineb and Irfan Anjum Manarvi in their studies explored that a number of software testing matrixes are based on the total number of defects / bugs are reported therefore; it is a common practice in developing software matrixes to eliminate the number of rejected bugs from the count in order to keep the project performances realistic [18]. The areas that have major problem causing bug rejections are bug reports and insufficient knowledge of testers over the developed software. They concluded that instead of wasting time over the reviews and rework on it, it would be better to provide training to the test team before deploying the test release.

Beer and Ramler studied the role of experience in the development of test cases; regression testing and automation testing in three case studies. They found that the domain knowledge, as against the testing knowledge, is more crucial in testing [22]. Kettunen et al. (2010) also reported in their studies that the domain knowledge to be the most important area of testers' expertize and further emphasized the importance of the role of technical knowledge, especially in the agile development context [16].

Prakash and Gopalakrishnan emphasized on the contrasting approach in the scripted testing vs. exploratory testing and revealed that latter approach was much more helpful to find some important bugs very quickly. This clearly indicates that in the scripted testing the amount of time taken to detect a bug is longer than that in the exploratory testing [19]. By giving freedom to the tester resulted in a quick identification because the bugs were spreading across the entire application.

Juha Itkonen, Mika V. Mantyla and Casper Laasenius have conducted an experiment in comparing the fault detection using test case based testing and exploratory testing [20]. In their experiment, they have found that Test Case based Testing (TCT) provides more false positive reports than exploratory

testing. It was because test case based testing techniques are based on theories and it does not help to find the bugs in all the software products.

Juha Itkonen, Mika V. Mäntylä and Casper Lassenius also conducted a study that supports the hypothesis that the testers, in practice, apply numerous techniques and strategies during test execution. It does not mechanically rely on test documentation [21]. Furthermore, they identified that execution-time techniques are partly similar to test case design techniques, but are strongly experience-based and applied in the non-systematic fashion during test execution. However, currently there are no models, methods or standards that exist to carry out the test execution.

Many different techniques, tools and automation strategies have been developed to make testing more efficient and effective. Despite the wide variety of proposed solutions, the fundamental challenges of software testing revealed the new defects in newly developed software or after major modifications in the existing software. It is still in practice largely dependent on the performance of human testers manually.

Furthermore, the existing research on the role of experience in software testing raises the hypothesis that experience and domain knowledge of the tester are equally important rather than depending on scripted testing or test automation entirely. However, there are no empirical studies exist so far.

Why Exploratory Testing?

The two main thrusts of testing are firstly based on the composition of the software, i.e. its internal structure. Secondly, based on the business or intended purpose of the software, i.e. the functional aspect of the software [23].

Based on the studies, it is evident that though many testing techniques and methods are existed, none of them can guarantee that the product is free from error, especially in the SME's when there are no standard practices adopted for testing procedure. And most of the testing is done by the developer during the development stage itself. From the literature, it is obvious that neither the testing methodology nor the tools adopted by the small software company will be able to identify the defect. Developers in the SME's write unit tests for their code because of the agile technology in the software industries.

In many software development contexts, the manual testing effort of professional testers and application domain experts are crucial to ensure that the products fulfill the needs of the users and the stakeholders. In this context, exploratory software testing has been proposed as an effective testing approach. The exploratory testing (ET) approaches differ significantly from the traditional software testing methods. It is not based on predesigned test

cases. It is creative and experience-based approach in which test design, execution, and learning are parallel activities [24]. The results of executed tests are immediately applied for designing further tests. This type of testing is best suited for agile development process especially in SME's industries.

4. Research Focus

The main goal of this research is to propose a framework for Exploratory Testing in small and medium size software enterprises. There are many types of testings exists and each testing methods has its own significance and weakness. In Agile development process, the testing activities become more complicated as there will be frequent changes in the requirement from the stakeholder, which in turn the developer has to change the code and the testing strategies must repeat again. This will lead to delay in the completion of the project or it undergoes poor testing which results in the project delivered with bugs. In SME's, especially in the unstructured organisation, when there is no proper documentation and guidelines for testing is exits, then chances of having bugs in the product is inevitable due the frequent changes in the requirement.

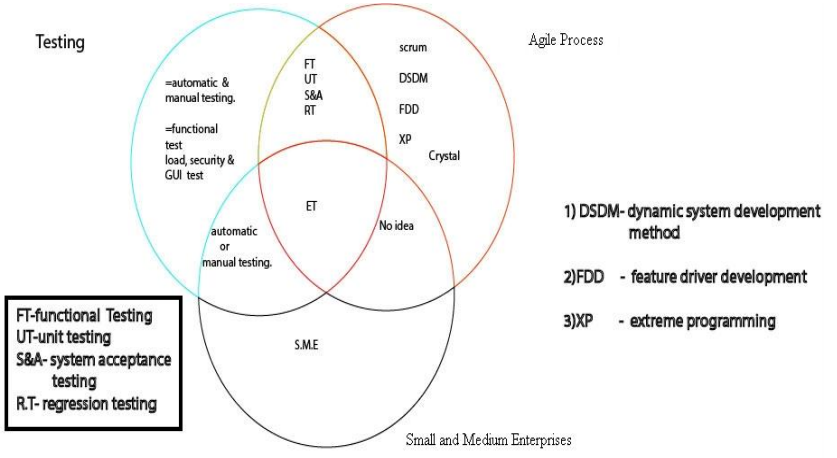

Fig. 2. Research Focus

The focus of the research is towards how testing is done in SME's using Agile development process and how to integrate Exploratory Testing in

SME's to ensure that the product is deleivered with free from errors. In fig. 2, various testing methods and agile processes are highlighted. In the agile development process, only few type of testing methods like regression testing, unit testing and functional testings are best suitable.

Also the testing methods used by SME's are either they use automated testing – using open source tool - or manual testing without having any test cases or test documentation. Mostly the testing is done by the developer and there may not be any separate testing (Quality Assurance) team to conduct the test. Hence, the new testing approach called Exploratpory testing is proposed with the aim to produce quality software by integrating all the three domains – Testing, Agile process and SME's.

Exploratory Testing Framework

Definition: *A framework is a set of rules, goals, facts or assumptions that provide support to achieve something (or) a set of ideas, principles, agreements, or rules that provides the basis or outline for something intended to be more fully developed at a later stage.*

The fig. 3 is the visual representation of the proposed framework to plan and control the testing processes in general and it may be adopted with some little modifications, if successful, in SME's. It also helps to find bugs with the simplest sequence of events, making diagnosis and bug advocacy easier.

Input combinations and constraints are the dominant factor for any testing methods. In this case, it may include data validation, verification, boundary value analysis etc., and other input combinations are also taken into consideration. The next step can be the data flow and control flow aiming to test the business logic in a prescribed manner. This helps to identify all the logical paths that are working as expected which the system aimed to serves its purpose.

Performance and Efficiency tests are aimed at where all the data goes through the system pushing to the maximum possible amount through each field and look for truncation or other forms of interruption. Various types of testings are stress testing, load testing, sanity testing and many others are available. It all depends upon the product element and the product environment. The testers may choose any one of them.

Scenario based testing, individual functions are tested one by one that replicate the behavior and execute large number of various scenarios to simulate real user behavior over a longer period.

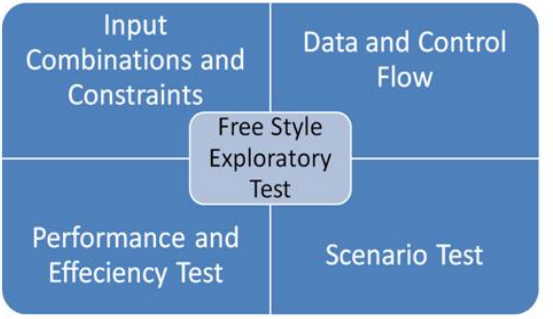

Fig. 3. A Visual Framework on Exploratory testing

Free Style Exploratory test is the main aspect of ET where the testers use their intellectual skills and employ the testing procedures. The tester tends to use his / her full knowledge of the system to do things like looking for race conditions, or multiple concurrent login on the same account, or edit URLs etc.

The strength of automated testing is the fact that the tests can be run often and never get tired, which is undeniable. However, it is expected to find problems that they are expected to find, according to the test cases. Rather, people who have the domain knowledge about the application can surprisingly find new problems that automated tests are incapable of revealing it. Hence, Exploratory Testing aims at improving the quality of the software by revealing the software bugs at the early stage to be fixed before the software is released to the customers and the stakeholders. However, there is no standard procedure or metrics available as how to measure the software quality by using this approach which is still under research.

The Benefits of ET Framework

- It reduces cost
- It saves time
- Use the resources efficiently towards testing
- Better understanding of the domain knowledge on the product by the tester, and
- To ensure customer satisfaction

5. Discussion and Future work

There is no dedicated test team or testers available to only perform the testing. All testing activities are performed by the developers. Hence exploratory testing, if used appropriately, will allow testers working on projects in SME to begin testing the system sooner, and also help them to avoid the pitfalls of inattentional blindness.

No tools or approaches can solve all problems, and exploratory testing may not completely replace scripted manual testing, but it does provide clear value for most projects in SME's where they do not have proper testing process.

This research work will continue further to develop a theoretical and conceptual framework to help solve their bug identification process in a simple way to all unstructure small and medium size software company, where by the quality and standard of the system are achieved.

References

1. Snežana Popović, Ljubomir Lazić, Nikos E Mastorakis.: Orthogonal Array and Virtualization as a Method for Configuration Testing Improvement, Engineering of Computer Based Systems, First IEEE Eastern European Conference on the , pp.148,149, 2009.
2. August L. P. de l'Annee de Betrancourt, Peter R. Lamers.: Devloping a suitable structured testing approach for a small web development company, 2004.
3. Ayşe Tosun, Ayşe Bener, Burak Turhan.: Implementation of a Software Quality Improvement Project in an SME: A Before and After Comparison, Proc. 35th Euromicro Conference on Software Engineering and Advanced Applications, 2009, pp. 203 – 209.
4. Johan van Zyl.: Software Testing in a Small company: A Case Study, Technical Report, University of Pretoria, 2010.

5 Pino FJ, Garcia Felix, Piattini M, editors.: Key processes to start software process improvement in small companies. SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing, 2009, pp. 509-516.

6 Von Wangenheim CG, Weber S, Hauck JCR, Trentin G.: Experiences on establishing software processes in small companies. Information and Software Technology, 2007, 48(9):890 - 900.

7 Evelyn Moritz.: Case study: How Analysis of Customer found Defects can be used by System Test to Improve Quality ICSE, 2009.

8 Cem Kaner.: Fundamental Challenges in Software Testing, Available at http://www.testingeducation.org/a/fundchal.pdf. 2003.
9 Lozina Shoaib, Aamer Nadeem, Aisha Akbar (2009).: An empirical Evaluation of the influence of Human Personality on Exploratory Software Testing, 2009.
10. Andrew Austin and Laurie Williams.: One Technique is not Enough: A comparison of Vulnerability Discovery Techniques, 2011.
11. B. Boehm.: Software Engineering Economics. USA; Printice (Prentice) Hall, 1984.
12. G. McGraw, Software Security: Building Security In. Boston, USA: Pearson Education, 2006.
13. Rodrigo Souza and Christina Chavez.: Characterizing Verification of Bugs Fixes in Two Open Source IDEs, MSR, 2012.
14. Sarah Al-Azzani, Rami Bahsoon.: Using Implied Scenarios In Security Testing, SESS '10, Cape Town, South Africa, May 2010.
15. Samer Al-Zain, Derar Eleyan, Joy Garfield. (2012), "Automated User Interface Testing for Web Applications and Test Complete", CUBE 2012, Pune, Maharashtra, India, September 3–5, 2012.
16. Vesa Kettunen, Jussi Kasurinen, Ossi Taipale, and Kari Smolander.: A Study on Agility and Testing Processes in Software Organizations" ISSTA'10, Trento, Italy, July 12–16, 2010.
17. Hisham M. Haddad, Nancy C. Ross, and Donald E. Meredith " A Framework forInstituting Software Metrics in Small Software Organizations", Int.J. of Software Engineering, IJSE, Vol.5, No.1, 2012.
18. Ghazia Zaineb and Irfan Anjum Manarvi.: Identification and Analysis of Causes for Software Bug Rejection with their Impact Over Testing Efficiency, International Journal of Software Engineering & Applications (IJSEA), Vol.2, No.4, 2011.
19. Prakash V and Gopalakrishnan.: Testing efficiency exploited: Scripted versus Exploratory testing, 2011.
20. Juha Itkonen, Mika V. Mantyala and Casper Lassenius.: Defect Detection Efficiency: Test case based Vs. Exploratory Testing, First International symposium on Empirical Software Engineering and Measurement, 2007
21. Juha Itkonen, Mika V. Mäntylä and Casper Lassenius.: How Do Testers Do It? An Exploratory Study on Manual Testing Practices, Third International Symposium on Empirical Software Engineering and Measurement, 2009.
22. Beer, A., & Ramler, R.: The Role of Experience in Software Testing in Practice, Proceedings of Euromic o Conference on Software Engineering and Advanced Applications, pp. 258-265, 2008.

23. Yoni Meijberg.: Time for Testing at an intermediate Dutch SME, BSc Thesis, University of Twente, 2008.
24. Bach, J.: Exploratory Testing. In E. van Veenendaal, ed. The Testing Practitioner. Den Bosch: UTN Publishers, pp 253 .- 265. 2004.

ISBN 978-967-0314-25-6