
UNIVERSITI PUTRA MALAYSIA

# WEB APPLICATION AS INTERMEDIARY TOOL FOR MANAGING CROWDTESTING IN PUBLIC SERVICE SOFTWARE PROJECT

MOHD SYAHMI BIN MUSTAPHA

FSKTM 2019 14

# MASTERS PROJECT REPORT

# WEB APPLICATION AS INTERMEDIARY TOOL FOR MANAGING CROWDTESTING IN PUBLIC SERVICE SOFTWARE PROJECT

## SSE5988 – PROJECT IN SOFTWARE ENGINEERING

**SUPERVISOR:**

**DR. MAR YAH BINTI SAID**

**STUDENT:**
**MOHD SYAHMI BIN MUSTAPHA (GS48105)**

## ABSTRACT

Software testing is important to ensure correctness of the software, gaining confidence from stakeholders, and contributing towards achieving high quality software. One approach to conduct software testing is through crowdtesting. It allows people from the crowd to test a particular software using their own devices in real environment. Currently in public service sector there is no existing intermediary tool to manage crowdtesting activities for public service software project. Therefore, public service software project relied on common testing approaches such as testing by internal employees or outsourced to specific suppliers, that in turn making public service software projects facing the risk of inadequate testing. This study intends to determine whether the implementation of crowdtesting is able to address the problems of inadequate testing in public service software project and to propose a web application as intermediary tool for crowdtesting in public service. This study employed interviews and survey with IT practitioners in public service sector to understand the applicability of crowdtesting in public service and specifications for the proposed intermediary tool. The intermediary tool were then evaluated through case study to determine its effectiveness to manage crowdtesting for public service software project. Based on the evaluation, it shows that most of the participant agree that the intermediary tool shows effectiveness in terms of defect detection, cost benefit, time, and testing coverage.

**TABLE OF CONTENT**

# CHAPTER 1

# OVERVIEW OF PROJECT

This chapter will explain in detail the introduction, problem statement in this study, objective of study, and scope of study.

## 1.1 Introduction

Software testing is an important phase in software development life cycle. The objective of software testing is to ensure correctness of the software, finding defects as early as possible, gaining confidence from stakeholders, and contributing towards achieving high quality software (Graham et al. 2008). Testing activities should be conducted throughout every level of software development life cycle and should start at the early stage of software development project. Early testing is crucial to ensure that defects can be detected as early as possible thus making it easier, cheaper to remove and preventing the defects from propagating into the software product. Software testing are commonly conducted by internal employees of companies or outsourced to specific suppliers (Zogaj et al. 2014; Yan et al. 2014). Another approach to conduct software testing is through crowdtesting that leverage on the concept of crowdsourcing by opening the testing activities for participation to the mass of people in public.

The term crowdsourcing is derived from the combination of the words outsourcing and crowd (Rouse, 2010). Crowdsourcing is a concept to define the outsourcing of certain tasks that are usually performed by employees of an organization to the mass of people in the public through public invitation (Howe, 2008). Crowdsourcing is an approach to outsource internal tasks to the mass of people in the public using internet in order to reduce costs, getting wide range of people to conduct the tasks, and faster delivery of the tasks' objective (Hoßfeld, Keimel et al. 2013). With the advancement of internet technology and Web 2.0, organizations and companies have taken the advantage to connect with the people in public to conduct crowdsourcing activities as it is considered to be more efficient and provide cost saving (Zogaj et al. 2014). Subsequently, crowdsourcing

has been applied in software engineering domain to conduct requirement elicitation, design, coding, and testing of software (Mao et al. 2017).

Crowdtesting (Leicht et al. 2016; Alyahya & Alrugebh, 2017; Leicht, 2018) also known as crowdsourced software testing is an approach of conducting software testing where people from the crowd located in different places to test particular software using their own devices in real environment. Project owners of the software essentially outsource the testing activity to the mass of people and granting access for the crowd to test their software (Leicht, 2018; Guaiani & Muccini, 2015). Crowdtesting is more focused on the output of the software under test based on certain given input since source code usually is not available to the crowd testers (Leicht et al. 2017).

Crowdtesting provides several advantageous for software testing. For testing that requires high number of testers, crowdtesting provides the avenue to attract high number of participants in relatively short time and low cost (Schneider & Cheung, 2013; Liu et al. 2012). Through crowdtesting, software can be tested in the real environment, conducted by the real user, getting quick responses from the testers, and expediting testing activities by leveraging on distributed resources in the crowd (Guaiani & Muccini, 2015; Mao et al. 2017). Furthermore, crowdtesting able to reduce the cost of internal staffing since it provides avenue for companies to engage with testers from the external crowd whenever it is required (Mao et al. 2017). By opening participation for testing to the crowd, crowdtesting can garner participation from expertise in software testing as well as common users to conduct specific testing activities such as usability testing.

## 1.2 Problem Statement

Currently in public service sector there is no existing intermediary tool that can manage crowdtesting activities specifically for public service software project. Unlike commercial software, public service software project requires more emphasize on the identity of testers involved as well as confidentiality of government data and resources. Existing commercial intermediary tools do not provide adequate measures for identity of testers and confidentiality which involve government data and resources. Therefore, public service software project only relied on common testing approaches such as testing by internal employees or outsourced to specific suppliers, that in turn making public service software projects facing the risk of inadequate testing for functional as well as non-functional of the software under testing. Since most of current software such as web applications and mobile applications can be accessed by users using wide range of platforms and devices, to conduct testing with all the possible combination of available platforms and devices by the common testing approaches are not practical and economically not viable (Leicht et al. 2017). Project that rely only on the common testing approaches such as internal testing team might not be able to test the software with all possible combination of platforms and devices due to time and cost constraints. The common testing approaches require allocating fixed number of staff to conduct the testing, and arranging fixed testing sessions for the testing to be done simultaneously, thus can result in time consuming activity and limited flexibility. Conducting testing with fixed number of staff in some specified location might not represent the sample of the actual user (Hoßfeld et al. 2013). The common testing approaches might not be suitable to test software that can run on multiple hardware such as mobile applications due to challenges that come wide variety of possible configurations that will affect the behavior of the software (Guaiani & Muccini, 2015). Based on a study of current practices in software development in Malaysia, it has found out that 73.2% of respondents viewed that current practices of software testing alone is inadequate to ensure the software under testing has achieved its specification (Baharorn et al. 2016).

## 1.3 Objective

This study consist of research work on the implementation of crowdtesting in public service domain and development of intermediary tool for the implementation of crowdtesting. The objectives of this study are as follow:

i. To propose web application as intermediary tool in managing implementation of crowdtesting in public service software project.
ii. To develop web application as intermediary tool for crowtesting in public service software project.
iii. To evaluate the effectiveness of the intermediary tool in managing crowdtesting for public service software project.

## 1.4 Scope

The scope of study is focusing on the implementation of crowdtesting approach for software projects in Malaysian public service sector. This study is to produce an intermediary web tool for the use of public service departments and agencies to conduct crowdtesting for public service software projects. In order to address the aspects of testers' identity and confidentiality of government data and resources, the crowd to be involved in the crowdtesting is designated from government servants.

## 1.5 Tools

This project employed several tools to accomplish the design and development of the intermediary tool. The design of the intermediary tool used online diagram software Lucidchart and Creately diagram maker to come up with the related UML diagrams. The intermediary tool was developed using language PHP version 7.2.2 and Laravel PHP Framework version 5.6.39 to provide structure and quality of the developed web application. Development was conducted on integrated development environment (IDE) Visual Studio Code version 1.30.2. Database engine of the intermediary tool employed MySQL

version 5.0.12 to store all the data involved. Web server Apache version 2.4.29 is used to run the intermediary tool as web application.

## 1.6 Summary of Contribution

Throughout this study, several outcomes have been achieved through activities that have been conducted such as interview, survey, development, and case study. Through interview and survey, current adoption level of crowdtesting in public sector has been identified and the suitability of implementing crowdtesting for public sector software project has been explored. The interview has also identified the main requirements of the crowdtesting intermediary tool for the implementation in public service domain. Based on the requirements, the prototype of the crowdtesting intermediary tool has been developed. Through the case study, the effectiveness of the proposed intermediary tool has been evaluated based on four metrics: defect detection capability, cost benefit, time, and testing coverage.

## 1.7 Thesis Structure

This report is structured into seven chapters. Chapter 2 covers on the literature review of existing studies related to the area of this study. In Chapter 3, methodologies that were employed in this study are explained in detail. Chapter 4 describes the interview and survey that have been conducted and provide the result and analysis. Chapter 5 elaborates on the development of the intermediary tool and the user interfaces of the tool. Chapter 6 explains on the case study that have been conducted to evaluate the effectiveness of the proposed intermediary tool and analyzes result of the evaluation. Finally, Chapter 7 discusses on the conclusion of this study.

REFERENCES


[1] Zogaj, S., Bretschneider, U., & Leimeister, J. M. (2014). Managing crowdsourced software testing: a case study based insight on the challenges of a crowdsourcing intermediary. Journal of Business Economics, 84(3), 375-405.

[2] Leicht, N., Blohm, I., & Leimeister, J. M. (2017). Leveraging the Power of the Crowd for Software Testing. IEEE Software, 34(2), 62-69.

[3] Leicht, N., Blohm, I., & Leimeister, J. M. (2016). How to Systematically Conduct Crowdsourced Software Testing? Insights from an Action Research Project. In Proceedings of International Conference on Information Systems (ICIS 2016).

[4] Leicht, N., Knop, N., Müller-Bloch, C., & Leimeister, J. M. (2016). When Is Crowdsourcing Advantageous? The Case of Crowdsourced Software Testing. In Proceedings of European Conference on Information Systems (ECIS 2016).

[5] Leicht, N. (2018). Given Enough Eyeballs, all Bugs are Shallow–A Literature Review for the Use of Crowdsourcing in Software Testing. In Proceedings of Hawaii International Conference on System Sciences (HICSS 2018).

[6] Tung, Y. H., Tseng, S. S., & Tsai, W. T. (2014). A novel approach to collaborative testing in a crowdsourcing environment. Journal of Systems and Software, 87, 137-137.

[7] Leicht, N., Rhyn, M., & Hansbauer, G. (2016). Can Laymen Outperform Experts? The Effects of User Expertise and Task Design in Crowdsourced Software Testing. In Proceedings of European Conference on Information Systems (ECIS 2016)

[8] Upadhyay, A., Khan, M. W., & Chaurasia, P. K. (2017). The Impact of Crowdsourcing in Software Engineering: Software Testing Perspective. International Journal of Advanced Research in Computer Science, 8(7).

[9] Cui, Q., Wang, J., Yang, G., Xie, M., Wang, Q., & Li, M. (2017, July). Who Should Be Selected to Perform a Task in Crowdsourced Testing?. In Proceedings of Computer Software and Applications Conference (COMPSAC), 2017 IEEE 41st Annual, 1, 75-84.

[10] Sherief, N., Jiang, N., Hosseini, M., Phalp, K., & Ali, R. (2014, May). Crowdsourcing software evaluation. In Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering

[11] He, M., Tang, H., Wu, G., Wei, J., & Zhong, H. (2015, November). A Crowdsourcing framework for Detecting Cross-Browser Issues in Web Application. In Proceedings of the 7th Asia-Pacific Symposium on Internetware, 239-242

[12] Howe, J. (2008). Crowdsourcing: How the power of the crowd is driving the future of business. New York, NY: Random House.

[13] Rouse, A. C. (2010). A preliminary taxonomy of crowdsourcing. In Proceedings of the Australasian Conference on Information Systems 2010, 76, 1-10.

[14] Hoßfeld, T., Hirth, M., & Tran-Gia, P. (2012). Crowdsourcing. Informatik-Spektrum, 35(3), 204-208.

[15] Baharorn, F., Deraman, A., & Hamdan, A. R. (2016). A survey on the current practices of software development process in Malaysia. Journal of ICT, 4, 57-76.

[16] Alyahya, S., & Alrugebh, D. (2017). Process Improvements for Crowdsourced Software Testing. International Journal Of Advanced Computer Science And Applications, 8(6), 32-40.

[17] Hoßfeld, T., Keimel, C., Hirth, M., Gardlo, B., Habigt, J., Diepold, K., & Tran-Gia, P. (2013). Crowdtesting: A novel methodology for subjective user studies and QoE evaluation. University of Würzburg, Tech. Rep, 486.

[18] Guaiani, F., & Muccini, H. (2015, May). Crowd and laboratory testing can they co-exist?: an exploratory study. In Proceedings of the Second International Workshop on CrowdSourcing in Software Engineering, 32-37.

[19] Zanatta, A. L., Machado, L. S., Pereira, G. B., Prikladnicki, R., & Carmel, E. (2016). Software Crowdsourcing Platforms. IEEE Software, 33(6), 112-116.

[20] Peng, X., Babar, M. A., & Ebert, C. (2014). Collaborative software development platforms for crowdsourcing. IEEE software, 31(2), 30-36.

[21] De Souza, E. F., de Almeida Falbo, R., & Vijaykumar, N. L. (2015). Knowledge management initiatives in software testing: A mapping study. Information and Software Technology, 57, 378-391

[22] Mao, K., Capra, L., Harman, M., & Jia, Y. (2017). A survey of the use of crowdsourcing in software engineering. Journal of Systems and Software, 126, 57-84.

[23] Olson, D. L., & Rosacker, K. (2013). Crowdsourcing and open source software participation. Service Business, 7(4), 499-511.

[24] Stol, K. J., & Fitzgerald, B. (2014). Two's company, three's a crowd: a case study of crowdsourcing software development. In Proceedings of the 36th International Conference on Software Engineering, 187-198.

[25] Rayna, T., & Striukova, L. (2010). Large-scale open innovation: open source vs. patent pools. International Journal of Technology Management, 52(3/4), 477-496.

[26] Schneider, C., & Cheung, T. (2013). The power of the crowd: Performing usability testing using an on-demand workforce. In Information Systems Development (pp. 551-560). Springer, New York, NY.

[27] Liu, D., Bias, R. G., Lease, M., & Kuipers, R. (2012). Crowdsourcing for usability testing. Proceedings of the Association for Information Science and Technology, 49(1), 1-10.

[28] Nebeling, M., Speicher, M., & Norrie, M. C. (2013). Crowdstudy: General toolkit for crowdsourced evaluation of web interfaces. In Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems (pp. 255-264). ACM.

[29] Graham, D., Van Veenendaal, E., & Evans, I. (2008). Foundations of software testing: ISTQB certification. Cengage Learning EMEA.

[30] Hass, A. M. (2014). Guide to advanced software testing. Artech House.

[31] Musson, R., Richards, J., Fisher, D., Bird, C., Bussone, B., & Ganguly, S. (2013). Leveraging the crowd: how 48,000 users helped improve Lync performance. IEEE software, 30(4), 38-45.

[32] Dolstra, E., Vliegendhart, R., & Pouwelse, J. (2013). Crowdsourcing gui tests. In Software Testing, Verification and Validation (ICST), 2013 IEEE Sixth International Conference on (pp. 332-341). IEEE.

[33] Vliegendhart, R., Dolstra, E., & Pouwelse, J. (2012). Crowdsourced user interface testing for multimedia applications. In Proceedings of the ACM multimedia 2012 workshop on Crowdsourcing for multimedia (pp. 21-22). ACM.

[34] Liang, C. J. M., Lane, N. D., Brouwers, N., Zhang, L., Karlsson, B. F., Liu, H., Liu, Y., Tang, J., Shan, X., Chandra, R., & Zhao, F. (2014). Caiipa: Automated large-scale mobile app testing through contextual fuzzing. In Proceedings of the 20th annual international conference on Mobile computing and networking (pp. 519-530). ACM.

[35] Yan, M., Sun, H., & Liu, X. (2014). iTest: testing software with mobile crowdsourcing. In Proceedings of the 1st International Workshop on Crowd-based Software Development Methods and Technologies (pp. 19-24). ACM.

[36] Mao, X., Hou, F., & Wu, W. (2015). Multi-Agent System Approach for Modeling and Supporting Software Crowdsourcing. In Crowdsourcing (pp. 73-89). Springer, Berlin, Heidelberg.

[37] Unterkalmsteiner, M., Gorschek, T., Islam, A. M., Cheng, C. K., Permadi, R. B., & Feldt, R. (2012). Evaluation and measurement of software process improvement—a systematic literature review. IEEE Transactions on Software Engineering, 38(2), 398-424.

[38] Ghapanchi, A. H., & Aurum, A. (2011). Measuring the effectiveness of the defect-fixing process in open source software projects. In System Sciences (HICSS), 2011 44th Hawaii International Conference on (pp. 1-11). IEEE.

[39] Ghapanchi, A. H., Aurum, A., & Daneshgar, F. (2012). The Impact of Process Effectiveness on User Interest in Contributing to the Open Source Software Projects. JSW, 7(1), 212-219.

[40] Wang, J., Yang, Y., Yu, Z., Menzies, T., & Wang, Q. (2018). Crowdtesting: When is The Party Over?. arXiv preprint arXiv:1805.03218.

[41] Knop, N., & Blohm, I. (2018). Leveraging the Internal Work Force through Crowdtesting “Crowdsourcing in Banking.

[42] Nemoto, T., & Beglar, D. (2014). Likert-scale questionnaires. In JALT 2013 Conference Proceedings (pp. 1-8).

[43] Yan, M., Sun, H., & Liu, X. (2015). Efficient Testing of Web Services with Mobile Crowdsourcing. In Proceedings of the 7th Asia-Pacific Symposium on Internetware (pp. 157-165). ACM.