



University of HUDDERSFIELD

University of Huddersfield Repository

Amen, Bakhtiar, Mahmood, Sardasht and Lu, Joan

Mobile application testing matrix and challenges

Original Citation

Amen, Bakhtiar, Mahmood, Sardasht and Lu, Joan (2015) Mobile application testing matrix and challenges. *Computer Science & Information Technology*. pp. 27-40. ISSN 2231-5403

This version is available at <http://eprints.hud.ac.uk/29254/>

The University Repository is a digital collection of the research output of the University, available on Open Access. Copyright and Moral Rights for the items on this site are retained by the individual author and/or other copyright owners. Users may access full items free of charge; copies of full text items generally can be reproduced, displayed or performed and given to third parties in any format or medium for personal research or study, educational or not-for-profit purposes without prior permission or charge, provided:

- The authors, title and full bibliographic details is credited in any copy;
- A hyperlink and/or URL is included for the original metadata page; and
- The content is not changed in any way.

For more information, including our policy and submission procedure, please contact the Repository Team at: E.mailbox@hud.ac.uk.

<http://eprints.hud.ac.uk/>

MOBILE APPLICATION TESTING MATRIX AND CHALLENGES

Bakhtiar M. Amen¹, Sardasht M. Mahmood² and Joan Lu³

^{1,3}School of Computing and Engineering,
University of Huddersfield, Huddersfield, UK
²Statistics and Computer, College of Commerce,
University of Sulaimani, Sulaimani, Iraq

ABSTRACT

The adoption of smart phones and the usages of mobile applications are increasing rapidly. Consequently, within limited time-range, mobile Internet usages have managed to take over the desktop usages particularly since the first smart phone-touched application released by iPhone in 2007. This paper is proposed to provide solution and answer the most demandable questions related to mobile application automated and manual testing limitations. Moreover, Mobile application testing requires agility and physically testing. Agile testing is to detect bugs through automated tools, whereas the compatibility testing is more to ensure that the apps operates on mobile OS (Operation Systems) as well as on the different real devices. Moreover, we have managed to answer automated or manual questions through two mobile application case studies MES (Mobile Exam System) and MLM (Mobile Lab Mate) by creating test scripts for both case studies and our experiment results have been discussed and evaluated on whether to adopt test on real devices or on emulators? In addition to this, we have introduced new mobile application testing matrix for the testers and some enterprises to obtain knowledge from.

KEYWORDS

Mobile App Testing, Testing Matrix, Automated and Manual Testing.

1. INTRODUCTION

The world of mobile application is emerging rapidly and it attracted extensive research interests [12]. In fact, due to easiness of technology, every day millions of mobile users are depending on their mobile apps to conduct and browse internet for social networking (e.g., Facebook, Twitter, LinkedIn, Instagram), for online banking (transaction and balance sheet), for emailing (arrange meeting and solving problems). According to [23] every year extraordinary numbers of applications are flooding onto the market with forecast of 76.9 billion global downloads in 2014 worth of US\$35 billion [34]. Therefore, the comprehensive mobile application testing is crucial to direct high quality of applications and satisfies user needs, whereas studies indicated that developers are more focusing on the application back end and functionality rather than use experiences. In fact, a user feedback is one of the fundamental parts of application's reputation to ensure app's owners with successful or failure of their application [20]. Commonly, users can easily drop interesting in problematic mobile app, and will abandon it after only one or two failed attempts.

The purpose of this paper is to investigate and provides solutions to firstly; whether agility testing or physical testing is the most appropriate to adopt? Secondly; identify new testing matrix for testers to obtaining knowledge from. Thirdly and finally; introduce new mobile application test

David C. Wyld et al. (Eds) : CCSIT, SIPP, AISC, NLP - 2015
pp. 27–40, 2015. © CS & IT-CSCP 2015

DOI : 10.5121/csit.2015.50403

strategy, testing state-of-art. More to this, we have analysed both case studies MLM and MES results and critically evaluate individual findings for experiment results.

This paper is organised as it follow; Section two is consists of mobile app definition, test definition, mobile test matrix including test environments, test techniques, test levels and test scopes. Section three presents existing mobile app testing tools while section four introduces testing strategy. Section five provides related work. Case studies experiment results illustrated in section six and section seven provide conclusion and future of work.

2. BACKGROUND

This section is consist of three parts; definitions of mobile application, testing definitions and mobile application testing matrix.

2.1 Mobile Application

Mobile application is a written source code in various programming languages (e.g. Java) and designed for smartphones to operate on Mobile OS platforms (e.g. Android, iOS). The purpose of mobile application is to enhance user's daily life throughout (online banking transactions and emails) or for entertainments like (social media and gaming). The novel of mobile app is designed for the user to input data from touch screen and expected output results efficiently and effectively regardless of the application's development knowledge.

2.2 Testing Definitions

Testing defined by [2] [25] [35] is 'the process of executing a program with the intent of finding errors'. In fact, test is one of the fundamental requirements of mobile app development methodology phases in the development life cycle to measure the quality of application's standard and to avoid vital bugs. Due to the rapid growth of mobile apps every year, developers and enterprises are losing confidence in to relays on the best testing techniques and adopt economical ways of delivering mobile apps in to the market [16] [19] [32].

2.3 Mobile Application Testing Matrix

Mobile Apps testing is more complicated than the software or web apps testing due to the nature of development specifications techniques like; OS platforms, devices and screen resolutions [14] [33]. However, we have managed to impalement and organise mobile application testing matrix from [40] to Test Techniques, Testing Environment, Test Level, and Test Scopes as depicted in Figure 1.

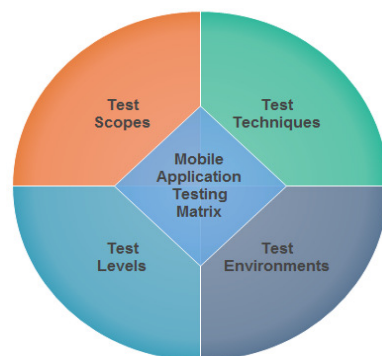


Figure 1: Mobile Application Testing Matrix

2.3.1. Test Techniques

According to Selvam in [40] the principal test challenge arise throughout mobile apps development process “how to test the apps”. The authors of [6] [7] [40] emphasized that, it’s very crucial to decide whether automated or manual testing are the most appropriate testing techniques to adopt in mobile apps testing stage, Figure 2 depicted the techniques. Moreover, we have conducted both techniques for our case studies of MES and MLM in order to obtain our paper’s objective questions. The experiment results of both case studies were demonstrated in the result section with emphasised issues in each technique. On the other hand, researchers are indicating that automated testing is more relying on programming development tool for instance Monkey Talk, Test Plant and other top mobile apps testing tools depicted in Table 3. Whereas, according to the researchers prospective, manual testing is more relying on human interaction like usability testing.

2.3.1.1 Automated Testing

Automated testing technique is highly desirable, for this reason automated testing is capable in decrease of human errors, efficiency in finding bugs, with less time consuming [3]. In fact, automated testing is permit tester to verify the main critical features of the application by testing different data sets [42]. According to Quilter in [39] automated testing has capability to execute large volumes of repeatable scenarios beyond human capabilities to undertake manually.

2.3.1.2 Manual Testing

Manual testing is very time-consuming compare to automated testing, and often it has limitation in testing through the limited user-interface of the mobile device. Manual testing acknowledge tester to create test case and follow the test case design, instruction design to achieve their specific test goals [19] [40]. In addition to this, automated Vs manual results would be demonstrate in the testing results section Seven.

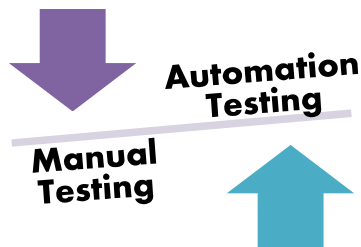


Figure 2: illustrates mobile App tests techniques

2.3.2 Test Environments

The critical demands on test environments are widely remained within scientist and enterprises. Kim in [28] argues that ‘developers establish mobile applications on a desktop computer using a development toolkit and emulator’. Therefore, this is indicating that developer is enabling to test on both real device and simulator. Whereas, Simulator has matching look and feel of real device and it executes on desktop operating system. According to Quilter in [39] Simulator-based approaches have various specific advantages like lower cost, scalability, time and easy of implement, opposite to the real device. Figure 3 depicted mobile application environments.

2.3.3 Test Levels

Test level is one of the fundamental crucial parts of mobile application development. Mobile apps test level consists of; Unit Testing [11] [24], Functionality Testing [24], Usability Testing [24] [35], Compatibility Testing, Regressions Testing [24], Security Testing [18][22], Acceptance Testing [18][22] and Network Testing [35].

Table 1 Different Testing Level in Mobile Application

Test Levels	Who does it?	Specification	Why this type?	When is Necessary?	Opacity
Unit Testing [11], [24]	Programmer	Complete the test automatically through run the test script to ensure that the test has turned from "red" (failure) to "green" (success) [11]	To check app code structures to find bugs and errors	When the Programmer wrote a piece codes	White box Testing
Functionality Testing [24]	Programmer	Verifies app/site content (images, text, controls, and links) as it is displayed on the actual mobile devices. [11] [22]	To check the app's functionality and compare the user's requirement	During and after the development stage	Black box and While box Testing
Usability Testing [24][35]	Client, Users	Refer to the app's effectiveness, efficient and satisfaction [24][35]	To check apps link validation, multiple browsers' support, screen resolution. [24]	After app's functionality completed.	Black box Testing
Compatibility Testing	Programmer Independent Tester	Refers to validation of the apps for different operating system, mobile devices [24]	To verify and validate of app's compatibility	When the app completed and before deliverable	Black box and While box Testing
Regressions Testing [24]	Client and Independent tester [24]	Expect apps operating as intends to [24][35]	To ensure the correctness of app's operation	Before the application deployment	Black box and While box Testing
Security Testing [18][22]	Programmer	To ensure with app's encryption/decryption techniques in used sensitive data of users (e.g. ID, Password, Credit card details) [35]	To ensure with information protection landscape [35]	At end of development process	Black box and While box Testing
Acceptance Testing [18][22]	Client	The objective of acceptance testing is to create confidence in the application [18][22]	To Delivery and evaluate the application in aspect of end user point of view	When the user acceptance criteria met with the requirements [18][22]	Black box and While box Testing
Network Testing [35]	Network expertise and Programmer	Compatibility app's with different Network signal (Wi-Fi, 2G, 3G, 4G) Impact of Connectivity Issues [35]	To check app's connection strength and weakness.	Before the app's deliverable phase	While box Testing

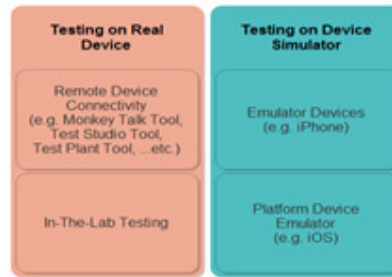


Figure 3: Mobile Application Testing Environments

2.3.4 Test Scopes

Generally, test scopes have been categorised in two major parts, functional (black box) and structural (white-box) [42] [14]. The following table is depicted the classification of each parts.



Table 2 Different Testing SCOPES in Mobile Application








Test Scopes	What is it?	Who does it?	Why this type?
Black Box Testing [14][42][43]	Known as functional and non-functional testing. Black box testing is a widely used in testing environments. The component under test has inputs, outputs, and a specification, which states the relationship between inputs and outputs. It ignores the internal mechanism of a system or component and focuses solely on the outputs generated without prior knowledge of it source code [14][24][42][43]	Independent undertake the test	To detect bugs, errors in the app's codes. Test app's functionalities [24]
White Box Testing [27][37]	Known as structural testing, cover the internal data structures that exercised to ensure validity of test conditions, with good knowledge of the source code [27][31][35][42][43]	Developers Execute This test	To detecting logical errors in the program code (Unit Test) [27]. [37]

3. STATE OF ART

In this section, testing tools that are supporting the testing techniques have been proposed specifically for mobile app testing environments. Each tool has been described in Table 3 in terms of their licenses whether they are open source tools, the table consists of tool's device support for instance Android, iPhone or multi platform as well as tool's scripting and languages. Finally provides the tool's specification testing types support.

Table 3: Mobile apps testing tools

Logo	License	Support Device	Scripting/ Language	Testing Types
	Open Source	Android	JAVA	Unit Testing, GUI interface [9]
 iOS Simulator	Open Source	Window or Mac iOS	Objective C	GUI interface Unit Testing [10]

	Open Source	iPhone & Android etc.	HTML5 & JAVA	Functional, GUI Testing [17]
	Open Source	Multi Platforms	Unit Test	GUI, Accepting Testing [46]
	Cost	iPhone & Android etc.	Test across mobile OS with a single script	GUI Test [44]
	Open Source	Variety of platforms	C#	Google Test UI
	Cost	Multi Platforms	A single Script	GUI Testing [38]
	Cost	Android, iPhone etc.	C#, Java, Perl & Python	Functionality & Speed Performance [13]
	Cost	iPhone, Android etc.	JAVA & Objective C	Functionality, Usability, Performance [26]

4. TEST STRATEGY

Before decide to adopt any test techniques on the mobile apps, it is necessary to have testing strategy in order to meet user's requirements, specifications and to avoid negative feedbacks from app's users. Furthermore, testing progress is important in terms of quality assurance. Figure 4 predicted test strategy plans for the testers to beware of from test document preparation to the application test acceptance/deliverable phase

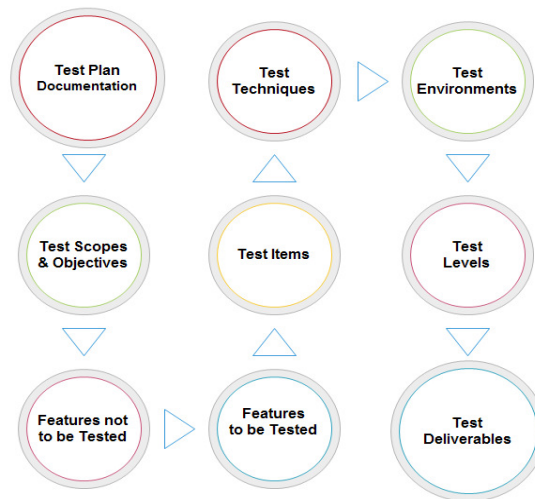


Figure 4: Mobile apps Test strategy plan

5. LITREATURE REVIEW

Haller in [20] proposed agile and compatibility testing for one a case study from Swisscom IT services to identify whether test on wild device and failure during the start-up application and focused on regression testing process.

Amalfitano et al. highlighted the results of automated testing experiment for Android mobile application platform [1]. The author specifically presents a technique for rapid crash testing and regression testing based on crawler that automatically builds models of application in GUI. Heo, Jeongyun et al. in [21] introduced new framework for evaluation of usability in mobile application that has been conducted based on a multi level, hierarchical model of usability factors, the author proposed case study for new framework to test his frameworks on in order to identify the characteristics of the framework.

Utest Inc. proposed usability testing mobile application for NHD Direct in 2011, whereas according to uTest Inc. the application is more likely to focus on symptom checking for mental health conditions, self-care and free advice [45]. Respectively, the objectives of NHS Direct Mobile application usability testing were to enhance the user's feedback and compotator app on top number one in iTunes charts for the best free apps within the first week of released app [4]. Knott suggested that it is necessary to implement some of the specific features of the application manually rather than conduct automated testing [29]. Functional testing consists of both input and output data. However, from the input aspect, mobile application receives two types of actions, whereas the first action is from GUI input by any keyboard keys and touch events, while the second action is the result output.

6. CASE STUDIES

Mobile Lab Mate (MLM) is one of the particular applications developed by the University of Huddersfield research team. The aim of this application is to support students in accessing into their account at anytime in anywhere in order to view their class materials and results effectively and efficiently. Furthermore, MLM application was a pilot case study and attempt to help developer to identify issues within application before the acceptance-testing phase. Figure 5 depicted the applications screen prints. Therefore, both testing techniques such as automated and manual have been conducted and the experiment result will be discussed in the result section.

On the other hand, Mobile Exam System (MES) was another pilot case study that has been conducted. The aim of this application was to support students throughout answering their questions online and save their exam time, to assist teachers to see the results efficiently and avoiding any misconduct mechanism during exam taken. In fact, both techniques of automated and manual testing have been carried out.



Figure 5: MLM & MES Mobile Application

Furthermore, both application case studies have been tested by open source automated tool known Monkey Talk. According to Corbett and Bridgwater 'Monkey Talk is the latest testing

platform from Gorilla Logic [5] [8]. Monkey Talk IDE (Integrated Development Environment) extends with Java Script API and it assist tester to creating, recording, and manage the test on actual devices. However, Monkey Talk is free and open source automated tool operates on iPhone and Android [5] [17]. The reason behind conducting Monkey Talk was to test the applications functionalities and have the test records while emphasis the demands of automated capabilities. Figure 6 depicted the use case design that we have made in the testing process in order to have better and clear of testing objectives

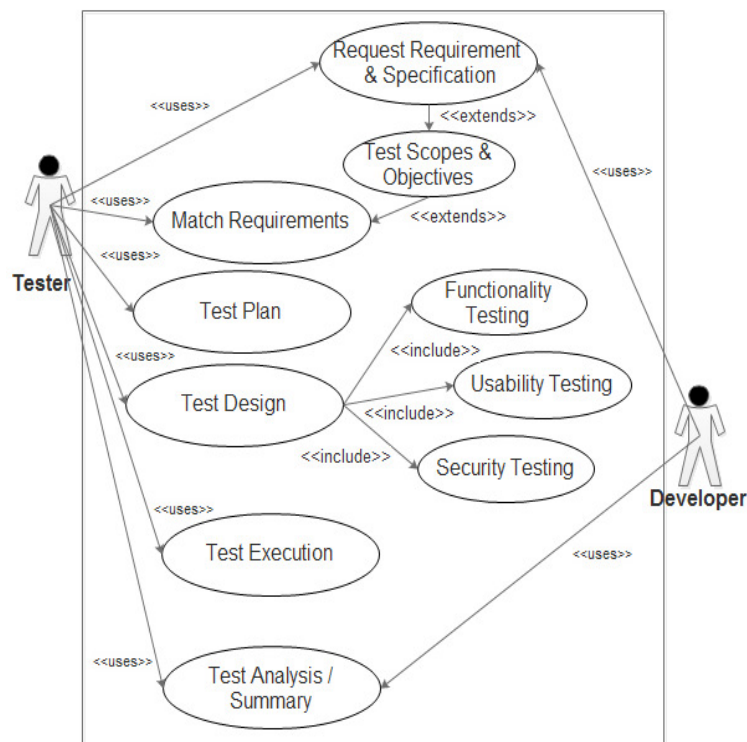


Figure 6: Case Study Use Case Test Process

7. RESULTS (EVALUATION AND ANALYSIS)

Test automated solution consists of: test scripts, connection between automated tool (PC) and the mobile device, remote control mechanism for the device, and an interaction strategy for the mobile device GUI depicted in (Figure 7,8,9 and 11). The selected solution affects the test script language. For example, Expertise, Keynote and Monkey Talk were only tools that capable of testing functionality as well as GUI. When scalable test configuration coverage is the main aim, the test scripts must run on multiple devices and potentially on various OS and OS versions. This requirement affects the connection between a test PC and the mobile device. First, a direct connection can exist from the PC to the device. Second, an indirect connection can exist that acts as a switch between various PCs and a large device pool.

The automated testing is a solution to improve the testing efficiency; it is the most important latest techniques to improve functionality testing as multiple device handlers, and to ensure that MES and MLM applications are resulting automated testing technique efficient and accurately. The following test script was for the MLM app's login function and result of the app's login function has predicted in Figure 8.

```

1) load("libs/MobileLabMate.js");
2) MobileLabMate.Login.prototype.run = function() {
3) this.app.login().run();
4) this.app.link("sign").click();
5) this.app.link("st").click();
6) this.app.input("studentname").enterText("Tester");
7) this.app.input("studentpassword").enterText("u0772370");
8) this.app.button("login").click();
9) this.app.link("Logout").click();
10)};

```

Figure 7: Login Test Script

Figure 8 depicted the test script for MLM new student who has not been registered before.

```

1) load("libs/MES.js");
2) MESapp.CreateAccount.prototype.run = function() {
3) this.app.createAccount().run();
4) this.app.link("sign").click();
5) this.app.link("i").click();
6) this.app.input("name").enterText("Tester2");
7) this.app.input("pass").enterText("1234567");
8) this.app.button("callAjax").click();
9) this.app.link("sign").click();
10) this.app.link("st").click();
11) this.app.input("studentname").enterText("tester2");
12) this.app.input("studentpassword").enterText("1234567");
13) this.app.button("login").click();
14) this.app.link("Logout").click();};

```

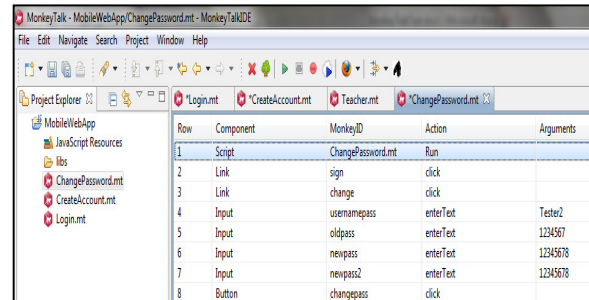
Figure 8: Create Account Test Script

```

1) load("libs/MobileLabMate.js");
2) MobileWebApp.ChangePassword.prototype.run = function() {
3) this.app.changePassword().run();
4) this.app.link("sign").click();
5) this.app.link("change").click();
6) this.app.input("usernamepass").enterText("Tester2");
7) this.app.input("oldpass").enterText("1234567");
8) this.app.input("newpass").enterText("12345678");
9) this.app.input("newpass2").enterText("12345678");
10) this.app.button("changePass").click();
11)};

```

Figure 9: Change password test script



Row	Component	MonkeyID	Action	Arguments
1	Script	ChangePassword.mt	Run	
2	Link	sign	click	
3	Link	change	click	
4	Input	usernamepass	enterText	Tester2
5	Input	oldpass	enterText	12345678
6	Input	newpass	enterText	12345678
7	Input	newpass2	enterText	12345678
8	Button	changepass	click	

Figure 10: change password test result

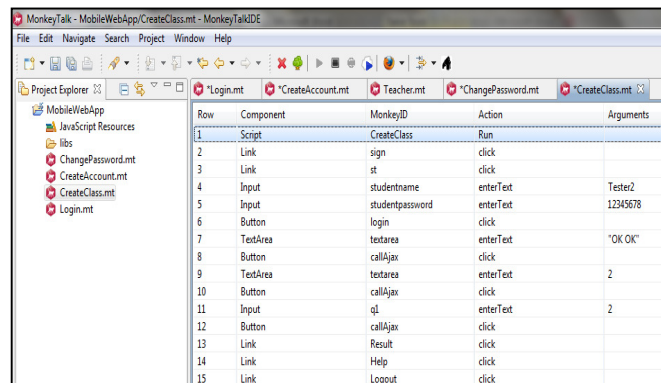
Figure 10 depicted the new class for the students and teachers screen print results and test scripts

```

1) load("libs/MobileLabMate.js");
2) MobileWebApp.CreateClass.prototype.run = function() {
3) this.app.createClass().run();
4) this.app.link("sign").click();
5) this.app.link("st").click();
6) this.app.input("studentname").enterText("Tester2");
7) this.app.input("studentpassword").enterText("12345678");
8) this.app.button("login").click();
9) this.app.textArea("textarea").enterText("OK OK");
10) this.app.button("callAjax").click();
11) this.app.textArea("textarea").enterText("2");
12) this.app.button("callAjax").click();
13) this.app.input("q1").enterText("2");
14) this.app.button("callAjax").click();
15) this.app.link("Result").click();
16) this.app.link("Help").click();
17) this.app.link("Logout").click();
18) };

```

Figure 11: Test script for new class



Row	Component	MonkeyID	Action	Arguments
1	Script	CreateClass	Run	
2	Link	sign	click	
3	Link	st	click	
4	Input	studentname	enterText	Tester2
5	Input	studentpassword	enterText	12345678
6	Button	login	click	
7	TextArea	textarea	enterText	"OK OK"
8	Button	callAjax	click	
9	TextArea	textarea	enterText	2
10	Button	callAjax	click	
11	Input	q1	enterText	2
12	Button	callAjax	click	
13	Link	Result	click	
14	Link	Help	click	
15	Link	Logout	click	

Figure 12: Test result for new class

On the other hand, one of the most important aspects was to consider and carry out functionality, usability and security testing. MLM application was operated normal, but still there were some bugs existed in the application during the functionality of “forgot password” link. However, change password functionality was not crucial and secure Figure 14 depicts the result of MLM functionality, usability as well as security.

In fact, due to the limited space, we have only illustrated a few initial test scripts while for each application of MLM and MES have had several test scripts. In fact, MES app was very secure in the aspects of authorisation, encryption and data store, but MLM apps has had some bugs within the application when the user have access to make more than 16 characters for username and password while in MLM user only unable to enter different characters accept numbers and letters between 8-20 length spaces. MLM apps do not have the limitation input. Therefore, these are some of weak points in MLM for the hacker to inject the database by random characters.

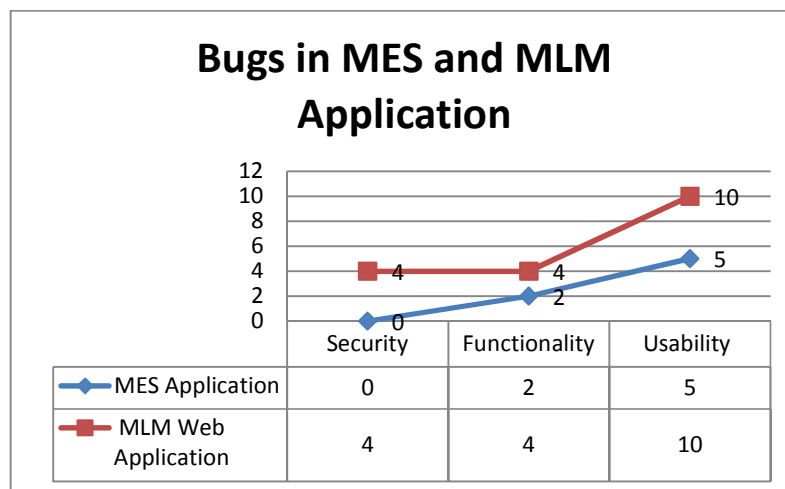


Figure 13: Bugs in both MES and MLM applications

Furthermore, testing functionality and usability activities were performed by real device as well as automated tool for each applications. Figure 13 indicates that MES apps have more bugs compare to MLM apps from manual testing results. Finally, the test scrip results are demonstrates that some functionality of MLM is not working as intends to do by automated limitation while they were working effectively on the real devices. On the other hand, the source code of the test scrip’s in Figure (7,8,9 and11) illustrated that some of functional of the MLM application is not structured accurately when two users were enable to create new account within the same email address and type in long characters or digits in the password field. However, from both case studies we have managed to highlight the limitation of each automated and manual testing in Table 4.

Table 4 DISTINCTIONS between Automated Testing and Manual Testing

Automated Testing	Manual Testing
<ul style="list-style-type: none"> - Testers require to conducting specific tool to execute the test. - Cost effectiveness - Programming knowledge is required. - Less staff’s required but tools are expensive. - Difficult to depend on automated, some app’s area still has to test manually. 	<ul style="list-style-type: none"> - Tester has to write a test case and executes on the application manually. - More likely to cost. - Not programming knowledge is required. - Skilled testers and staffs required. - Testing apps on real device is time-consuming [41]

<ul style="list-style-type: none"> - Automated avoid overloaded work and save more times. - Requirements does not changing frequently. 	<ul style="list-style-type: none"> - Staffs Training expensive. - Manual testing is more time considered to perform a test case. - Requirements more likely to changing frequently.
--	--

8. CONCLUSION AND FUTURE WORK

This paper is managed to answer the most demandable questions related to each mobile app's testing techniques, whether to conduct automated or manual testing. Tests were executed for both case study applications by Monkey Talk open source in order to identify bugs and errors in both case studies. Moreover, it is difficult decision for the testers to decide whether adopt automated or manual testing environments, for this reason, tester has to investigate in selected tool's limitation before the testing strategy had has planned. In fact, it is necessary for the testers to keep in mind testing objectives, testing has to be performed on many combination of devices, browsers, and operating systems rather than just depends on one test technique. Automated testing cannot to be judged by manual testing, for the following reasons:

1. Automated testing has only functional capabilities.
2. Automated testing has benefits of reducing time and cost.
3. Usability testing difficult to be conducted by automated testing.
4. More tests can be run in a shorter time in automated.

Finally, In order to obtain higher standard and quality mobile applications feedback, testing different activities throughout the application's development process and effective models, methods, techniques and tools are essential to be considered by the testers. Furthermore, we highly recommend testers to conduct both test techniques of automated and manual in order to cope with the fundamental necessity of the rapid delivery of these applications, for these reasons, combined both testing techniques will assist testers to identify some of the bugs and errors within the apps efficiently while it might be difficult to identifies them in automated testing on the real devices as we have predicted in our case studies result. To conclude, Automated testing is one of the efficient methods to guarantee of app's quality and performance within the mobile testing environments compare to manual testing. In the future, we implement our Mobile App's Testing Matrix and Testing Strategy in several real time applications within enterprises in order to enhance one powerful test technique for the testers to relays on.

REFERENCES

- [1] Amalfitano, D. Fasolino, A. Tramontana, P. and Federico, N. (2011). A GUI Crawling-based technique for Android Mobile Application Testing.
- [2] Bach, J.(1999). General Functionality and Stability Test Procedure. [online] Available at: <http://www.satisfice.com/tools/procedure.pdf> [Accessed 15th August 2014].
- [3] Bartley, M.(2008). "Improved time to market through au tomated software testing". Automation Testing, [Online] Available at: <http://www.testingexperience.com> [Accessed 23rd September 2014].
- [4] Bastien, C.(2008). "Usability Testing : A Review of Some Methodological and Technical Aspects of the Method." In ternational Journal of Medical Informatics 79(4): e18–e23. [Online] Available at: <http://dx.doi.org/10.1016/j.ijmedinf.2008.12.004>. [Accessed 23rd November 2014]. Available at:
- [5] Bridgwater, A.(2012). MonkeyTalk Tests iOS/Android Apps In The Mobile Jungle. [online] <http://www.drdobbs.com/open-source/monkeytalk-tests-iosandroid-apps-in-the/232602236> [Accessed 13th October 2014].
- [6] Brown, M.(2011). Mobile Developers Get Best Practices For App Testing. [online] Available at: <http://www.mobileapptesting.com/mobile-developers-get-best-practices-for-app-testing/2011/03/>. [Accessed 3rd September 2014].

- [7] Brown, M.(2011). MFiobile Functional Testing: Manual or Automated?. [online] Available at: <http://www.mobileapptesting.com/mobile-functional-testing-manual-orautomated/2011/05/>. [Accessed 8th August 2014].
- [8] Corbett, J.(2012). Quality Assurance testing in a mobile world. [online] Available at: <http://www.peteramayer.com/quality-assurance-testing-in-a-mobile-world>>. [Accessed 26th October 2014].
- [9] Developer.android.com (n. d). Testing. Retrieved from <https://developer.android.com/tools/testing/index.html> [Accessed 6th November 2014].
- [10] Developer.apple.com (2014). Testing and Debugging in iOS Simulator. Retrieved from https://developer.apple.com/library/ios/documentation/ides/conceptual/iOS_Simulator_Guide/TestingontheiOSSimulator/TestingontheiOSSimulator.html [Accessed 6thNovember 2014].
- [11] Dimitrios, V. (2013). "Estimation for Unit Root Testing."
- [12] Dongsong, Z. and Adipat, B. (2005). "Challenges Metho dologies, and Issues in the Usability Testing of Mobile Applications", International Journal of Human Computer Interaction, Vol. 18, 3
- [13] Experitest.com (n. d). SeeTestAutomation. Retrieved from <http://experitest.com/support/getting-started/download-2-2/> [Accessed 6th November 2014].
- [14] Freeman, H. (2002). "Softawre Testing". IEEE Instrumentation & Measurement Magazine, P.48-50. [online] Available at: <https://confluence.xpeqt.com/confluence/download/attachments/10322031/01028373.pdf> [Accessed 23rd June 2014].
- [15] Galorath, D. (2012). Software Project Failure Costs Billions. Better Estimation & Planning Can Help. [online] Available at: <http://www.galorath.com/wp/software-project-failure-costs-billions-better-estimation-planning-can-help.php> [Accessed 16 September 2014]
- [16] Gao Z. and Long, X. (2010). Adaptive Random Testing of Mobile Application. P.297, Vol 2. [Online] Available at: <http://ieeexplore.ieee.org.libaccess.hud.ac.uk/stamp/stamp.jsp?tp=&arnumber=5485442> [Accessed 26rd September 2013].
- [17] Gorillalogic.com (n.d). MonkeyTalk Open source Automation Testing Tool. [online] Available at: <https://www.gorillalogic.com/monkeytalk> [Accessed 18 August 2014].
- [18] Graham, D., Veenendaal, E., Evans, I. and Black, R. (2008). Foundations of Software Testing: ISTQB Certification. UK: Cengage Learning EMEA.
- [19] Harty, J.(2009). "A Practical Guide to Testing Wireless Smartphone Applications". Synthesis Lectures on Mobile and Pervasive Computing. 4(1), pp. 1 – 99. [online] Available at: www.morganclaypool.com [Accessed 11th October 2014].
- [20] Haller, K.(2013). "Mobile Testing". ACM SIGSOFT Software Engineering Notes, 38, 6, 1-8. [Online] Available at: <http://doi.acm.org/10.1145/2532780.2532813> (November 3rd, 2014).
- [21] Heo,J., Ham, D., Park, S., Song, C. and Yoon, W. (2009). "A Framework for Evaluating the Usability of Mobile Phones Based on Multi-level, Hierarchical Model of Usability Factors," Interacting with Computers, Vol. 21(4), pp. 263-275
- [22] Hetzel, B.(1998). The complete guide to software testing. (2nd ed.). Chichester: John Wiley & Sons.
- [23] Jacob, J.and Tharakan, M. (2012). "Roadblocks and their workaround, while testing Mobile Applications".The Magazine for Professional Testers. P8-16, Vol 19. [Online] Available at: <http://www.testingexperience.com/> [Accessed 23rd September 2014].
- [24] Jorgensen, P. (2002). "Software testing: acraftman's approach," CRCPress, p.359.
- [25] Kaner, C. (2002). Black Box Software Testing (Professional Seminar), Risk Based Testing And Risk-Based Test Management, [online] Available at: www.testing-education.org> [Accessed 12 August 2014]
- [26] Keynote.com (n. d). Mobile Testing. Retrieved from <http://www.keynote.com/solutions/testing/mobile-testing> [Accessed 6th November 2014].
- [27] Khan, M.and Sadiq, M. (2011). "Analysis of Black Box Software Testing Techniques: A Case Study". IEEE. pp. 1-5. [online] Available at: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6107931> [Accessed 2nd May 2014].
- [28] Kim, H., Choi, B. and Wong, W (2009). Performance Testing based on Test-Driven Development for Mobile Applications.P612-617. [online] Available at: <http://dl.acm.org.libaccess.hud.ac.uk/citation.cfm?id=1516349> [Accessed 24rd July 2014].
- [29] Knott, D. (2012). "The Enterprise Mobile Applications Development Framework", Best Practices in Mobile App Testing.P26, [online] Available at: <http://www.testingexperience.com/> [Accessed 23rd June 2014].

- [30] Kumiega, A. and Vliet, B. (2008). *Quality Money Management: Process Engineering and Best Practices for Systematic Trading and Investment*. USA: Academic Press.
- [31] Lewis, W. (2008). *Software Testing and Continuous Quality Improvement*. (3rd ed.). United States: Auerbach.
- [32] Liu, Z. Gao, X. Long, X. (2010). "Adaptive Random Testing of Mobile", *Computer Engineering and Technology (ICCET)*, 2010 2nd International Conference on , vol.2, no., pp.V2-297,V2-301, 16-18. [online] Available at: <http://ieeexplore.ieee.org.libaccess.hud.ac.uk/stamp/stamp.jsp?tp=&arnumber=5485442&tag=1> [Accessed 24rd June 2014].
- [33] Milano, D. (2001). *Android Application Testing Guide*. USA: Packt Publishing Ltd.
- [34] Muccini, H., Di Francesco, A. and Esposito, P. (2012), "Software testing of mobile applications: Challenges and future research directions," *Automation of Software Test (AST)*, 2012 7th International Workshop on , vol., no., pp.29,35 [online] Available at: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6228987&isnumber=6228973> [Accessed 23rd September 2012].
- [35] Myers, G., Badgett, T. and Sandler, C. (2012) *The Art of Software Testing*. (3rd ed.). Hoboken, New Jersey: John Wiley & Sons, Inc.
- [36] Naik, S. and Tripathy, P. (2008) *Software Testing and Quality Assurance: Theory and Practice*. Hoboken: John Wiley & Sons, Inc.
- [37] Nidhra, S. and Dondeti, J. (2012). "Black Box and White Box Testing Techniques- A Literature Review". *International Journal of Embedded Systems and Applications (IJESA)* Vol.2, No.2 [online] Available at: <http://aircse.org/journal/ijesa/papers/2212ijesa04.pdf> (Accessed November 12, 2014).
- [38] Perfectomobile.com (n.d). *Test Automation*. Retrieved from <http://www.perfectomobile.com/solution/test-automation> [Accessed 6th November 2014].
- [39] Quilter, P. (2011). "Automated Software Testing with Traditional Tools & Beyond". *Automated Software Testing with Traditional Tools & Beyond*. P20, Vol 3 (5). [Online] Available at: www.automatedtestinginstitute.com[Accessed 23rd July 2014].
- [40] Selvam, R. (2011). 'Mobile Software Testing – Automated Test Case Design Strategies. *International Journal on Computer Science and Engineering*' (IJCSE) Vol.3.
- [41] She, S., Sivapalan, S. and Warren, I., (2009). *Hermes: A Tool for Testing Mobile Device Applications*. *Software Engineering Conference, 2009. ASWEC '09. Australian* , vol., no., pp.121,130, 14-17 [Online] Available at: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5076634> [Accessed 12th October 2014].
- [42] Sofokleous, A. and Andreou, A. (2008). "Automatic, evolutionary test data generation for dynamic software testing". *Journal of Systems and Software*.P 1883–1898, Vol 81 (11). [online] Available at: <http://www.sciencedirect.com.libaccess.hud.ac.uk/science/article/pii/S0164121208000101> [Accessed 24rd July 2014].
- [43] Stottlemeyer, D. (2001). *Automated Web Testing Toolkit: Expert Methods for Testing and Managing Web Applications*. Canada: John Wiley & Sons.
- [44] Testplant.com (n. d). *eggplant*. Retrieved from <http://docs.testplant.com/?q=content/using-eggplant> [Accessed 6th November 2014].
- [45] uTest.com (2011). *Software Security Testing. The Keys to Launching a Safe, Secure Application* [online] Available at: http://c0954852.cdn.cloudfiles.rackspacecloud.com/uTest_eBook_Mobile_Testing.pdf [Accessed 20 Septemeber 2014].
- [46] Wynne, M. and Hellosoy, A. (2012). *The Cucumber Book: Behaviour-Driven Development for Testers and Developer*. (1st ed.) North Carolina: Pragmatic Programmers, LLC.

AUTHOR

Bakhtiar M. Amen is a PhD research candidate in School of Computer Science at the University of Huddersfield. Bakhtiar holds BSc in Software Engineering and MSc in advanced computer science at the University of Huddersfield. Bakhtiar's research interests are in the areas of mobile application testing, mobile data age, cloud computing, big data and big data analytics, distributed computing, and Internet services. He has published over 4 international journal and conference papers. He is a recipient of 2013 VC Scholarship from the University of Huddersfield. He is a member of British Computer Society BCS.

