An Automated Testing Framework for Testing Android Mobile Applications in the Cloud

C.Mano Prathibhan[1], A.Malini[2], N.Venkatesh[3], K.Sundarakantham[4]
[1,2,3,4]CSE Department, Thiagarajar College of Engineering,Madurai, India
[1]manoprathi@gmail.com, [2]amcse@tce.edu, [3]venkateshtceit@gmail.com, [4]kskcse@tce.edu

***Abstract* — The testing of mobile application faces many issues due to the complexity of testing these applications and the limited resources available in mobile devices. Testing in various mobile devices under varying conditions takes a lot of time when done manually. Also by using emulators it is not possible to generate the same real time network connections and real device characteristics. There is a need for a testing framework that allows automated testing of mobile applications in many mobile devices in limited time . In this paper we propose a mobile testing framework in the cloud environment that aims to provide automated testing of mobile applications in various mobile devices. This testing framework has an automated testing tool, the Mobile Application Testing (MAT) Tool integrated to it that performs functional, performance and compatibility testing of mobile applications.**

Keywords* — *Mobile Testing, Automated Testing, Testing as a Service, Cloud Testing

I. INTRODUCTION

The evolution of mobile computing devices and mobile OS in the past years has influenced the development of various mobile applications for many needs. And these mobile applications need to be tested accordingly so that the use and purpose of these applications are satisfied. That is the performance and functionality of the mobile application should be tested in various mobile devices and OS before they are deployed. There comes the problem as testing a mobile application for compatibility is a tedious procedure when considering the real network and real mobile devices. The performance of any mobile application should be tested in more than one mobile device and the comparison is made to identify the most compatible mobile device for a particular application..

The most preferable environment for performing automated testing of mobile applications is the cloud environment [1]. The cloud has many features suitable for testing purpose. The cloud based testing framework provides an on-demand Testing as a Service (TaaS) for any software application including the mobile. The cloud based testing is most preferable since all the resources are available in the cloud and these resources can be utilized in an efficient way by making use of the various cloud services. The execution of the testing process is faster in cloud based testing than using normal testing methods.

The proposed framework is a cloud based mobile application testing framework that can be used to perform various types of testing in any given Android mobile application. This framework has an automated testing tool deployed within it. This tool integrated with the testing framework provides Android Testing as a Service (ATaaS) for the customers. The various features of this tool and the testing procedure are discussed in the later sessions of the paper.

The overall content of this paper is organized as: in Section II we discuss about the various existing mobile testing frameworks and testing tools available in the cloud; in Session III we discuss about the proposed testing framework and the services provided here; in Session IV we talk about the mobile application testing tool deployed in the testing framework and in Session V we discuss the advantages of using the automated testing framework and the future enhancements.

II. EXISTING FRAMEWORKS AND TOOLS

Many frameworks available for testing mobile application for various purposes. Each framework has its own method of testing procedure and test process. The improvement in the field of mobile OS and mobile computing devices enabled the development of these frameworks for testing. Many such frameworks exist but only the frameworks developed in the cloud environment are discussed here since testing is more preferable and efficient in the cloud. Some of the existing mobile testing frameworks are,

- The MobiTest [2] is an automated cross-platform tool that provides mobile application testing. This tool can be integrated to the cloud for testing.
- A cloud based software testing paradigm [3] proposed a mobile application testing framework in the cloud that provides various testing services.
- Testdroid [4] is an online mobile testing framework that provides UI testing for mobile applications in many devices.
- SOASTA CloudTest [5] is an automated mobile performance and touch test tool integrated in the SOASTA cloud
- A Model-Driven approach [6] that is used for automation of mobile applications using DSML.

Other than these various online mobile application testing frameworks are also available. Some of these are: BlazeMeter [7], DeviceAnywhere [8], PerfectoMobile [9], Parasoft Test [10], etc. The problems identified in these existing solutions are that they do not show any compatibility testing [11] and effective result comparison. That is they don't show the most

ISBN No. 978-1-4799-3914-5/14/$31.00 ©2014 IEEE

compatible mobile device and OS for an application. Also most of these frameworks and tools are not open source, so there is a need to pay for what is used in testing procedure.

III. Proposed Framework

The proposed testing framework is a mobile application testing framework specially deployed for testing of android applications by making use of the cloud environment. It provides Android Testing as a Service (ATaaS) that includes services such as: automated mobile application testing [11], testing application compatibility [12], performance testing [13], android testing as a service [14] and emulator testing [15]. The deployed framework is shown in Fig 1. The mobile device in which the application is to be tested is connected to the cloud testing framework using any means.

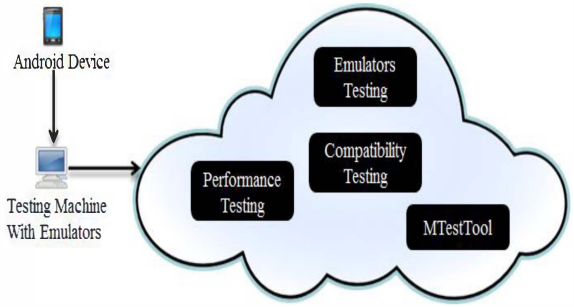

Fig 1. Overall Testing Framework

As shown in Fig 1 the Mobile Application Testing (MAT) is deployed in the cloud environment and various testing services are provided. The testing can be done in both emulator and real devices. This testing tool also compares the test results which could be served for a comparative study and verifications.

A. Android Testing as a Service (ATaaS)

The ATaaS is the main service provided by the proposed testing framework to test Android mobile applications. More than other mobile OS the need for testing Android platform is more. This is due to the availability of many versions of Android platforms such as jellybean, gingerbread, ice-cream sandwich, etc. Other than these we also have many versions of Android OS. The Android applications are developed to be compatible to all these versions and so we need to test the applications in all these versions before releasing the application in the market. To test the applications in all Android versions for different mobile devices we make use of the ATaaS service provided by the testing framework in the cloud. The various Android versions available and the percentage of these versions in the market as per August 2013 [16] are shown below in Figure.2.

From the statistics in Fig 2 we can see that the Android versions of Gingerbread, Ice-cream Sandwich and Jellybean are used more than 95% in the market. So our proposed framework tests the mobile applications in these versions alone. Also the market for Jellybean has increased from 12% to 40% since last year and it's estimated to go up in future. We give more priority for testing Jellybean applications.

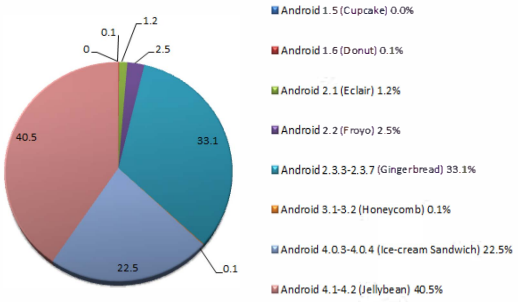

Fig 2. Different Android versions

B. Functional Testing

Functional testing [17] is a basic type of testing that is used to test each individual functionalities and features available in any application or software. To test the functionality of any mobile application we need to understand how the application works. Whichever mobile device the application runs in the function of the mobile application is always the same. For example a calculator application main functionality is to provide arithmetic calculations and it provides this same function when it is run on any type of mobile device. But still we need to test these functions if they give the exact same result that it was designed to provide.

In the proposed framework the testing tool checks if the given mobile application gives the expected result for every function available. The characteristics of the mobile application are tested. The functional testing can be done with only the APK file of the mobile application and there is no need for the source code. The functional testing is done in a black box testing method that will not be visible to the tester.

C. Performance Testing

The performance is the measure of how well an application responds to the user. That is the performance of any mobile application can be calculated by using the response time and throughput of that application with respect to the mobile device and the network latency. The response time (R) is the time taken by the mobile application to respond back to the user after providing any input and excluding the network delay. The testing tool has an inbuilt function to calculate the response time as follows,

R = (To – Ti) – L seconds/request

Where,

R → Response Time

Ti → Time at which the user request the service

To → Time at which the request service is completed

L → Network Latency (if any)

Similarly, Throughput (T) is the measure of the number of events/requests processed in a given amount of time. The value of throughput in turn depends on the value of the response time

values calculated before. The computation done by the testing tool for calculating throughput is,

T = t/Avg(R) requests/second

Avg(R) = Sum(Ri)/E seconds/request

Where,

t → Time (in seconds)

E → Number of requests

Ri → Response Time of request i

Avg(R) → Average Response Time for i Requests

The Network Latency (L) is the delay in the network under which the mobile device is connected and tested. This will be the delay available in the cloud network. The value of the latency depends both on the latency available in the mobile device and the network. The testing tool uses the following formulae,

L = (RTT/2)+D+E seconds

Where,

RTT → Round Trip Time/Delay

D → Trace route Delay

E → End point computational Delay

These values of R, T and L are the performance metrics and they can be used to compute the compatibility measure of the mobile applications with respect to the mobile device.

D. Compatibility Testing

The compatibility of any mobile application can also be tested by using the proposed testing framework. The compatibility defines how much the mobile device and platform supports the proper working of the application. It depends on various factors and features such as the application scalability, platform support, mobile device features like hardware and the software. Other than these it also depends on the performance of the mobile application. Using the performance metrics we can calculate the compatibility measure (C) by assuming certain ranges of values for each of the performance metrics R, T and L. We calculate performance measure (M) for each of the ranges in every performance metrics and the average of the values of M gives the overall compatibility measure. The value of M ranges from 1 to 10 depending on the value ranges of R, T and L. We take the value of M from 1 to 10 because this can be converted into percentage to find the compatibility percent. The compatibility function in the testing tool makes use of the following,

C = (MR + MT + ML)/3

Where,

MR → Performance Measure of Response Time

MT → Performance Measure of Throughput

ML → Performance Measure of Network Latency

IV. MAT TOOL DESCRIPTION

The MAT Tool is the automated testing tool that is integrated with the proposed framework in the cloud. All the services and testing provided by the testing framework are supported by the MAT tool. The basic architecture of the MAT Tool is shown in Fig 3 as follows:

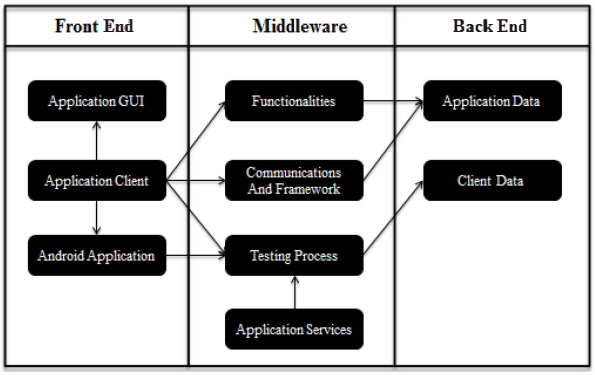

Fig 3. MAT Tool Architecture

The testing tool enables an easy to use GUI for the customer that helps him to perform the testing with ease. The testing process described in the architecture is the important module of the testing tool. All the process and calculations are performed here and the application services of the testing tool are provided here with the help of cloud. The testing services and process can be accessed by the user by using the MAT Tool. The test data are stored in the user profiles and can be accessed anytime.

The whole testing framework discussed in the previous session depends on the MAT Tool. This tool acts as an interface between the user and the cloud. All the services provided by the testing framework are provided through this tool. The testing process takes care of the full procedure of the testing that includes the calculation of performance metrics and the compatibility measures. That is the procedure for implementing the functional, performance and compatibility testing are available within the testing process of the MAT Tool. The flow of the testing process is shown in Fig 4.

The Android application is given as the input for the MAT Tool as an APK file. So there is no need for the application source code to perform testing. After the application is uploaded to the cloud using the tool we can start the testing process. The functionality can be tested for each available functions of the mobile application. To do performance testing we need to run the mobile application in either an emulator available in the tool or by connecting a real mobile device to the testing framework. Once the application is executed the user gives the input in the form of GUI events and they are recorded by using a crawler. The test cases are generated and executed automatically by the tool and the performance results are displayed to the user.

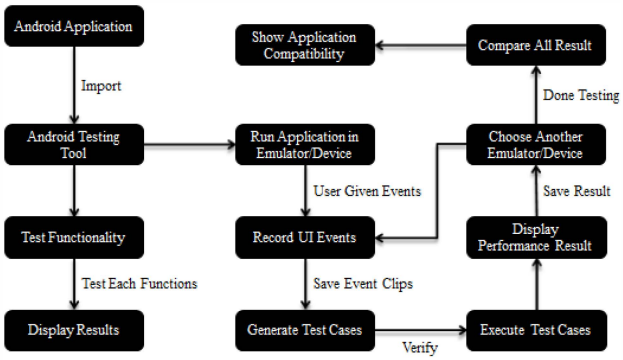

Fig 4. The Testing Process

The performance testing results include the measure of response time (R), throughput (T) and the latency (L). To perform a compatibility testing we can do the performance testing of the mobile application in various emulators or devices in a repeated process and thus by finally computing the compatibility measure (C) for every emulator or device. Then we can compare these results to find the most compatible mobile device for that application.

V. Advantages of MAT Tool and Framework

The testing framework has many advantages same as that of all the other existing frameworks in the cloud. The advantages of using the cloud [18] are more when it comes to testing cost, testing time and the scalability of testing. This testing framework can be deployed in any type of cloud to provide Android Testing as a Service (ATaaS). Apart from these the features of the MAT Tool provides various other advantages.

The MAT Tool makes use of only the APK file of the mobile application so we don't need to worry about the source code of the application. Also the testing process is automated completely to produce detailed results. The compatibility testing of an Android application in various platforms and devices is tested for the first time in this framework. Moreover we can perform testing in both real devices and emulators and can compare these results also.

VI. Conclusion

The testing of mobile applications faces many critical issues due to various reasons and complexities. In this paper we proposed an automated testing framework in the cloud for testing android applications. This framework makes use of the MAT Tool to automate mobile application testing. We can do functional testing, performance testing and compatibility testing in Android mobile applications. The deployment and implementation of the testing framework are discussed here and the various testing services provided are explained. This framework is unique for only testing Android mobile applications as it provides Android Testing as a Service (ATaaS) for the customers.

In future we plan to integrate more testing such as the interruption testing and load testing for mobile applications with this testing framework. Also it is possible to extend the proposed testing framework by adding more features and reducing the testing process by enhancing the MAT Tool architecture.

References

[1] Koray Incki, Ismail Ari, Hasan Sozer, "A Survey of Software Testing in the Cloud", IEEE 6th International Conference on Software Security and Reliability Companion, June 2012.

[2] Ian Bayley, Derek Flood, Rachel Harrison, Clare Martin, "MobiTest: A Cross-Platform Tool for Testing Mobile Applications", The 7th International Conference on Software Engineering Advances, 2012.

[3] Srikanth Baride and Kamalesh Dutta, "A Cloud based Software Testing Paradigm for Mobile Application", ACM SIGSOFT Software Engineering Notes, Volume 36, Issue 3, May 2011,

[4] Jouko Kaasila, Denzil Ferreira, Vassilis Kostakos, Timo Ojala, "Testdroid: Automated remote UI testing on android", ACM Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia, Article No. 28, December 2012

[5] http://www.soasta.com/

[6] Youssef Ridene, Franck Barbier, "A model-driven approach for automating mobile applications testing", ACM Proceedings of the 5th Internationsl Conference on Software Architecture: Companion Volume, Article No. 9, September 2011

[7] http://blazemeter.com/

[8] http://www.keynotedeviceanywhere.com/

[9] http://www.perfectomobile.com/

[10] http://www.parasoft.com/jsp/products/soatest.jsp?itemId=101

[11] Liu Zhifang, Liu Bin, Gao Xiaopeng, "Test Automation on Mobile Devices", ACM Proceedings of the 5th Workshop on Automation of Software Test, 2010, pp. 1-7

[12] Tai-hoon Kim, Hojjat Adeli, Haeng-kon Kim, Heau-jo Kang, Kyung Jung Kim, Akingbehin Kiumi, Byeong-Ho Kang, "Mobile Application Compatibility Test System Design for Android Fragmentation", Software Engineering, Business Continuity, and Education, 2011, pp. 314-320.

[13] Heejin Kim, Byoungju Choi, Seokjin Yoon, "Performance testing based on test-driven development for mobile applications", ACM Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication, January 2009, pp. 612-617

[14] http://www.vogella.com/articles/AndroidTesting/article.html

[15] http://51degrees.mobi/Support/MobileEmulators.aspx

[16] http://developer.android.com/about/dashboards/index.html

[17] Youssef Ridene, Nicolas Belloir, Franck Barbier, Nadine Couture, "A DSML for mobile phone applications testing", ACM Proceedings of the 10th Workshop on Domain-Specific Modelling, Article No.3, 2010

[18] A.Vanitha Katherine, K.Alagarsamy, "Conventional Software Testing Vs. Cloud Testing", International Journal Of Science & Engineering Research, Volume 3, Issue 9, September 2012.