Performance Enhancement of Android Application Testing using Android Devices as a Service Cloud Model

Dr. N.N.Pise
Associate Professor
Department of Computer Engineering
Maharashtra Institute of Technology
Pune, India
nitin.pise@mitpune.edu.in

Prasad P.Purnaye
Assistant Professor
Department of Computer Engineering
Maharashtra Institute of Technology
Pune, India
prasad.purnaye@mitpune.edu.in

Malay Dube
Department of Computer Engineering
Maharashtra Institute of Technology
Pune, India
malaydube196@gmail.com

Mratyunjay Prasad Mishra
Department of Computer Engineering
Maharashtra Institute of Technology
Pune, India
mratyunjay.078@gmail.com

Manas Pandey
Department of Computer Engineering
Maharashtra Institute of Technology
Pune, India
manap208@gmail.com

Rishi Jain
Department of Computer Engineering
Maharashtra Institute of Technology
Pune, India
rishi.j21@gmail.com

Abstract—The recent spurt in Android devices has given rise to the development of millions of Android applications. Hence there is a need for rapid and efficient testing techniques to satiate the global app demand. Hence the paper proposes an automated testing framework by deployment testing practices on the cloud. Android devices could be tested by connecting them through wired or wireless connections. This framework can be addressed as Android Testing as a Service (ATaaS) which assists in carrying out functional testing, performance testing etc. The paper also demonstrates the use of a MAT tool to carryout automated testing, the results of which can be compared to conventional testing practices. It also gives us a gist of virtualization. Hence Cloud Testing Platform (CTP) aims at providing all the conventional testing practices at higher speed, availability and lower cost of testing these applications and the limited resources available in mobile devices.. Developers can conduct compatibility tests in various Android devices only by uploading test programs and scripts to cloud testing platforms.

Keywords*—Android Testing, Cloud Computing, ATaas, Virtual Machine.*

*****

I. INTRODUCTION

In the past years, mobile devices have seen an ever increasing demand in market .So the Mobile application developer and tester have to keep up with the change that comes with the variation of OS version and diversity of devices. Mobile application testing is process by which application software developed for hand held mobile devices is tested for its functionality, usability and consistency. There are various mobile app testing approaches such as testing on real device, simulator, emulator and testing on cloud. The most preferable environment for performing automated testing of mobile applications is the cloud environment.

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 Cloud Testing Platform (CTP) [10] is aiming to provide similar testing services as those aforementioned. It is an open-source cloud-based system specifically for testing Android apps. It allows users to perform varied tests for an Android app on a set of user-selected devices in parallel over cloud to examine if the app can run well on the devices. To test an Android app across multiple devices concurrently, each selected device has a corresponding virtual machine (VM) allocated in CTP to interact with the device for executing the tests.

II. RELATED WORK

A. Open Stack:

Open Stack allows users to deploy a private cloud and storage environment Open Stack consists of a computing module, networking module, and storage module and is collocated with a dashboard for centralized management, thereby creating an Open Stack shared service. In addition, it provides VMs and offers computing resources externally, enabling flexible expansion or adjustment. Developers of applications can communicate with Open Stack through an application programming interface (API), such as adjusting the number of VMs and turning on and off the VMs. Internally, Open Stack manages computing resources and the communication and coordination between components.

B. Virtualization:

Virtualization describes a technology in which application, guest operating system or data storage is abstracted away from the true underlying hardware or software. A key use of virtualization technology is server virtualization, which uses a software layer called a hypervisor to emulate the underlying hardware. This often includes the CPU's memory, I/O and network traffic. The guest operating system, normally interacting with true hardware, is now doing so with a software emulation of that hardware, and often the guest operating system has no idea it's on virtualized hardware. While the performance of this virtual system is not equal to the performance of the operating system running on true hardware, the concept of virtualization works because most guest operating systems and applications don't need the full use of the underlying hardware. This allows for greater flexibility, control and isolation by removing the dependency on a given hardware platform. While initially meant for server virtualization, the concept of virtualization has spread to applications, networks, data and desktops.

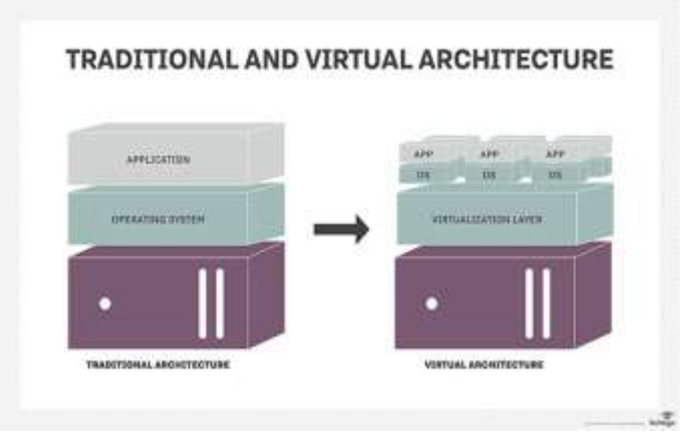

Figure 1 Traditional and Virtual Architecture

C. Android Testing 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 , testing application compatibility , performance testing , android testing as a service and emulator testing . 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.

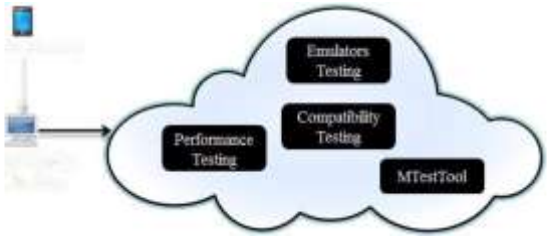

Figure 2 MAT TOOL

As shown in Fig 2 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.

III. SYSTEM DESIGN

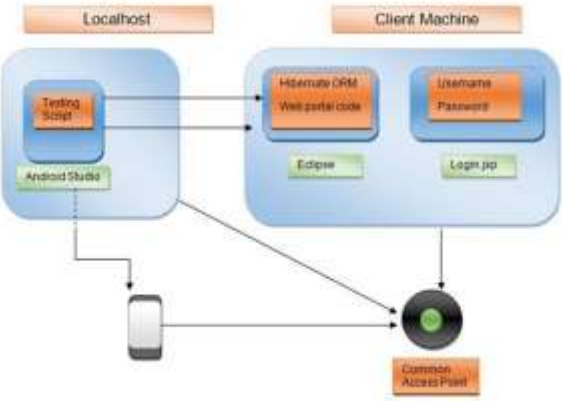

Figure 3 System Architecture

A. Cloud Service

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Private Cloud

Private cloud has different meanings to different people but basically, it's a cloud infrastructure set up, managed ,and upgraded by an organization or community for their own use. Commercial vendors are entering in this domain quite fast and Open Source providers are also there: Eucalyptus, Open Nebula, open Stack.

Steps for Setting Up Private Cloud

- Adopt a machine virtualization strategy
- Profile application compute ,memory, and storage usage and performance requirements
- Design a virtual machine development consultancy
- Accounting and recharge policies adapted to self-service
- Architect a deployment and deploy a private e cloud

To achieve dynamic resource adaption and load balancing, the design and queue structure of CTP have been refactored to support the features of dynamic VM and workload adjustment. This architecture is composed of three subsystems: Main, Rabbit MQ , and Open Stack. The Main subsystem is responsible for managing and controlling the Android app testing. The Rabbit MQ is a widely used open source software system for exchanging messages across applications. It is used to provide job queue management for CTP. The Open Stack is a well-known open source cloud computing software system that consists of a set of tools for building and managing cloud platforms. It is used to provide could environment for CTP. The Resource Scheduler is mainly used to control and monitor the varied device resources of CTP for scheduling test jobs. It will interact with the Build Job Handler which is used to build test jobs and dispatch the jobs to corresponding queues in RabbitMQ on the first-come, first-served basis. In addition, the Resource Scheduler will invoke the Job Balancer that is responsible for balancing the workloads across multiple devices. Moreover, the Resource Scheduler will also interact with the IS Monitor that is responsible for the management of Integration Station (IS) and VM resources. The Integration Station is a component of CTP executed on the VM. Basically, the IS will retrieve a test job from the queue of RabbitMQ and interact with device of request to perform the test job, collect the test result, and put the result into RabbitMQ for further processing. The IS Monitor will be in charge of the VM deployment, starting, stopping, restarting, removal, and deletion. Moreover, the VM images are managed through using OpenStack APIs.

B. Mat Tool

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

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.

IV. Future Work

In this project we are designing a system that generates test cases for single android application of different devices using highly efficient testing procedure (public cloud, PaaS) which should bring a considerable change in the current testing scenario of android application.

Mobile application automation testing has a very vital scope in the field of software testing. Mobile application testing is considered as one of the important testing practice, as day by day, we see drastic growth in mobile app development, i.e. each day, lots of mobile apps are developed, which could be for Android .Mobile test automation provides various test solutions, which help you to verify and validate the overall functioning of the mobile apps. Today, Automated software testing is considered as one of the most trending term in the field of software testing, as it is considered as one of the efficient as well as effective software methodology, making the overall software development/testing life cycle better and taking it to some different level.

REFERENCES

[1] OpenStack Foundation. (2014). “OpenStack APIs,” Available: http://docs.openstack.org/api/quickstart/content/index.html#Compute_API_Quick_Start

[2] OpenStack Foundation. (2014). “Releases,” Available: http://wiki.openstack.org/wiki/Releases.

[3] “Cloud Computing in Smart phone: Is offloading a betterbet?”, Milindkumar H. Tandel, Vijay S. Venkitachalam,CS837-F12- MW-04A

[4] Xiaoqiang Ma, Yuan Zhao, Lei Zhang, and Haiyang Wang, Limei Peng,”When Mobile Terminals Meet the Cloud: Computation Offloading as the Bridge”, 2013 IEEE

[5] J. Andrus and J. Nieh, “Teaching Operating Systems using Android,” in *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education (SIGCSE ’12)*, 2012, pp. 613–618.

[6] P. A. Brooks and A. M. Memon, “Automated GUI Testing Guided by Usage Profiles,” in *Proceedings of the Twenty-second IEEE/ACM International Conference on Automated Software Engineering (ASE ’07)*, 2007, pp. 333–342.

[7] R. Mahmood, N. Esfahani, T. Kacem, N. Mirzaei, S. Malek, and A. Stavrou, “A Whitebox Approach for Automated Security Testing of Android Applications on the Cloud,” in *Proceedings of the 7th International Workshop on Automation of Software Test (AST ’12)*, 2012, pp. 22–28.

[8] Android Developers. The Developer’s Guide. Available at: http://developer.android.com/.

[9] Android Emulator, available at: http://developer.android.com/guide/developing/tools/emulator.html

[10] H. Kim, B. Choi, W. Eric Wong. Performance Testing of Mobile Applications at the Unit Test Level. Proc. of 2009

[11] Z. Liu, X. Gao, X.Long. Adaptive Random Testing of Mobile Application. Proc. of 2nd International Conference on Computer Engineering and Technology (ICCET), IEEE Comp. Soc. Press, 2010,pp. 297-301

[12] A.Wasserman. Software Engineering Issues for Mobile Application Development. Proc. of the FSE/SDP workshop on Future of software engineering research, FOSER ,IEEE Comp. Soc.Press,2010,pp.397-400.