p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 10
June 2016

Dual Sim Platform Automation on Android Phones

Tejashree M S
M. Tech Student, Dept. of IS&E,
Sri Jayachamarajendra College of Engineering,
Autonomous under VTU, Mysore, Karnataka, India

Prathibha R J
Assistant Professor, Dept. of IS&E,
Sri Jayachamarajendra College of Engineering,
Autonomous under VTU, Mysore, Karnataka, India

ABSTRACT: In the era where mobile is dominating all other domains and almost every fortnight a new mobile phone is launched. Mobile Testers are under a huge pressure when it comes to manual testing. The solution is Automation.

The objective is to automate DSDS (Dual SIM, Dual Standby) functionality on android phones. This would involve developing and designing test suites/framework, enhancing existing test cases to support various platforms running Dual SIM support on Android Phones. The main focus is to automate the testing of dual sim functionality on given any product and to validate the software by give the verdicts of Pass or Fail.

KEY WORDS: DUT – Device under test, AT – Automation Tool, DSDS – Dual Sim Dual Standby, TS- Testing System, ADB - Android Debug Bridge, AAT – Android Automation Tool,

INTRODUCTION:

In today´s fast moving world, it is a challenge for any company to continuously maintain and improve the quality and efficiency of software systems development. In many software projects, testing is neglected because of time or cost constraints. This leads to a lack of product quality, followed by customer dissatisfaction and ultimately to increased overall quality costs.

Automated tests can run fast and frequently, which is cost-effective for software products with a long maintenance life. When testing in an agile environment, the ability to quickly react to ever-changing software systems and requirements is necessary. New test cases are generated continuously and can be added to existing automation in parallel to the development of the software itself.

In the era where mobile is dominating all other domains and almost every fortnight a new mobile phone is launched. Mobile Testers are under a huge pressure when it comes to manual testing. The solution is **Automation.**

Android is a mobile operating system (OS) based on the Linux kernel and currently developed by Google. With a user interface based on direct manipulation, Android is designed primarily for touchscreen mobile devices such as smartphones and tablet computers. The OS uses touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, and a virtual keyboard. Despite being primarily designed for touchscreen input, it has also been used in game consoles, digital cameras, regular PCs, and other electronics. As of 2015, Android has the largest installed base of all operating systems.

A **Dual SIM** mobile phone is one which holds two SIM cards. Initially, dual-SIM adapters were made available to use in regular mobile phones to allow them to contain two SIMs, and to switch from one to the other as required. This combination is called a standby dual-SIM phone.

Dual SIM Dual Standby - These types of dual SIM devices are abundant in Indian Android Market and are present over a wide price range. As the name suggest these type of phones support both SIMs in Standby mode. That is to say until you are not using any of your SIM cards, both are active and on standby. But once you start using either SIM1 or SIM2 the other one becomes inactive. If someone calls you on the other SIM he will hear a not reachable message from your service provider. This can be a graver problem if you use your secondary SIM to make long duration calls, because it will render your primary SIM inactive and make you not reachable.

LITERATURE SURVEY

The literature related to the research topic has been reviewed for last few years in order to find out work carried out by various researchers. It is noticed that most of the research carried out belongs to the following categories:

1. Dual SIM Android Platform Creation

This project is meant to address the AP side architecture of product Dual SIM Dual Standby platform. This platform supports two GSM/WCDMA modems. But, only one WCDMA will be in service while the other modem has to be in GSM mode (G+W DSDS). This will cover telephony application, framework and other necessary modules. This is used to define Product's DSDS telephony overall architecture, component design and UI design for use cases.

However this project was more focused on AP side creation and the automation related work was minimum which is more focused in this project.

2. High Level Design Document AP side of Product Platform – DSDS Telephony System

Creating a full featured Dual SIM Dual Standby (DSDS) Android Phone has a key advantage –it is required for emerging markets like China and India. This discusses the technical realization of the DSDS feature on Android platform. Aims at availability of this framework allows to roll out all smartphone and tablet designs with this feature for all Android based platforms, solution is transparent to all the default Android applications/services and easily portable across different Android versions.

This project mainly focused on framework of DSDS Telephony system, they did not give a generic solution for automation of dual sim.

3. Dual Sim Dual Standby Steps to achieve DSDS functionality in Android base

As the name implies, DSDS provides a Dual SIM user the ability to keep both the SIM active for incoming call, but only one for outgoing call (thus one is in "Standby" mode). The project has been implemented for Android devices and has been tested with the latest Android ICS code base. DSDS extends the capability of Android to dual-SIM phones, with the benefit of using both the SIM at any point of time. Here the user decides which SIM he wants to keep active for outgoing calls and he can swap to the inactive SIM anytime. As mentioned above, both the SIM are active for the incoming call, thus allowing the user to use both the SIMs effectively. DSDS has been implemented with keeping the features of Android intact and giving it all the benefits of the Android system with addition to the benefits of DSDS.

This paper aimed at steps to achieve DSDS functionality. No specific solution was given for automation by using android API's and 4G functionality testing scenarios.

METHODOLOGY

In order to accomplish this, the following method is carried on:

The development of python scripts where the actual functions will be performed.

- Each python script has 3 parts: Setup (initialization), Run (execution), Teardown (de-initialization)
- AT calls these three methods in sequence as the script executes
- Reports & Logs are generated at test case as well as campaign level
- On completion of execution, a consolidated report containing verdicts of all the test cases is generated

The development of Test Engine APK helps establish a TCP connection between the host PC and the Device under Test (DUT).

- It has the implementation of the APIs that are called in the python script.
- These APIs in turn invoke default Android APIs in their implementation

The development of GEH serves as the control point for the campaign. Various functions of the GEH are listed below:

- Initialization and termination of individual test cases and overall campaign
- Installing of APKs, start/stop of ADB server, logging mechanisms
- Handling DUT crashes/freezes, core dumps
- Responsible for DUT recovery in case device goes offline
- 'Flashing' has also been implemented but is optional.

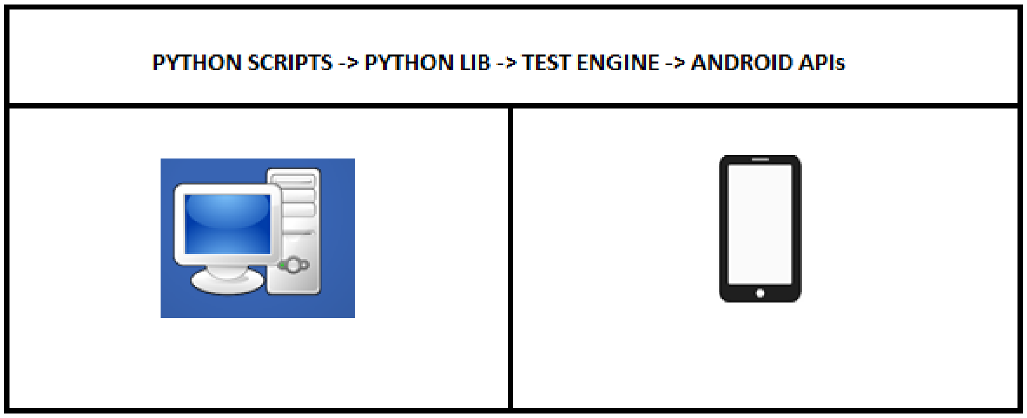

Figure 1: Work flow of Project Methodology

IMPLEMENTATION:

1. UI Automation

The UI Automaton testing framework provides a set of APIs to build UI tests that perform interactions on user apps and system apps. The UI Automaton APIs allows you to perform operations such as opening the Settings menu or the app launcher in a test device. The UI Automaton testing framework is well-suited for writing black box-style automated tests, where the test code does not rely on internal implementation details of the target app. User interface (UI) testing lets you ensure that your app meets its functional requirements and achieves a high standard of quality such that it is more likely to be successfully adopted by users.

To automate UI tests with Android Studio or Eclipse you implement your test code. The tool builds a test app based on your test code, then loads the test app on the same device as the target app. In your test code, you can use UI testing frameworks to simulate user interactions on the target app, in order to perform testing tasks that cover specific usage scenarios.

UI Automator Viewer: The uiautomator viewer tool provides a convenient GUI to scan and analyze the UI components currently displayed on an Android device. You can use this tool to inspect the layout hierarchy and view the properties of UI components that are visible on the foreground of the device. This information lets you create more fine-grained tests using

UI Automator, for example by creating a UI selector that matches a specific visible property. The uiautomator viewer tool is located in the <android-sdk>/tools/ directory.

2. Automation Tool

Automation Tool (AT) is the general test automation framework. It is primarily used for the automated verification of mobile platforms and its components. Users implement test cases in AT to test & validate their system under test. AT is the test engine in Stability Rack (MTBF Testing) and TS tests. It is integrated with AT, thus enabling re-use of existing automation tool chain (AT, TS) to be extended for verification of Android platforms. AT-AAT offers testing in various areas such as Sanity, Stress, System, Feature Interaction, Random scenarios, Component testing. This integration allows for sub-system testing to be performed under TS e.g. Power Drivers.

3. Android Automation Tool

Android Automation System should match the needs for automatic testing off all parties involved: Android Developers, Android QA and Android Distribution. Android Automation Tool (AAT) is a host based automation test setup for verifying Android based platforms. It's a collective term used to refer to a set of host based python scripts, target based Test Engine and Android applications collectively used to achieve Android Test Automation.

As part of tools integration, AAT has been integrated with an in-house Modem testing tool, Automation Tool (AT), thus providing a single integrated tool to test both Modem-Only and Android platforms. AT serves as a vehicle for execution of the automation scripts. Additionally, the tool leverages Legacy Modem Tracing, Lab & TS setup capabilities.

4. Integrating AT with TS

Testing System is application software and physical infrastructure that enables quality gates and test campaigns to run on target devices. It evolved from ARTS which is the part of OptiCM5 that handles automated tests (replaced by T-Cloud in OptiCM6).It is an integral part of the Continuous Integration workflow for development community. Runs in Pre-Commit Quality Gates, Post-Commit Quality Gates, manual & other scenarios, operates at a large scale with hundreds of deployments worldwide.

The TS IO Lite is a hardware switch board used to connect/disconnect USBs in the modem, power lines to the modem and Button press on the modem. The HIL board is the heart of the HW setup in TS. USB cables from PC and power connectors from power supply goes to the modem board via HIL board which is controlled from PC via USB by sending control information to turn ON/OFF particular lines. The HIL hardware consists of a switch board mounted on top of a micro-controller board (Arduino Mega 2560). Arduino is an open-source electronics platform.

TEST RESULTS AND OBSERVATION

The testing result of these test cases are validated by comparing TS verdict and manual testing results. If the test fails in manual testing and passes in our automation or the other way pass in manual testing and fails in automation then the system is considered as failure. Both the results should match each other for a set of scenarios and multiple iteration, then this system can be considered as working. These tests are used by Integration team, Verification team, MTBF teams and many others to validate the latest software functionality as a basic check.

If these results are not stable then manual sanity team is involved to check the software and to handle the software. But before this the test cases are done regression testing with multiple scenarios to make sure if the test works fine in all possible scenarios. Both Pass and Fail cases are made to run multiple times with working and non-working software and then if the test cases passes in all cases then they are integrated with TS for daily mainline health checkup of the device with the latest software with new functionalities.

This project mainly looks into the dual sim functionality of the device, hence we have tested multiple iteration on the test cases that validates the functionality and can come to a conclusion if the latest software is good enough to integrate with mainline and to deploy them. The main focus here is on dual sim calling, dual sim messaging and browsing. Test cases are written to perform both Sim1 and Sim2 functionalities. Sims are switched to 4G, 3G and 2G network registrations and all the scenarios the dual sim functionalities are tested.

REFERENCES

1. http://developer.android.com/tools/testing-support-library/index.html
2. https://en.wikipedia.org/wiki/Microsoft_UI_Automation
3. http://developer.android.com/training/testing/ui-testing/uiautomator-testing.html
4. https://opensource.intel.com/how-to/quality-assurance/qa-architecture-forum/android-test-automation#ATF
5. http://developer.android.com/reference/android/support/test/uiautomator/UiScrollable.html#scrollDescriptionIntoView(java.lang.String)
6. Dual Sim Dual Standby Steps to achieve DSDS functionality in Android base. - http://www.irma-international.org/viewtitle/63846/