Multi-Game Automation Approach For Cocos-2dx Based Card Game

Akshay Kumar H[1], Sarath Kumar Purushothaman[2], Viral Singh [3], Abhishek Bharti [4], Nishant Kashyap [5]
Play Games 24x7 Private Limited,
Mumbai- 400 064, India
[1]akshaykumarh94@gmail.com, [2]sarathkmr21@gmail.com, [3]viralsingh@gmail.com, [4]abhishek.manit@gmail.com,
[5]nishantkashyap@gmail.com

***Abstract*—Automation in testing is crucial for bug discovery and enhancing the quality of the software product. Automation also saves time and helps delivering the product faster. Automating a game play in a card based game is challenging due to the indefinite cards which are served during game play and also due to the fact that the dynamic logic should be applied according to the type of card one receives during game play. A successful approach of building a test automation framework for automating the game play of a single instance of a card based game was achieved previously.[1] This test automation framework is further enhanced to achieve the objective of automating the game play of multiple games of a single player i.e,. a player playing multiple instances of the card game simultaneously. The automation is demonstrated in automating the game play of Rummy. However, this approach can be adapted to automate any card based game. The unique approach is discussed and evaluated in achieving this objective.**

***Index Terms*—Multi-game automation; Card based game; Cocos-2dx; Test automation framework**

I. Introduction

Test Automation is the process of developing software to control the execution of tests and developing tests which compare the generated output of a given system with the expected output in order to evaluate the correctness of its execution. Automation is generally done on tasks which are repetitive. It could also be used for performing tasks which are not possible manually.

A Test automation framework is an integrated system which defines the procedure for automating a product. It is a system containing libraries, reusable modules and test cases which work in tandem to verify the correctness of a software product and aid in achieving better quality and stability.

Testing in games mostly involves manual approach due to its ad hoc nature. Automating common tasks like joining the game, verifying the occurrence of elements, etc., can be achieved. However, automating the way in which any human player could play a game has always been a challenge. Some of the challenges involve:

- Ad hoc nature of the game. Different games may involve different logic and strategies.
- A game may involve many components and services which would work in a co-ordinated fashion. Automating all of these components and services and achieving the co-ordination between them could be challenging.
- The plethora of scenarios in the game play with different combination of elements that the given game features could be tedious to cover.

The game automation framework developed manages to overcome many of these challenges. It is a custom designed framework built on top of Selenium and Appium. Few of features worth mentioning are:

- Automation of the game works for different resolutions of the game and different devices.
- Automation of the game play covers majority of the scenarios to play the game successfully.
- The average time to successfully play a game is 2 minutes.
- The automation framework logs every action, takes screen shots in case it is not able to proceed further, thereby making debugging convenient.
- The same tests run for different game variants (pool rummy, deals rummy, points rummy) and different number of players (2 and 6) without any modification in the code.
- The tests are also independent of the channels in which the game runs. The different channels are desktop website, mobile website and Android app.

II. Literature Survey

The following are existing Test frameworks:

A. Espresso Framework

It is an automation framework based on instrumentation mainly for android devices. It is mainly based on Java and Junit. Espresso compiles all the test cases as a separate APK which could be installed on the target device along with the application. It executes methods in a synchronized way due to which the stability of the tests are very high. It has a built in mechanism of finding the objects which are being displayed on the screen before proceeding to the test cases. This avoids the commonly occurring ”object not detected” and other errors occurring with identifying elements or objects.[2]

Some of the main features are as follows:

- Execution of tests does not interfere with the actual code.
- Avoids element identification errors.

B. TestDroid

It is an automation framework developed by Bitbar. It can be used to execute tests on real Android and iOS devices. Following are some of its features:

- It features a built in Appcrawler which identifies common UI elements and performs the required actions on them like clicking a button or changing the menus present. In addition, it logs actions, takes screen shots if necessary and tracks the status of the application.
- The user can also upload his own tests in the form of a file (.apk, .zip, .jar etc) depending on the type of automation framework that he/she choses to use.
- In addition, certain advanced settings can be specified such as how the test cases can be executed on the devices (simultaneously on all devices, one device at a time or the first available device.). Different language preferences can also be selected. App authentication can also be done by providing the credentials in Testdroid.
- It provides detailed test reports such as CPU consumption, memory usage, logs of test execution and device behavior during the test run.
- It also allows manual intervention to control the remote devices.[3]

C. Calabash

Calabash is another automation framework which serves to automate android and iOS devices.It is currently maintained by Xamarine. The following are some of its features:

- Easy to use as anyone, even those without coding skills can automate.
- It is mainly used for automated UI acceptance testing as it automates common UI interactions like pressing a button,inputting text and validating responses very well.
- Easy to integrate with Continuous Integration(CI) / Continuous Development(CD) tools.[4]

D. UI Automator

It is a test automation framework build by Google to aid in cross-app functional UI testing. It has the following features:

- It is a lightweight framework.
- The test code written in it does not depend on any application code.
- It has a feature rich API which can handle common UI interactions.
- It can be seamlessly integrated with android applications to enable UI automation.[5]

An existing game automation framework is:

E. Build Verification System

Is an automation framework built to test the league of legends game at the Roit Games Engineering.It is a framework primarily designed for the game client and the server. It approximately contains 100,000 tests, which runs every hour to make sure that less bugs are shipped to the customers when it is delivered.It runs on the Continuous Integration system and provides the results within an hour of code check-in. It acquires the artifacts required to run the tests, deploys them to a test machine ,starts these machines, executes the tests and generates the reports. It makes use of remote procedure call which is present in the client and the server in order to issue commands and monitor the game state.Once the test execution is completed, the reports are stored in an archive containing the history of reports since six months. When the tests are run locally, a web page opens up showing the number of tests that has passed and failed. If the tests are run on a test farm, new bug tickets are created for discovered issues and sends emails to all those people who had written the source code in case the tests fail.[6]

The following are some of the features:

- Fast generation of test results.
- Automatic issue or bug reporting.
- Automatic email generation to the respective developers regarding the bugs.
- Maintains an archive of test reports.
- Beautified test reports.

It is observed that TestDroid[3], Calabash[4], UI automator[5] primarily help in UI automation in general, while Build Verification System(BVS) [6] checks certain features of the game play for the game League of Legends. BVS also automates for a single instance of the game. The proposed system automates entire game play from the UI with help of messages from the server (or APIs) for multiple games. It is not a generic UI automation framework but is customized to automate the game play of Rummy. It also automates the basic operations, thereby performing a sanity like joining a game according to the user's preference and playing that preferred game completely until a valid declaration is made. The existing systems are either a generic UI automation framework or they automate a particular feature in the game like the BVS. In addition, the BVS framework may not automate an entire game play. The Following are the unique features of the proposed automation framework:

- The game play is automated using a unique approach in which the messages from the game engine are exposed via a web socket which is accessed with the help Java Script hooks. These hooks return a JSON message which provides the required information to determine the state of the game.
- The actions during the game play are also performed with the help of the Java Script hooks or functions. Each of the actions to be performed on the game table or during game play is associated with a hook/function.
- Each of the Java Script hooks/functions returns a JSON message in the case when some information about the game is requested or they return true or false when some action is performed during the game play. A True value indicates that the action was performed successfully.
- Since each of these actions/message requests are done on different games, therefore a unique ID of the game is appended to these hooks before being sent to the server