THE DEVELOPMENT OF MOBILE APPLICATIONS AND OPEN-SOURCE FRAMEWORKS FOR TESTING

Olga Ristić[1], PhD; Vlade Urošević[1], PhD;
[1] Faculty of Technical Science, Čačak, SERBIA, olga.ristic@ftn.kg.ac.rs, vlade.urosevic@ftn.kg.ac.rs

Abstract: *Today, a large number of people have mobile devices, so the development of mobile applications is becoming more intense. Since there is huge demand for mobile apps that need to be tested to provide good quality. Some companies lose customers and money due to poor quality of mobile applications. Testing of mobile apps are the most difficult task due to its varieties and different operating systems. Although there are simulators and emulators available but they only simulate the working of operating system and cannot test the core functionalities for the mobile device. There are many different open-source frameworks for testing mobile apps and in this paper will be presented three most popular (Appium, Robotium and Solendroid). Here are given the advantages and disadvantages of these tools and the ability to apply on different platforms. Depending on the usage and complexity of mobile applications, the selected framework for testing will also depend.*

Keywords: *mobile app, testing, open-source*

1. INTRODUCTION

The first iPhone device and operating system (OS) for smart phones was created in 2007. Thus, ordinary mobile phones were replaced soon by smart phones with the operating system and highly developed hardware, so now it have almost all functionality of a personal computer. Large number of mobile apps was created until now and used by many people. There are a number of different OS for mobile devices, but most are now distributed Android and iPhone OS. Smart phones with Android OS most prevalent (about 80%) in comparison with other [1]. It is believed that in the future, the number of mobile apps increase significantly, and that is the main reason for development tools that are used to test mobile apps. With the increased use of mobile devices in learning, surfing, playing games, GPS, etc., smart phones have become an important part of our lives and their number are increase from year to year. This is the main reason for the rapid development of operating systems for different types of smart phones as well as increasing the number of apps for mobile devices.
The development of mobile apps is becoming one of the most important areas of IT industry. It is therefore essential that apps are tested, in order to avoid errors and problems in their functionality. In recent years, several million mobile apps have been developed and all apps have to be tested. Many IT companies are developing open-source frameworks for testing mobile apps. The main objective of testing mobile and web apps is the use of test methods and tools to ensure the quality of the work, performance, functions, features, privacy, security, mobility, connectivity, reuse, etc.
Traditionally, software testing can be performed manually and automatically, and this method can be applied to mobile apps. Manual testing performed by people, and for automatic testing are developed different software that is used to generate test cases, execution and verification. So, many software companies engaged in the development of tools for testing mobile apps with different capabilities and characteristics. There are a large number of open-source software for testing, as well as the commercial version [2].

2. DEVELOPING MOBILE APLICATIONS

The most widespread operating systems for mobile phones are:

- Andrioid (Google),
- iOS (Apple),
- Windows (Microsoft).

In the world, currently the most used are Android OS (Figure 1). With the advent of Android OS 2010, the distribution was growing to 72.88% in 2017 and the Apple iOS is using about 19.37% [3]. The present mobile phones are

increasingly used because they are cheaper, which contributed to the development of a large number of mobile apps compared to iOS mobile phones.

Apps that are created for mobile devices such as mobile phones and tablets can be divided into three main groups:

1. native apps,
2. web apps and
3. hybrid apps.

The differences between these types of apps are as follows:

- Native app must be downloaded and installed only on certain OS. They can be used in the installation on a mobile device in offline mode. This app are used for informational and productivity purposes. The most used are calendar, contacts, calculator, email, mobile games, GPS and weather information.
- Web apps are apps that can be accessed on the Internet by using the appropriate browser. Download and installation of these types of apps is not necessary, because the apps are on the web server and can always be used when accessing the Internet. The number of android app are growing so fast (Fig. 2). The main reason is that a large number of people have Android smartphones, so they can use Android apps.
- Hybrid apps are a combination of native and web apps. They are usually created using standard web tools such as HTML, CSS and JavaScript. These apps must be downloaded and installed on mobile device, in order to provide a possibility to be used.

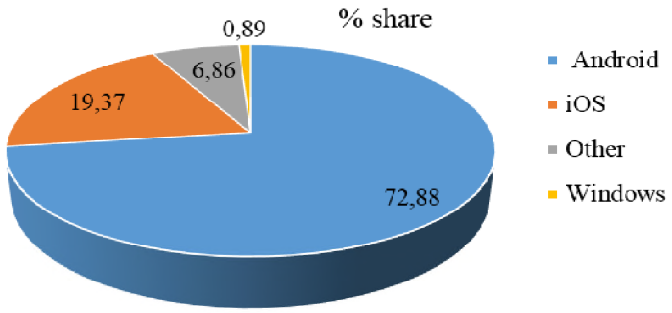

Figure 1: The share of mobile phone operating systems worldwide [3]

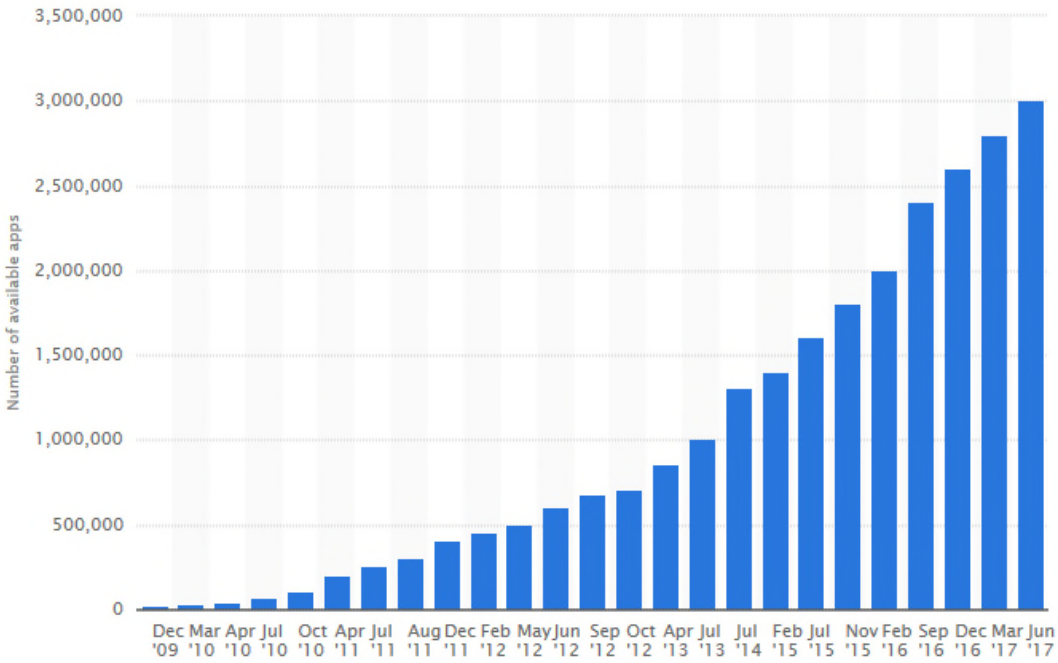

Figure 2: Number of available mobile app on Google Play [4]

2.1. Android application development

Android is a mobile OS which provides large set of features that supports mobile apps. Android app development refers as process of creating Android apps using Java. Android apps have four types of components [5]:

- Activities: An Android app consists of a many activities. Activities are screens that users interact in order to perform an action like send an e-mail or surfing on the web. Generally there is a main activity which is presented to the user when the app starts. In order to perform multiple actions activities call each other.
- Services: A service runs in the background to perform long running or remote processes and does not provide a user interface. For example a service enables the music to play in the background without interrupting a running app.
- Content providers: Content providers enable for query and modify data stored in file system, database or any other storage location.
- Broadcast receivers: Broadcast receivers deal with system or app broadcast announcements such as low battery, screen turn off. They do not provide a user interface. They may send notifications to alert user about the announcement.

The app development environment for android includes Java Development Kit, Android SDK, an IDE (Eclipse or Android Studio) and a virtual device (emulator). Android SDK provides a variety of tools that helps for developing Android apps. Virtual device which is provided by Android SDK, helps developers to develop, run and test apps without using a physical device [6].
When an Android code is compiled, an APK (Android package) file is generated. This APK file contains all the contents of the app. In order to install an Android app to the device, this APK file is used.

3. MOBILE APPLICATION TESTING

Mobile app testing is the process in which app developed for mobile devices such are smartphones and tablets are tested for usability, functionality and to be without bugs. A huge number of mobile testing tools have been developed in recent years to support mobile development. As more companies are developing mobile devices and in the marketplace is existing more devices, platforms, and versions. All of this affect the necessity of testing mobile apps. When it comes choosing what mobile testing tool, there is a huge array of options, each with different strengths and weaknesses.
Mobile apps are so different than web or desktop apps and need specific testing tools. The new testing strategies are mostly caused by the different nature of mobile apps. The cost of mobile apps will be lower if bugs are detecting before it install on the internet. There are different operating system, app framework, phone manufacturer and hardware layers, so the automation of testing is needed. Mobile testing can be divided into the following categories (Fig. 3):

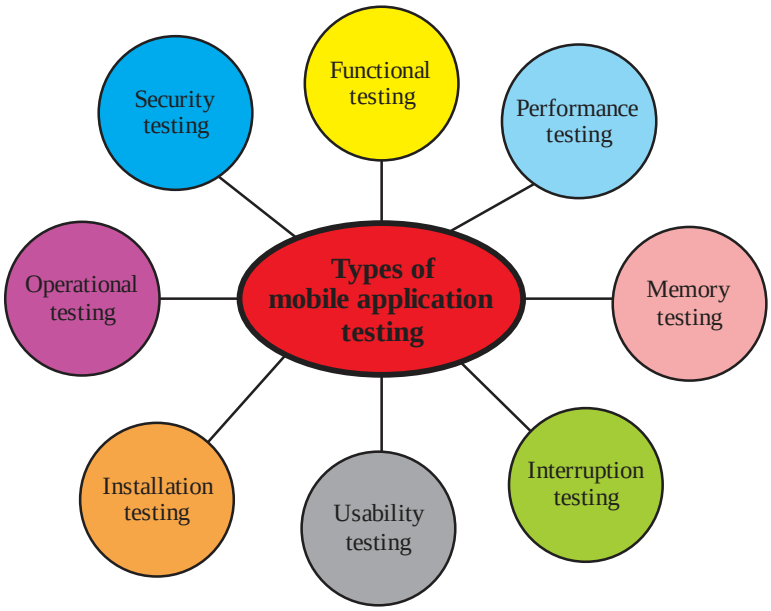

Figure 3: Categories of mobile testing

- *Functional testing* is used to check functionalities of mobile app in accordance with the requirements of the *specification*

- *Performance testing* is create for testing performance of client app, server, and network.
- *Memory testing* is performed to test the optimized memory usage by mobile app. Mobile devices has limited memory as compared to computers and that is main reason for this type of testing.
- *Interruption testing* is used while running the app to check interruption of incoming call or SMS, low memory warning, low battery warning etc.
- *Installation testing* is used to check installation process includes updating and uninstalling.
- *Usability testing* is used to check efficiency, effectiveness, and satisfaction of mobile app.
- *Operational testing* is used to test backups and recovery plan if battery goes down or data loss while upgrading the app form store.
- *Security testing* is used to test if the information system is protects app data or not. The main goal is to avoid someone to break app and steal information in confidential mobile app (bank app, money transactions, confidential information …).

Testing is an integral part of software development lifecycle. It helps to improve the quality of apps, ensure user satisfaction, and reduce development time which will spent on fixing defects [6]. Thus, new testing technologies and strategies are being developed to make this process faster, much efficient and more reliable. Anyone who making apps should test them. Failure to catch bugs or regressions can cost companies thousands of dollars a day, and releasing broken apps can frustrate and alienate end users.

Testing can be manual or automated. Any complex app must be testing with automated test. The reasons for making automated mobile testing are probably the same as for traditional web development. While manual testing can be useful, it is a slow and human resource-heavy process. By automating testing, a suite of tests can run and that would take a manual tester hours to complete in minutes or seconds. Speeding up testing can allow to expand test coverage so it can be more confident that app are releasing with bug-free code. Not only is manual testing is slower, but increasing testing can be difficult. With automated mobile testing tools, increasing which platforms you are testing on and running significantly more tests is easy. The ability to reuse tests over and over also increases test capabilities.

Automated testing can save you time and money, since developer can spend resources on manual testing. An automated mobile testing can be a cost-effective solution for ensuring that app is releasing bug-free apps.

As mobile apps have become an essential part of our lives, importance of mobile app testing increases continuously. Mobile apps have special characteristics that make mobile app testing different from traditional and web app testing. These typical characteristics can be listed as follows [7]:

- Mobile connectivity: Mobile apps connect to mobile networks which may be different in terms of speed, security and reliability. This property arises the need for extra functional testing performed under different connectivity scenarios.
- Limited resources: Mobile devices are far away from computers in terms of hardware resources such as RAM, disk space and CPU. Thus, resource shortage should be considered during testing process.
- Autonomy: The functionality of traditional computers relies on electricity supply. On the other hand each mobile app may require different energy consumption. For instance an app that requires continuous 3G connectivity, strongly affects the autonomy of the device. That is why; energy consumption of mobile device should be evaluated during testing process.
- New user interface: Mobile devices vary in terms of UI properties such as screen size and resolution. Mobile apps may look and differently on different UI. Thus, during GUI testing mobile apps it is necessary to consider that different mobile devices may react differently to the same app because of the differences of user interface.
- Context awareness: Mobile apps may sense the context and act according to the different contextual input such as temperature, location and brightness. The number of contextual input may be huge. Thus, context specific testing techniques and coverage criteria should be used for mobile app testing.
- Adaptation: Mobile app may adapt to the contextual information during its execution. This adaptation should be considered during testing process.
- New programming language: New frameworks, APIs, libraries and programming languages such as Objective-C is used for mobile apps. Traditional testing techniques are needed to be revised according to them.
- New Mobile Operating System: Mobile operating systems (Android and iOS) are different from computer operating systems and from each other. Moreover, new versions of mobile operating systems are released continuously. Testing approaches that may detect bugs related to the unreliability and variety of operating systems should be used for mobile app testing.
- Diversity of phones and phone makers: There are a big number of different mobile devices and vendors. It is stated that 1.800 hardware/OS different configurations exist. This situation arises the need for testing techniques to cover maximum diversity.
- Touch screens: The main input source of mobile apps are touch screens. Thus, considering touch screen functionalities is an important step of mobile app testing.

Android app testing refers to the set of activities to evaluate an Android app. As Android is a mobile operating system, Android apps have the properties stated above and these properties should be considered during testing process. It allows running test suites on either a real Android device or a virtual Android device [8].
Android testing is started to use few years ago. The number of books, papers and reports are poorly. Documentation taking from Internet sources, such as tutorials, blogs and forums are more up to date and extensive than papers and books.

3.1. Testing approaches

Here are given four popular mobile app testing approaches, based on the underlying client–server infrastructure. Figure 4 illustrates the different infrastructures [9].
The emulation-based testing (Fig. 4a)) use a mobile device emulator (simulator), which creates a virtual machine version of a mobile device for study on a personal computer. It is often necessary to include a mobile platform's software development kit (for instance Android SDK). It is relatively inexpensive because no mobile devices have to be purchased or no testing laboratory is needed but it can only be used to assess very limited system functionality. This approach is low-cost and has several limitations as difficulty validating a full set of gestures. Most emulators support very limited gestures and device-specific functions. It has a limited scale for testing quality of service (QoS). To overcome these problems, a simulation-based approach can create a mobile test simulator to mimic various mobile client operations (such as diverse gestures) and support more than one mobile client. It is impossible to deal with different devices and mobile platforms because emulators are usually based on a specific device or platform.

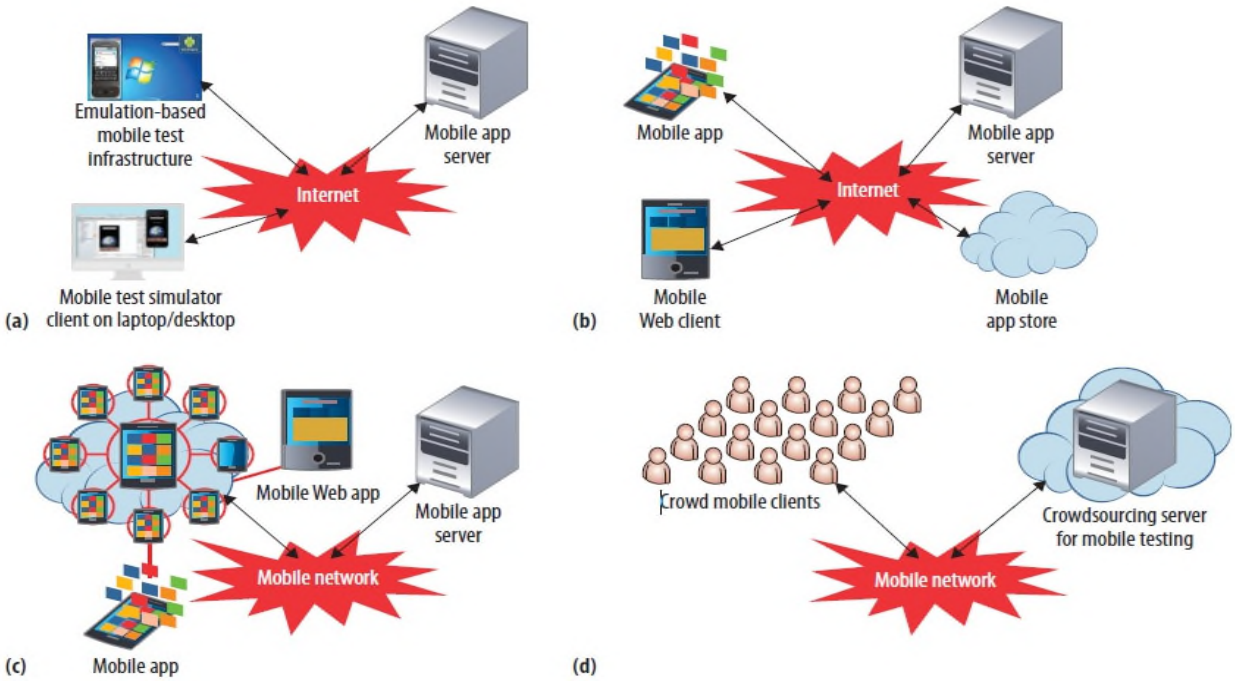

Figure 4: Different mobile test infrastructures: a) emulation, b) cloud, c) device, and d) crowd based [9]

The device-based testing approach (Figure 4b)) requires using testing laboratory and purchasing mobile devices, which is more costly than emulation based approaches but can verify device-based functions, behaviors, and QoS parameters that other approaches cannot. The advantage of this is being able to validate its base mobile networks via reconfigurations and selections in a testing environment. One of the major problem with this approach is rapid changes in mobile devices and platforms. Another challenge is its limitations related to system QoS because different tests require many mobile devices, which is usually impossible for companies.
Cloud testing approach (Figure 4c)) based on testing through the cloud. The main goal is to create a mobile device cloud that can support big testing services. This approach has the significant increase in demand for mobile testing services. It also allows different mobile users to supply their required testing environments with a rental service model. This can be more cost-effective than other approaches for comprehensive applications, and it is more effective for testing activities on mobile devices.
The crowd-based testing approach (Figure 4d)) involves using freelance or contracted testing engineers or a community of end users such as uTest (www.utest.com). With a crowd-based testing infrastructure and a service management server supported diverse users. Currently, a service vendor supports primitive test management, a testing service, and

bug reporting. Most mobile test operations are managed with very limited mobile test automation tools. This approach offers the benefits that was no need to invest in a lab or purchase or rent devices, but at the risk of low testing quality.

4. OPEN-SOURCE FRAMEWORK FOR TESTING

There are many open-source framework for testing mobile apps. In this paper we are presented few most used testing frameworks.

4.1. Robotium

Robotium is one of the most popular open source framework for Android app. It was develop in 2010 by Renas Reda. Robotium is used for system and acceptance test scenarios and functional user interface testing. It can be used for hybrid and native app tests. The main reason for using this tool is similarity to Selenium which is one of most used framework for web app. Robotium framework development is not finish and anyone can access on GitHub and give contribution to development [9]. Benefit of using this framework are:

- Short time for write test code,
- Developing test cases for unknown app,
- It support at the same time several automatic activities,
- Automatic timing and delays
- Fast test execution
- Integration with different frameworks (Maven or Ant)

Drawbacks of Robotium are:

- Robotium cannot handle Flash or Web components,
- It handles only one app at a time,
- It cannot simulate clicking on soft keyboard using Robotium (need to use 'enterText()' to enter text into an EditText field)
- Robotium cannot interact with Status Bar Notifications – that is, pull down the Notification area and click on a specified Notification.
- Can be a slower when is running on older devices.

4.2. Appium

Appium is open source and cross-platform testing framework and it can be used for Android and iOS apps. Writing one test code could be running on both platforms without changing the code. Its first release was 2012 by Don Cuellar on GitHub. It supports different programing language like Java, PHP, C#, Ruby and testing native, hybrid and mobile web app. Mobile web apps do not require installation and needs to be tested using different mobile browsers [10].
Appium is a strong mobile testing tool that will continuously improve. Testing the same app before put it on the marketplace. If one version of app are testing and submit another version, there is a risk for bugs that didn't predict in the marketplace. It is reasonable to test the same app to minimize this risk. Tests can be write in any framework using any language. Testers choose different language to write test cases and many even write in multiple languages. Any tool that limits what languages and frameworks it supports will be limiting to those using it. Appium support many languages and frameworks, to give developers great flexibility. It use existing specifications and standards for web testing tools, and reuse and adapt these existing standards to determine what the standards for mobile testing tools should be.
One of the main disadvantage of this framework is that less suitable for unit testing.

4.3. Selendroid

An open source automation framework which drives off the UI of android native, hybrid and mobile web app. A powerful testing tool that can be used on emulators and real devices. It can write tests using the Selenium 2 client APIs because it still reuses the existing infrastructure for web. The Selendroid suite consists four components: Web Driver client, Selendroid-Server, Android Driver App and Selendroid-stand alone. Java JDK, Android SDK and Eclipse must be installed if Selendroid is using [10].
Benefits of using this framework are:

- Can interact with multiple Android devices and simulators at the same time,
- Can simulate human actions like touch, swipe, drag etc. on devices,
- Compatible with JSON Wire Protocol/Selenium 3 Ready.
- No modification of app under test required in order to automate it.

- Testing the mobile web using built in Android driver webview app.
- Same concept for automating native or hybrid apps.
- UI elements can be found by different locator types.
- Existing Emulators are started automatically.
- Supports hot plugging of hardware devices.
- Can be extended at runtime with your own extensions.

Drawbacks of Selendroid:

- It is quite slow and on some machines with less than 4GB RAM, it is unusable.

4.4. How to choose the right framework for testing

It is difficult to choose one software tool which will be the best for testing mobile app. Testers usually decide to pick open-source framework which will finish testing fast, but that will be mistake. Depending of platform which will be chosen for mobile app, and some other detail, in Table 1. is presented comparative analysis of most used frameworks [9].

Table 1: Comparative analysis of open-source framework

Features/Tools	Appium	Solendroid	Robotium
Android OS	Yes	Yes	Yes
iOS	Yes	No	No
UI Driven	Yes	Yes	Yes
Unit Test	No	No	No
Native	Yes	Yes	Yes
Hybrid	Yes	Yes	Yes
Mobile Web	Yes	Yes	No
Scripting Language	Java, C#, Pyton, Ruby, Perl	Java, C#, Pyton, Ruby, Perl	Java
Test Recording Capability	Partial (Using Appium Inspector)	No	Yes (only in paid version)
Multi Device Execution (on in-house devices)	Yes (Selenium Grid)	Yes (Selenium Grid)	Framework Dev. required
Integration with CI tools (Jenkins)	Plug-ins available	Framework Dev. required	Framework Dev. required
Easy of learning	Medium	Medium	Simple
GUI-based function testing	Yes	Yes	Yes
Performance testing	No	No	No
Emulation based testing	Yes	Yes	Yes
Device-based testing	Yes	Yes	Yes
License/Subscription	No	No	No

5. CONCLUSION

Software testing is one of the most important stages in software development life cycle. Web and mobile apps need to be tested to reduce the number of bugs and increase software quality. All bigger computer companies use many testing tools (frameworks) to test developed apps. In recent years, a large number of free tools have been developed to test mobile apps. Companies that have developed these tools allow users to complement and upgrade that tools, with main goal to improve tools capabilities. It is impossible to determine which tool is the best, as this depends primarily on the type of mobile apps. When choosing a tool, it must pay attention to whether the mobile app has been developed for cross platforms for example Android or iOS apps.

No one can predict how many open-source framework will be developed in the future, but it is obviously that number of mobile apps will growing rapidly and testing will be required.

REFERENCES

[1] https://en.wikipedia.org/wiki/Smartphone Internet access: 1. June 2017.
[2] https://en.wikipedia.org/wiki/Mobile_app_testing Internet access: 1. June 2017.
[3] http://gs.statcounter.com/os-market-share/mobile/worldwide Internet access: 1. June 2017.
[4] https://www.statista.com/statistics/266210/number-of-available-apps-in-the-google-play-store/ Internet access: 1. June 2017.
[5] HOLL, K.; ELBERZHAGER, F.: *Mobile Application Quality Assurance: Reading Scenarios as Inspection and Testing Support*, 2016 42th Euromicro Conference on Software Engineering and Advanced Applications, Limassol, Cyprus, pp. 245-249.
[6] TARLINDER, A.: *Developer Testing - Building Quality into Software*, Addison-Wesley, ISBN-13: 978-0-13-429106-2, 2017, 313 p.
[7] NIMBALKAR, R. R.: *Mobile App Testing and Challenges*, International Journal of Science and Research (IJSR), India Online ISSN: 2319-7064, Volume 2, Issue 7, July 2013, pp. 56-58.
[8] BLUNDELL, P., MILANO, D. T.: *Learning Android Application Testing*, PACKT Publishing, ISBN 978-1-78439-533-9, 2015, 247 p.
[9] GAO, J.; BAI, X.; TSAI, W. T.; UEHARA, T.: Mobile Application Testing: A Tutorial, *Computer*, vol. 47, No. 2, Feb. 2014, pp. 46-55.
[10] ZADGAONKAR, H.: *Robotium Automated Testing Android - Efficiently automate test cases for Android applications using Robotium*, PACKT Publishing, ISBN 978-1-78216-801-0, 2013, 81 p.
[11] GARG, S.: *Appium Recipes*, Apress, ISBN: 978-1-4842-2417-5, 2016, 179 p.