Information and Software Technology xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Characterizing mobile apps from a source and test code viewpoint

Davi Bernardo Silva[a,b], Marcelo Medeiros Eler[c], Vinicius H.S. Durelli[d], Andre Takeshi Endo*[,a]

[a] Universidade Tecnologica Federal do Parana (UTFPR), Cornelio Procopio, Brazil
[b] Instituto Federal de Santa Catarina (IFSC), Geraldo Werninghaus, Brazil
[c] Universidade de Sao Paulo (USP), Sao Paulo, Brazil
[d] Universidade Federal de Sao Joao del Rei (UFSJ), Sao Joao del Rei, Brazil

ARTICLE INFO

Keywords:
Android
Mobile computing
Open source
Software testing
Automated tests

ABSTRACT

Context: while the mobile computing market has expanded and become critical, the amount and complexity of mobile apps have also increased. To assure reliability, these apps require software engineering methods, mainly verification, validation, and testing. However, mobile app testing is a challenging activity due to the diversity and limitations found in mobile devices. Thus, it would be interesting to characterize mobile apps in hopes of assisting in the definition of more efficient and effective testing approaches. *Objective:* this paper aims to identify and quantify the specific characteristics of mobile apps so that testers can draw from this knowledge and tailor software testing activities to mobile apps. We investigate the presence of automated tests, adopted frameworks, external connectivity, graphical user interface (GUI) elements, sensors, and different system configurations. *Method:* we developed a tool to support the automatic extraction of characteristics from Android apps. We conducted an empirical study with a sample of 663 open source mobile apps. *Results:* we found that one third of the projects perform automated testing. The frameworks used in these projects can be divided into three groups: unit testing, GUI testing, and mocking. There is a medium correlation between project size and test presence. Specific features of mobile apps (connectivity, GUI, sensors, and multiple configurations) are present in the projects, however, they are fully covered by tests. *Conclusion:* automated tests are still not developed in a systematic way. Interestingly, measures of app popularity (number of downloads and rating) do not seem to be correlated with the presence of tests. However, the results show a correlation of the project size and more critical domains with the existence of automated tests. Although challenges such as connectivity, sensors, and multiple configurations are present in the examined apps, only one tool has been identified to support the testing of these challenges.

1. Introduction

Mobile computing has gone mainstream due to the many technological advances over the past decades. According to an Ericsson mobility report, there are 2.6 billion smartphone subscriptions in the world and this number is expected to hit 6.1 billion by 2020 [1]. The smartphones, along with tablets, e-readers and wearables, run mobile apps on specific operating systems (OSes), such as Google Android [2], Apple iOS [3], and Windows Phone [4]. Among them, Android was the first platform to be provided as open source, allowing not only the development of mobile apps [5], but also the conduction of empirical studies.

This appealing technology and its massive market attracted several information technology (IT) professionals to develop software for mobile computing [6,7]. Mobile apps were initially developed for entertainment purposes, but they have reached even critical domains like health, finance, and industry [8]. Recent reports have shown that Google Play, the most popular mobile app[1] market for the Android platform, currently offers 2.8 million apps for multiple domains. Similarly, Apple's App Store, the app market for the iOS platform, has 2.2 million apps available [9].

Mobile app business is booming, thus a way to gain advantage over other competitors is to deliver apps that are reliable. Naturally, developers have to resort to software testing approaches in order to assure the quality of their products. Gao et al. [10] define mobile app testing as all test activities that, by means of well-defined methods and tools, intend to assure quality in functionalities, behaviors, performance and services, as well as mobility, connectivity, security, usability, privacy, and interoperability. In particular, approaches and tools have been developed to automate the execution of test cases created by developers as well as to automatically generate test cases [11,12]. In this paper,

* Corresponding author.
E-mail addresses: davi.bernardo@ifsc.edu.br (D.B. Silva), marceloeler@usp.br (M.M. Eler), durelli@ufsj.edu.br (V.H.S. Durelli), andreendo@utfpr.edu.br (A.T. Endo).

[1] Throughout this document the terms mobile app and app will be used interchangeably.

https://doi.org/10.1016/j.infsof.2018.05.006
Received 26 May 2017; Received in revised form 24 March 2018; Accepted 9 May 2018

however, we focus only on approaches that automate the execution of predefined test cases (as opposed to automatically generated test cases) in a systematic and formal fashion [13].

The development of mobile apps is relatively new, thus researchers and practitioners have come up with different approaches for dealing with the testing challenges posed by mobile apps [7,8,14]. For instance, researchers have been investigating how to test native apps that use specific features of the devices (e.g., camera, sensors, accelerometers, and geolocalization) [8,15–18]. However, there is a lack of studies that characterize the mobile apps with respect to the challenges they present to the testing activity. The literature also lacks studies reporting whether and how such challenges have been addressed in mobile development projects.

In this context, we set out to identify and quantify the characteristics of mobile apps that are relevant from a software testing viewpoint. In particular, we conducted an empirical study with 663 open source projects aiming to extract pieces of information related to the presence of automated tests, frameworks adopted, the presence of challenges (namely, rich GUIs, sensors, connectivity, and multiple configurations) and whether they are tested or not. Moreover, we correlated the presence of tests with project size, category, and popularity measures (i.e., number of downloads and rating). We automated the data extraction process by developing a static analysis tool able to extract the aforementioned information from Android projects.

Preliminary results of this paper were published in Silva et al. [19]. The improvements made to this paper are fourfold: *(i)* we adopted a much larger sample of apps (in the previous version of this study, only 19 mobile apps were taken into account); *(ii)* we revisited and refined the research questions; *(iii)* we developed a tool that automates the process of extracting the information that is relevant to answer our research questions; and *(iv)* we provide a comprehensive analysis and discussion of the results, which takes into account a larger sample comprising 663 apps.

The remainder of this paper is organized as follows. Section 2 covers background and challenges related to mobile app testing. Section 3 describes the setup of our empirical study. Section 4 presents an analysis of results and Section 5 further discusses our results. Section 6 outlines related work. Section 7 presents the conclusion and outlines future work.

2. Background

Mobile computing is the manipulation of portable devices through mobile apps to exchange information regardless of their physical location [20]. Thus, a mobile app is any software that is developed to run on mobile devices [21,22]. Currently, such apps execute under specific platforms like Android, Apple iOS, or Windows Phone, for example. The Android platform is an open source, layered software environment based on the Linux kernel [5]. Android was primarily designed for smartphones, tablets, and e-readers. For end users, Android is a complete OS that delivers user interface, code libraries, applications, development frameworks, and telecommunications resources [5].

The literature identifies specific features of mobile apps that pose challenges to software testing [7,8,10,23]. We analyze the following challenges.

2.1. Connectivity

mobile connectivity is one of the most peculiar and critical characteristics for which the mobile devices were developed. Mobile apps typically rely on wireless networks or Bluetooth to connect people and systems, transmitting and receiving information. Therefore, the speed, security, and performance of mobile networks, as well as the reliability and proper operation of apps rely heavily on the type of connection available.

2.2. Rich GUIs

touch screens are subject to greater interaction with the user and they heavily rely on system response time to work properly. Although there are some guidelines and initiatives for coming up with standards for mobile development, apps can behave differently across a range of screen resolutions. For instance, it is common for the same app to be broken down into several branches of development to meet the diversity of devices [7].

2.3. Sensors

driven by contextual information, mobile apps can evolve and adapt. The portability offered by mobile devices requires apps capable of identifying changes in the environment and operate properly. Mobile apps rely on data provided by context providers such as sensors and connectivity. These data are provided by the environment and can represent an enormous and unpredictable amount of input data. Testing whether the app works correctly in any environment, transparently, and with any contextual input is a challenge.

2.4. Multiple configurations

There are a myriad of different mobile devices on the market. They are produced by a diverse group of suppliers and have different characteristics in terms of software and hardware. New versions of OSes are often launched and existing mobile apps are supposed to run flawlessly on these new versions. Usually, when running on different devices, mobile apps may behave differently due to variations in both hardware and software. For example, the sensors are typically calibrated differently so that two distinct devices running the same app in the same environment could yield diverging outputs.

3. Study setting

We devised an empirical study to characterize mobile apps from a software testing viewpoint. Table 1 shows the research questions (RQs) we set out to answer in this paper.

The rationale behind the first RQ is to probe into whether app projects present any evidence of test automation ($\mathbf{RQ}_1$), how extensive test cases are ($\mathbf{RQ}_{1.1}$), and which are the most popular testing frameworks ($\mathbf{RQ}_{1.2}$). Additionally, we set out to investigate if there is any correlation between the presence of test automation and several other metrics ($\mathbf{RQ}_{1.3}$): project size, thus shedding some light on whether larger projects require more test automation than smaller projects; categories, so that we can look into whether specific types of apps are more prone to be automated; rating and number of downloads, which are related to some notion of quality attributed by the end user (i.e., rating) and popularity (i.e., number of downloads) among users.

Answering $\mathbf{RQ}_1$ may provide insight into app developers culture when it comes to automating software testing activities. In particular, results can motivate the development of new features to support automated testing if it is broadly adopted, or motivate the investigation

Table 1
Research questions.

ID	Questions
$\mathbf{RQ}_1$	How often does test execution automation appear in mobile app projects?
$\mathbf{RQ}_{1.1}$	What is the ratio between production code and test code?
$\mathbf{RQ}_{1.2}$	Which testing frameworks are adopted?
$\mathbf{RQ}_{1.3}$	Is the presence of test automation correlated with project size, categories, rating, and number of downloads?
$\mathbf{RQ}_2$	How often the mobile app testing's challenges appear in existing projects?
$\mathbf{RQ}_3$	Does the existing test code address the testing challenges identified?