Experience Report on Developing a Crowdsourcing Test Platform for Mobile Applications

Nguyen Thanh Binh[1](✉), Mariem Allagui[2], Oum-El-Kheir Aktouf[2], Ioannis Parissis[2], and Le Thi Thanh Binh[3]

[1] The University of Danang - Vietnam Korea, University of Information and Communication Technology (VKU), Da Nang, Viet Nam
ntbinh@vku.udn.vn

[2] Grenoble Alpes Univ. Grenoble INP, LCIS, Valence, France
mariem.allagui@grenoble-inp.org,
oum-el-kheir.aktouf@lcis.grenoble-inp.fr,
ioannis.parissis@grenoble-inp.fr

[3] The University of Danang, University of Science and Education, Da Nang, Viet Nam
lttbinh@ued.udn.vn

Abstract. Crowdsourcing-based testing is a recent approach where testing is operated by volunteer users through the cloud. This approach is particularly suited for mobile applications since various users operating in various contexts can be involved. In the field of software engineering, crowd-testing has acquired a reputation for supporting the testing tasks, not only by professional testers, but also by end users. In this paper, we present TMACSTest (Testing of Mobile Applications using Crowdsourcing). This platform provides the important features for crowdsourcing testing of mobile apps by means of the following functionalities: It allows mobile app providers to register and upload mobile apps for testing, and it allows volunteering Internet users to register and test uploaded mobile apps. Expected behavior is that uploaded mobile apps are tested by many different Internet users in order to cover different runtime platforms and meaningful geographical locations.

Keywords: Mobile application testing · Crowdsourced testing · Crowd-based testing

1 Introduction

Nowadays, mobile applications are becoming increasingly popular. Some studies estimate that by 2020 there will be 378 billion downloads of mobile applications. However, some statistics consider that about 70% of users uninstall mobile apps due to a bug [1]. Therefore, mobile apps need to be tested across a full range of operating contexts before their release on the market to ensure a high-quality user experience. Mobile apps are usually expected to work on many devices with various screen settings and operating

M. Hernes et al. (Eds.): ICCCI 2020, CCIS 1287, pp. 651–661, 2020.
https://doi.org/10.1007/978-3-030-63119-2_53

systems, in different locations, and across a host of carriers and networking technologies. As a result, mobile app testing has become difficult and pricey in terms of resources.

There are a multitude of challenges and issues in mobile app testing [2, 14, 15]. Some of them are:

- Higher testing cost from using real mobile devices and testers.
- Mobile apps are commonly supported by various wireless connectivity infrastructures and service plans.
- This requires mobile networking service costs and infrastructure support in a lab-based testing environment.
- Mobility provides location-based services, which need to be tested in many real user environments on different geographical locations with different languages.

Thus, we need new cost-effective test methods and tools to ensure quality of mobile apps.

Recently, inspired by the crowdsourcing concept-crowdsourcing is an emerging distributed problem solving model by online workers [3]. A new testing approach named crowdsourced testing has gained much attention in the software engineering community [4, 5].

Crowdsourced testing uses an online platform to assign software test tasks to a group of online testers. And it has become an imperative approach and has gained a reputation for the software engineering community [6]. However, with regards to test efficiency, two main questions have been arisen: (1) how to ensure an efficient test execution by crowdsourced testers, who are not necessarily mobile applications or even software tester experts, and (2) how to allow the analysis of test reports and how to ensure that specific test coverage measures are reached.

This paper presents an ongoing development of a web application, called TMACSTest, that uses crowdsourcing to test mobile applications. As a web application testing platform, TMACSTest connects mobile apps developers with thousands of crowdsourced mobile testers to test those applications. To answer question (1) above, TMACSTest provides users with specific mobile application models to allow efficient application of model-based testing to mobile applications. Question (2) is taken into account by providing an aggregation procedure of test result reports with an in-depth analysis to deduce relevant test results and test coverage measures. By doing so, TMACSTest makes the usage of crowdsourced test for mobile applications easier and more efficient.

The remaining parts of this paper are organized as follows. In Sect. 2, we present the background on crowdsourced testing. In Sect. 3, we describe the developed solution for crowdsourced testing provided by TMACSTest. Section 4 provides the details of TMACSTest development. Finally, Sect. 5 concludes the paper.

2 Background

2.1 Basics on Crowdsourced Testing

Crowdsourcing is an distributed problem-solving and production model. It was used to describe how businesses were using the Internet to outsource work to the crowd. The term “crowdsourcing” was coined in 2006 by Howe and Robinson. Crowdsourcing used in software testing is called “Crowdsourced Testing” or “Crowd Testing” [7].

Crowdsourced testing activities including performance testing, usability testing, GUI testing… This method differs from the traditional testing approach in that the testing is carried out by testers from different areas, testers who can be professional testers, novice testers, real application users, etc.

We use mobile crowdsourced testing to refer to test activities for mobile applications that use freelance testing engineers or end-users communities on diverse wireless connectivity infrastructures to ensure the quality of mobile applications in terms of functions, behavior, performance and service quality, as well as special features of mobile applications such as mobility, usability, security and compatibility.

Mobile crowdsourced testing has three following main components [7] (Fig. 1):

- The crowd mobile testers: individuals who perform the test. They work as freelance mobile testers for selected mobile application testing projects. They use their own mobile devices which were pre-configured real-world wireless network infrastructures to carry out the tasks.
- The crowd mobile seekers: people submit projects for testing. They hire free mobile testers to complete their published mobile app testing projects and tasks, and provide them with incentives and payments based on their services.
- An intermediation platform: building a link between the crowd mobile testers and the crowd mobile searchers. This serves as a community support tool that allows customers to express their needs and the individuals and companies that make up the crowd to meet these needs.

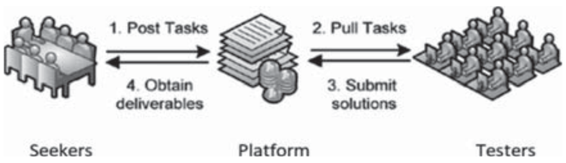


Fig. 1. Components in mobile crowdsourced testing

Today, a number of mobile crowdsourced testing platforms have been developed. These crowdsourced testing platforms provide a cloud-based infrastructure to connect mobile app developers with thousands of crowdsourced mobile testers. Some of these platforms areUtest, MyCroud QA, Pay4Bug, Crowdsourcedtesting, Testin, QA infotech and 99Test.

The particularity of Utest is that it offers to testers articles and tools (Forum, courses) to improve their technical skills in testing applications. The Utest platform emerged in 2007 and is based on the principle of crowd-based testing. Since its foundation, the platform has undergone strong growth given the large number of registered testers and developers and the quality of service delivered for testing web, mobile and desktop applications [8].

This platform offers developers three types of tests, the Beta test, the compatibility test and the functional test. The Testin platform is used by leading companies in their field (IBM, DELL, Intel, Samsung…) [9].

This platform gives developers the ability to test several types of applications: websites, mobile apps, video games and desktop applications. This platform provides developers three types of tests: functional test, user test and location test [10].

Crowdsourcing is the collection from a crowd of people working at diverse workplaces. Particularly, the process is online, leading to good results. Testers involved in crowdsourcing are paid only when their work is finished or when the bug is found. Crowdsourcing testing is the testing approach is called "taas: test as a service".

2.2 Technical Aspects of Crowdsourced Testing

Testing is necessary to effectively implement the software application or product. It is extremely important to ensure that the application will not lead to any failures as it can be very costly in the future or in later stages of development. The easiest and fastest way to test software applications is to use cloud-based testing tools available online [13].

There are some differences when compared to traditional methods such as testers who are from different locations, testers who do not belong to any specific organization, testers may not be professional or experienced, etc. Crowd testing provides benefits, cost effectiveness and makes the software reliable and mostly error free.

There are various reasons for which testers should go with crowdsourced testing [11]. The reasons are listed below:

- Crowdsourced testers can test an application in many different environments, simultaneously with different internet bandwidths, with different devices, with different test streams, etc.
- Crowdsourced testing performed by testers are unbiased because they do not belong to any organization.
- The crowdsourced model can complete the test in a short amount of time because a large number of testers participate in the testing process around the globe.
- Crowdsourced testers are being paid based on the valid errors they find, so this is a very cost-effective test.
- There is no time limit for crowdsourced testers because they belong to different time zones and they can work even after office hours or weekends or work late at night, which leads to quick order for a quality product.

Figure 2 indicates that an application the app seeker can issue a test task that including the mobile applications under test and expected test requirements. These requirements often contain the detailed instructions to test for mobile application testing and the test result. Each task is distributed online as a web page within a time window which setting by the publisher. All crowd employees can choose published tasks to work according to their preferences and other motivations. During testing, testers use their own resources, such as their mobile devices to test mobile applications as required by the task and send test reports containing errors with details [16].

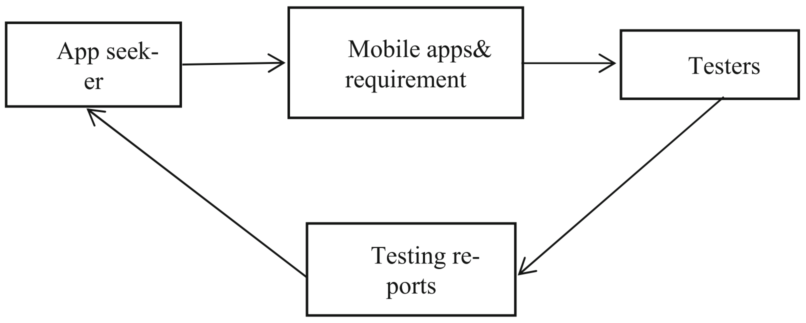


Fig. 2. Crowdsourced testing model for mobile apps

3 Development of a Test Platform: TMACSTEST

This section reports the development of mobile application testing with crowdsourcing (TMACSTest). Particularly, we illustrate two main reasons for developing this new platform: Providing mobile app models for performing model-based testing and providing an automated analysis of bug reports. Although work on these aspects is still ongoing, we report below the main directions that our research is currently investigating. In subsect. 3.1 below we make the link with mobile app development and the component-based software paradigm. Then we provide some conclusion regarding the testing models that can be used for mobile apps. In subsect. 3.2, the need for test results analysis is introduced.

3.1 Model-Based Testing for Mobile Applications

Component-based software development has proved for many years to be an effective and efficient way for software development, allowing component reuse and system evolution. Software components and libraries have historically offered tested and optimized solutions to various issues in software development. The concept of component-based software engineering already exists in mobile app development today – as UI widgets for native Android/iOS or as commercial Marketplace modules for Appcelerator. These components could provide developers with the building blocks needed to bridge the gap between web and native apps today. As a result, this approach has been investigated in the development of mobile applications [18]. For example, Android application development identifies specific main types of components within a mobile application:

- Activities: these are components that handle the user interaction;

- Services: these components handle background processing associated with a mobile application;
- Broadcast receivers: these components deal with the communication between the OS (Android) and the mobile applications;
- Content providers: are components that handle database management issues.

In addition to these basic component types, Android mobile application development distinguishes other components like views (UI elements), fragments (portions of user interfaces), etc.

Component-based models have function that follow us to understand the individual behavior of application components as well as interactions among them. They also provide a simple and the ultimate way to develop specific model-based testing approaches.

Software component testing refers to testing activity that examines component along with its design, generates component tests, identifies component faults and evaluates component reliability. Therefore, component testing plays a significant role for the development of a quality component based software product [4].

Software testing techniques developed for software components are mainly [2]:

- The Component Meta-data way;
- UML based test model for Component;
- Component Interaction Graph (CIG);
- Built-in-tests in components (BIT);
- Component Interaction Testing (CIT).

Table 1. below provides the main features of these testing techniques.

Table 1. Summary of component-based software testing techniques

Technique	Testing criteria
The Component Meta-data way	Attach additional information with the components
UML-based test model	Use Sequence and Collaboration diagrams to extract faults
Component Interaction Graph	Detect faults in the interfaces and interaction among components
Built-in-tests in components	Built-in-test as member function in source code
Component Interaction Testing	Capturing assumptions as formal test requirements

Using such models supports application analysis and helps to focus on the toughest and most important features while generating test cases or deriving test strategies. One specific concern of mobile applications is the location features and corresponding services [19].

Our issue concerns the testing of mobile app component services given a user/app location. Consequently, as a first step in our research, we intend to work on UML-based test model for component endowed with location features. Indeed, the proposed model aims to cover the location aspects of a given mobile app.

3.2 Test Results Analysis

The test results generated by all the testers for a test task would be submitted to the crowdtesting platform.

How to aggregate the test results together without conflict and provide useful analysis for the application or service developer is also a challenge to a mobile crowdtesting platform. Work on this topic should be based like preceding fault analysis approaches developed in [20].

4 Describing the Details of TMACSTest Development

This section reports the requirements specification phase and the design phase of TMACSTest.

4.1 The Requirements Specification Phase

In TMACSTest, there are two actors: tester and provider. The relationship between them is shown in the use case diagram (Fig. 3).

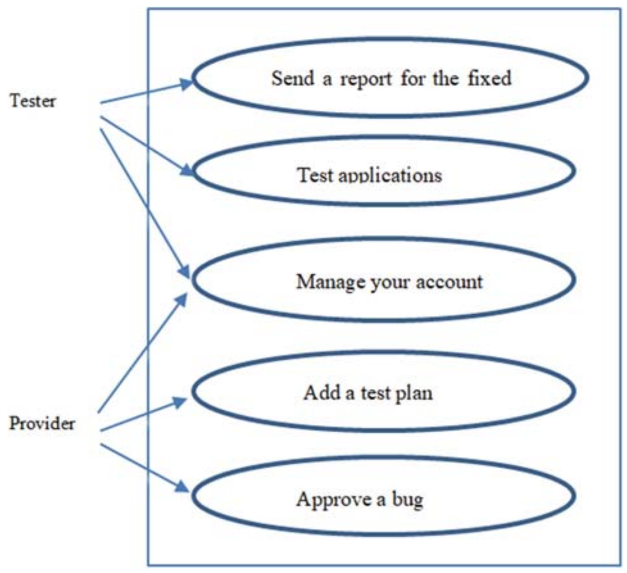


Fig. 3. Diagram of use cases of TMACSTest

Providers can release crowdsourced tasks for mobile app testing. Testers registered on the TMACSTest platform can then choose test tasks of interest to perform in their own hardware and software environments, and submit test reports for their completed tasks. The large number of participants naturally covers various hardware and software environments, which makes it possible to achieve a high operating contexts coverage.

4.2 The Design Phase

Component technologies are perceived as an important means to keep software architectures flexible. Though the ways for implementation of the concept vary greatly, we applied in TMACSTest the 3-tier architecture concept. Figure 4 below shows the detailed architecture of the application.

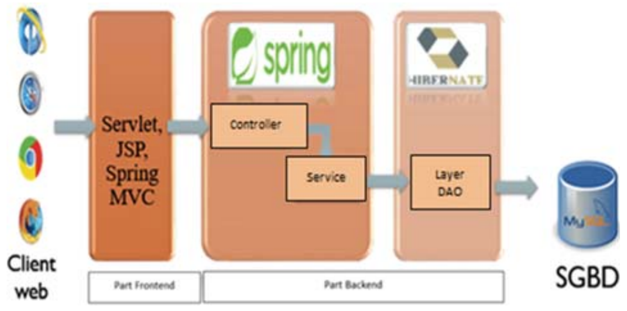


Fig. 4. Architecture of TMACSTest

TMACSTest is composed of two main parts:

- Backend part concerns the business layer and persistence layer of data previously represented in the 3-tier application architecture [12]. It is in this part that lies the core business of the application and the various treatments representing the application's functionalities.
- Frontend part is the web interface of the application. Web pages are linked to the application's controllers to satisfy the user's needs.

We used several technologies such as JAVA/JEE, Spring, Hibernate, Maven, Junit, MySQL as database and Cloud Foundry as a cloud host (Fig. 5).

TMACSTest has the following features:

- Configuration of the database associated with the application;
- Access to add, modify and even delete parts of the application;
- Functional web interface:
- Homepage (static page - Fig. 6, register page -Fig. 7).
- Provider page: the provider can add an application and display the list of applications (Fig. 8).
- Tester page contains the tester profile, the bug list, the test cycle list, and the addition of a bug for a test cycle (Fig. 9).

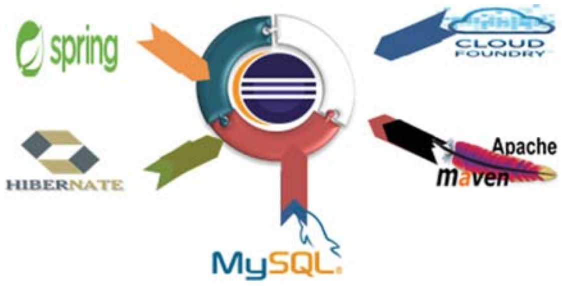
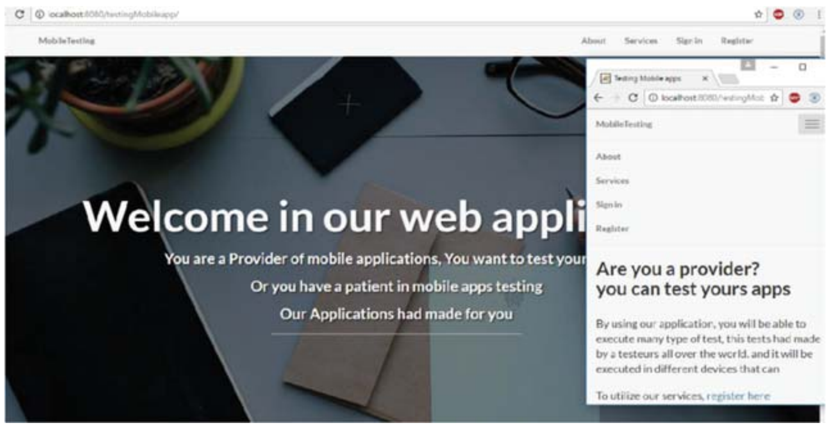


Fig. 5. Used technologies


localhost:8080/testingMobileapp/

MobileTesting　About　Services　Sign in　Register

Welcome in our web appli

You are a Provider of mobile applications, You want to test your

Or you have a patient in mobile apps testing

Our Applications had made for you

Testing Mobile apps

localhost:8080/testingMob

MobileTesting

About

Services

Sign in

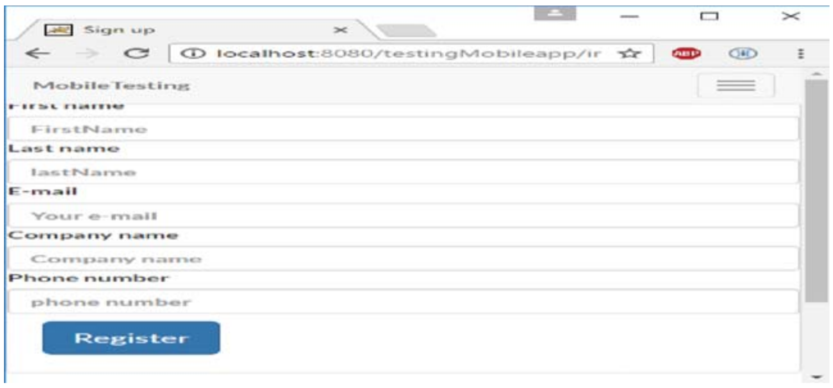
Register

Are you a provider?
you can test yours apps

By using our application, you will be able to execute many type of test, this tests had made by a testeurs all over the world. and it will be executed in different devices that can

To utilize our services, register here

Fig. 6. Application homepage


Sign up

localhost:8080/testingMobileapp/ir

MobileTesting

First name
FirstName

Last name
lastName

E-mail
Your e-mail

Company name
Company name

Phone number
phone number

Register

Fig. 7. Supplier registration form

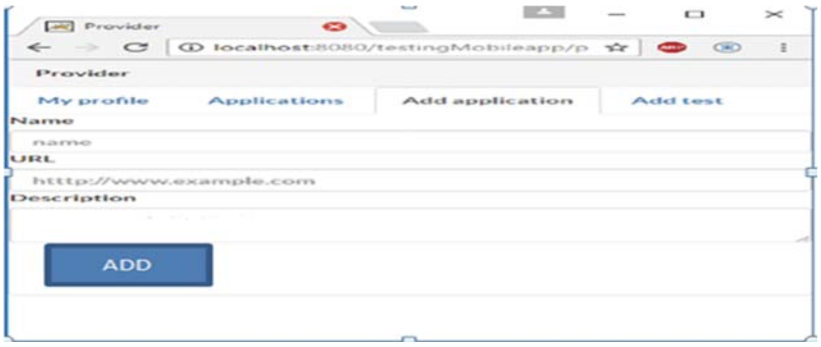


Fig. 8. Adding an application

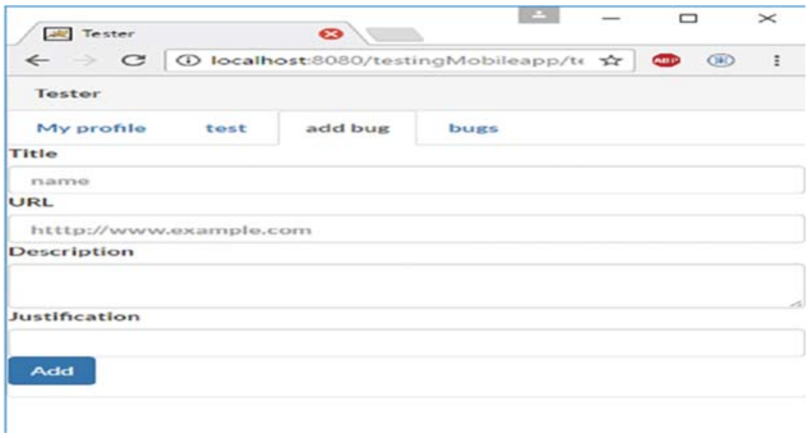
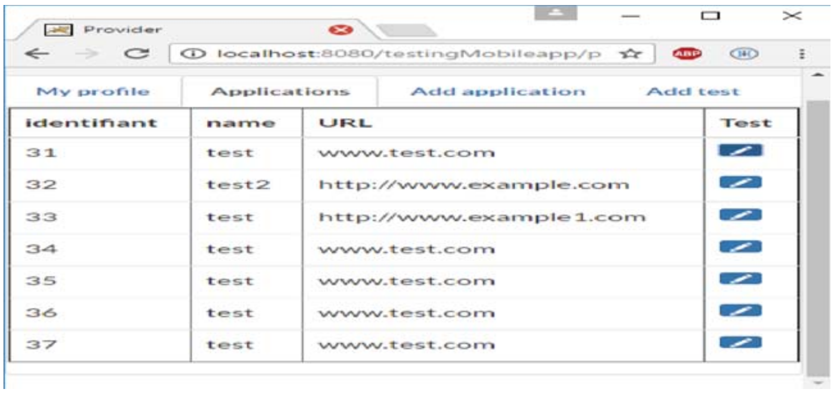


Fig. 9. Adding a bug



identifiant	name	URL	Test
31	test	www.test.com	
32	test2	http://www.example.com	
33	test	http://www.example1.com	
34	test	www.test.com	
35	test	www.test.com	
36	test	www.test.com	
37	test	www.test.com	

Fig. 10. Vendor application list

5 Conclusion and Future Work

TMACSTest is an intermediary linking the mobile application providers who need to test these applications before releasing them on the market, and the testers who test these applications (Fig. 10).

There are currently several perspectives for this work to come. First, we need to complete ongoing experimental work and provide corresponding results. Then, we aim to introduce the use of non-relational databases. These databases offer significant power for heterogeneous data processing against relational databases and use a large distribution of data and associated processing across multiple servers [17]. They are used primarily for distributed applications in the cloud.

References

1. Liang, C.-J.M., et al.: Caiipa: automated large-scale mobile app testing through contextual fuzzing. In: 20th International Conference on Mobile Computing and Networking, pp. 519–530 (2014)
2. Gao, J., Bai, X., Tsai, W.T.: Mobile application testing: a tutorial. Computer **47**, 46–55 (2014)
3. Mao, K., Capra, L., Harman, M., Jia, Y.: A survey of the use of crowdsourcing in software engineering. J. Syst. Softw. **126**, 57–84 (2016)
4. Chen, Z., Luo, B.: Quasi-crowdsourcing testing for educational projects. In: 36th International Conference on Software Engineering, pp. 272–275 (2014)
5. Liu, D., Bias, R.G., Lease, M., Kuipers, R.: Crowdsourcing for usability testing. Am. Soc. Inf. Sci. Technol. **49**, 1–10 (2012)
6. LaToza, T.D., van der Hoek, A.: Crowdsourcing in software engineering: models, motivations, and challenges. IEEE Softw. **33**, 74–80 (2016)
7. Speidel, D., Sridharan, M.: A framework and research agenda for crowdsourced testing (2013)
8. uTest. https://www.utest.com
9. TestCloud. https://www.xamarin.com/test-cloud
10. Crowdsourced Testing. https://crowdsourcedtesting.com
11. Guide to crowdsourced testing. https://www.softwaretestinghelp.com/guide-to-crowdsourced-testing
12. Gao, J., Tsai, W.T., Paul, R., Bai, X.Y.: Mobile testing-as-a-service (mobile taas) - infrastructure, issues, solutions and needs. In: IEEE International Symposium of High Assurance Systems Engineering, pp. 158–167 (2014)
13. Tsai, W.T., et al.: A cloud-based Taas infrastructure with tools for SaaS validation, performance and scalability evaluation, pp. 464–471 (2012)
14. Blokland, K., Mengerink, J., Pol, M.: Testing cloud service (2013)
15. Vashistha, A., Ahmed, P.: SaaS multi-tenancy isolation testing-challenges and issues. Int. J. Soft Comput. Eng. (IJSCE), **2**(5), 49–50 (2012). ISSN: 2231–2307
16. Naganathan, V., Sankarayya, S.: Overcoming challenges associated with SaaS testing. Infosys (2011)
17. Crowdsourced Usability Testing. http://alexcrockett.com/wp-content/uploads/downloads/Books/Crowdsourced_Usability_Testing.pdf
18. Orso, A., Rothermel, G.: Software testing: a research travelogue (2014)
19. Xie, M., Wang, Q., Yang, G., Li, M.: COCOON: crowdsourced testing quality maximization under context coverage constraint. In: IEEE 28th International Symposium on Software Reliability Engineering (2017)
20. Khedam, R., Aktouf, O., Parissis, I., Boughazi, S.: Monitoring of RFID failures resulting from LLRP misconfigurations. In: SoftCOM, pp. 1–6 (2013)