A Software Quality Framework for Mobile Application Testing

Yajie Wang, Ming Jiang, Yueming Wei
China Telecom Corporation Limited Beijing Research Institute
Beijing, China
Email: {wangyj, jiangming, weiym}@ctbri.com.cn

***Abstract*-With the explosion of mobile applications, all application providers expect to work out a popular mobile service. There are two features of a popular mobile service: adapting for mobile device's diversity and achieving high user satisfaction. For Quality Assurance (QA) testers, the former feature brings heavy testing workload and the latter claims testers try their best for good quality and good usability of the service. Therefore, improving work efficiency and test completeness are critical for mobile application QA testers. In this paper, a test framework for mobile applications is proposed, which aims to help QA testers work with high efficiency and contribute to good products with nice user experience. Moreover, a case applying this framework is presented for validating it.**

Keywords-Quality assurance; QA Tester; Mobile application; Usability

I. INTRODUCTION

At present, with the development of wireless networks and the popularization of mobile devices, mobile applications become more and more popular [1][2]. The traditional desktop software developers are putting considerable effort into the development of the mobile applications gradually. Also, telecom operators are caring more about the increase of business profits obtained from mobile applications and try their best to seek some "killer" mobile applications. With the rapid growth of mobile applications market, demands on software quality rises rapidly. The applications are expected to be stable, be quick response and have good UI experiences [3][4]. To satisfy these requirements, project team members, including software designer, developer, tester, project leader and QA member [5] should work together. Everyone should take special care of the characteristics of mobile applications and assure the typical quality of them in their working phase. In this paper, we focus on mobile software quality [5][6] only from the view of test and validation.

Most test concepts and principles of desktop software can be adopted in mobile application testing [2]. However, there are some obvious differences between software for mobile devices and desktop software [3]. The characteristics of mobile device and the complex application scenario of using applications cause the difference. As another point of view, adapting for mobile device's diversity and achieving high user satisfaction [4] are critical factors of a successful application. In desktop software, the PC, browser, connection, and context of use are so standard that even researchers do not realize or remember to mention them affecting software quality and user experience [7]. The traditional test schema is not appropriate. There should be new approaches and concerns fitting to these differences. Therefore, as QA testers, we make extra emphasis on these aspects: the mobile devices features and diversity, usage scenario in real life and user experience.

This paper proposes a systematic framework for improving software quality of mobile applications by analyzing the characteristics from all its aspects and from multiple perspectives. This paper is comprised of five sections. In Section II, we give an overview of the framework and make a brief description of its components. Section III is dedicated to describing the implementation of the components. In Section IV, a case study is presented to valid the framework. Section V concludes this paper with a summary and outlines the field of research for future work.

II. TEST FRAMEWORK ARCHITECTURE

An optimal quality assurance system for mobile software means to work in an accurate and efficient way, and to submit products with high user satisfaction. The framework proposed in the paper is composed of four components: a mobile devices information system, a defect system for mobile applications, an aggregation of key test scenarios and a mechanism for usability test.

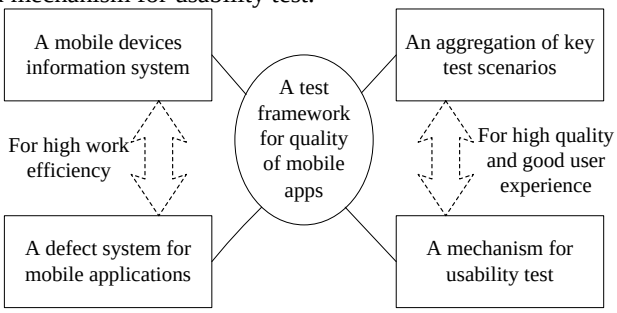

Figure 1 Architecture of the framework

The first two parts contribute to providing an accurate and efficient working condition. The mobile devices information system includes a device database and a real device management system. The database and the management system can be accessed by the whole team members. To QA testers, the database is an effective support to choose test objects and to make contrasting test plan for different devices; at the same time, the real device management system assists testers to find available devices

 ISBN: 978-1-61208-233-2

as early as possible. The defect system for mobile applications is distinguished from an ordinary defect system. It defines some special types of defects and some special attributes of a defect, which are peculiar to mobile applications. QA testers apply the defect system to accurately describe defects they found and then make it easy to be understood and dealt with among every team of the project.

The last two parts are dedicated to giving users high satisfaction. According to the characteristics of mobile device, the complex usage scenario of mobile applications and the problems easy to be neglected in mobile software testing, an aggregation of key test scenarios is defined. It collects five parts: test scenarios related to resource limitation [3], test scenarios related to imitating real usage activities, test of the server portion of the application, test of those related to charge, privacy and legacy [3], and test for good user experience. Only a mobile application verified from these five aspects can be called a valid application, not just being a software meeting service logic. A mechanism for usability test [4] describes an effective way to have a usability test. It is used by QA tests to gain usability challenge and advice from outside the application's working team, most of which are greatly valuable contributions.

III. IMPLEMENTATION OF THE FRAMEWORK

In this section, the implementation of four components comprising this framework is described in detail.

A. Structuring a Mobile Devices Information System

Diversity of mobile devices makes great difference to mobile application development and test. Unlike traditional desktop software, a good mobile application should be adapted for various devices. Large amount of work was spent on the adaption. Creating a device database to keep track of device information is a great way to improve work efficiency.

Here "database" generally refers to anything from a Microsoft Excel spreadsheet to a little SQL database [3], or any other software or system, if practicable. Scale of the database can vary according to the company's scale and cost. Devices information can be stored in the database during the requirements phase of a project or later as a change in project scope [3]. A record for a mobile device should at least include the following items:

- Important device technical specification details (screen resolution, OS version, hardware details, supported media formats, input methods, localization, any optional features, etc.)
- Any firmware upgrade or modification information, especially those related to hardware modifications.
- Any known bugs and important limitations with the device.

In addition, the information of how to get actual testing device (such as available from real device library, purchased or loaned through manufacturer or carrier loaner programs) is suggested to be recorded in the database.

For real device management, two aspects are highlighted. One is implementing a library check-in and check-out system. Team members can reserve devices for testing and development purposes. It facilitates sharing devices across teams, and then improves work efficiency greatly. The other is defining what device is a "clean" device [3] and how to return to the same starting state. At present, there is no good way to "image" a device; however, it is a basic testing policy for QA testers. There are some common ways, such as a specific uninstall process, some manual clean-up, or sometimes a factory reset. If the detailed operation steps are recorded, testers will save much time for learn and trial.

So, how do QA testers use the device information system? First, they analyze the similarity of devices and divide them into different groups, for example, grouping by the platform OS. Next, they choose the target devices according to the actual project and make test plan. Besides primary function, test priority and special test scenarios for every device should be included in the test plan. Then, QA testers use the real device management system to get the real devices rapidly. Finally, execute the test.

B. Building a Defect System for Mobile Applications

Almost all defects for desktop software may occur on mobile applications. Some typical defaults are program crashing and unexpected terminations, inadequate input validation, features not functioning expected, responsiveness problems and poor usability issues. We redefine the term defect for mobile applications from a larger range which not only includes these typical defects. Some types of defects typical on mobile applications are highlighted:

- Using too much disk space/memory on the device, not releasing memory or resources appropriately.
- Usability issues related to input methods, font sizes, and cluttered screen real estate. Cosmetic problems that cause the screen to display incorrectly [3].
- Application "not playing nicely" on the device [3], such as not compatible with other applications, overusing network resources, incurring extensive user charges, etc.
- Not handling private data securely. This includes not ensuring data safety of mobile device and server, or not guaranteeing safety of data transmission on network.
- Application not conforming to the third-party agreements, such as Android SDK license agreement (if involved), Google Maps API term (if involved), or any other terms if applied to the application.

Most defect tracking systems can be customized to work for the test of mobile applications. Whatever a system is adapted; there are some important defect attributes to be encompassed in the system, for the purpose of clarifying a mobile software defect. These attributes are:

- The application version information, language, and so on.
- Device configuration and state information including device type, platform version, network state, and carrier information.
- Steps to reproduce the problem. The steps should be described exactly using predefined standardized

 ISBN: 978-1-61208-233-2

terms, such as clear versus back, click versus tap, and so on.

- Device screenshots, which can be taken through screenshots software developed for mobile devices.

C. Defining an Aggregation of Key Test Scenarios

1) Resource limitations of mobile applications

Although mobile devices experience a massive gain in performance in recent years [2], resource limitation is still a topic we have to talk about. These limitations include devices limitations and network limitations. Devices limitations vary in terms of memory, processing power, screen type, battery level, storage capacity, platform version, input method, etc., network limitations vary in terms of accessibility and bandwidth.

To stand-alone applications, most core functions run in local memory, so testing of these applications often focuses on the limitations of the device itself. To network-driven applications, it provides a lightweight client on the device but relies on the network to provide a good portion of its content and functionality, so besides devices limitation, we also should focus network limitations when testing of these applications.

2) Imitating real usage activities

QA testers should try doing anything impossible on the mobile device, or imitating some "strange" activities when testing the application. We divide these activities into two aspects: whether compatible with other programs and how to deal with accidents.

In the real world, your application is only one of many installed on the device. You should check whether the software works well together with other device functions or applications. You should consider many things. Will your application rely on other service or content provider? Will your application act as a service or content provider? After all, the recommended way is to install some other most popular applications on the device and use them really, which can reveal integration issues that don't mesh well with the rest of the device.

Testers need to imitate real use scenarios to decrease the probability of problems found in real use. Testers must verify the common events of operating system interrupt, such as calls received, message arriving, device shutdown, etc. In addition, testers should be creative to produce certain types of events. For example, for a game, test low battery warning popping up when playing the game. Another example, for an application related to LBS (Location Based Service), step in an elevator without signal when using the application. In sum, the more you consider, the less potential problems will remain.

3) Server and service testing

Testers often focus on the client portion of the mobile application. In fact, most applications depend on a server or remote service to operate. If so, make sure thorough server and service testing is part of the overall test plan-not just the client portion implemented on the device.

Some fundamental tests, such as performance test and security test, should be covered for the application server. On this basis, QA testers should make special concern on the problems related to server upgrade, maintenance or service interruptions, because users always expect applications to be available any time. Testers should test if the users are notified when the service is unavailable and if the applications work well when the server is upgraded.

4) Related to charge, privacy and legacy

QA testers should test if an application complies with policies, protocols and agreements which the application must meet. These common agreements (if applicable) are Android License Agreement Requirements, Mobile Carrier/Operator Requirements (if applicable) and Application Certification Requirements, etc. There are some general and import rules in these agreements for QA concern: do not interfere with device phone and messaging services; do not break or exploit the device hardware and firmware; do not abuse or cause problems on operator networks.

Protection of private user data is always included in the above agreements. If your application accesses or uses private data, it is a good way to include an End User License Agreement [3] and a Privacy Policy with your application. Testers will check if this information is stored in plain text, and if it is transmitted without any safeguard.

If an application would cause the user to incur any fees, testers should test if the charge information is striking enough, if the delivery occurs when the user pays, otherwise the entire transaction is rolled back.

5) For good user experience

The first concern is installation and upgrade. QA testers should test installation on devices with low resources. If the application is available from the marketplace, you should test installation online or with the downloaded media. When a new version of the platform is released, you must re-test your application before your users are upgraded.

The second is user interface experience. QA testers may be check if screens is filled sparingly, if size graphics appropriately, and if the keys, clicks and glides are convenient.

The third is stability and responsiveness [8]. QA testers should test if the application start up fast and resume fast, if users are informed during long operations by using progress bars, and if resource consumption is reasonable.

Finally, do not forget to test features that are not readily apparent to the user, such as the backup/restore services, the sync features, and the help information.

D. Establishing a Mechanism for Usability Test

It is well known that good user experience is crucial for successful mobile applications. We just talked about test for good user experience, which is mainly a reference for your company's own QA testers to do some related tests. In this section, we introduce a mechanism for usability test which is different with the above mentioned. It is a combination of laboratory tests and field tests [9]. Laboratory tests are traditional way for usability tests, which are usually conducted in usability test laboratories, consisting of e.g. a living room or office-like area connected to a monitoring area with a one-way mirror [9]. While it is also concerned that laboratory evaluations do not simulate the context where mobile devices are used and lack the desired ecological

 ISBN: 978-1-61208-233-2

validity [9]. Interruptions, movement, noise, multitasking etc. that could affect the users' performance are not present in laboratory tests [9]. Therefore, field tests are worthwhile for mobile applications.

Both laboratory tests and field tests are through watching people actually use the application. They have several same key points in their procedures.

One is selecting of participated users. Generally, representative users are more likely to experience the same problems as the people who actually use the application [4]. So people who are representative of the target users conveniently, please do it. However, it isn't quite as important as it may seem, because many of the most serious usability problems are related to things like navigation, page layout, visual hierarchy, and so on problems that almost anybody will encounter [4]. So it's not always necessary and much more time-consuming and costly to find actual users. Whatever, it is very vital that the recruiter should be with reasonable common sense who's comfortable taking. We not only want to observe the user's action, but also want to know why to take the action.

Another is compiling test scenarios list. A good description should clarify the things you want them to try to do. Remember not to use research (unless search is being tested, of course) in the steps [4]. A pilot test of test scenarios should be done to find anything not clear in the scenario.

During the test, the observer should try to get the participants to externalize their thought process [4], and give neutral prompt to participants when encountering difficulties. Also, observers should be guaranteed to be able to observe the participant's action and words thoroughly.

Last, the debrief should take place as soon as possible after the test sessions, while what happened is still fresh in everyone's mind. Every observer can present their problems. These problems are summarized and arrayed by severity. Finally the top serious problems are chosen to be concerned primarily.

The greatest difference between laboratory tests and field tests is the context within which people uses the application. As a result, the time needed by field tests is more consuming. In general, when performing a user interface evaluation of mobile applications, laboratory tests can give sufficient information to improve the user interface and interaction of the system [9], not less than those found by field tests. While the field test method is suitable for situations where not only interaction with a system is tested, but also user behaviors and environment are examined [9]. In addition, confidentiality of the application or device in the industry often drives the decision towards the laboratory testing; especially in the beginning of the development cycle [9].

IV. CASE STUDY: TEST OF MOBILE APPLICATION OF QUESTIONNAIR SURVERY USING THIS FRAMEWORK

This section presents how the framework can be used in the mobile application of questionnaire survey.

We already have structured a devices database using MySQL and had some real devices in our test lab. This application was designed only for mobile phones of Android platform. Using the database information, QA testers choose two devices: MOTO XT800 with Android 2.0 and SAMSUNG I929 with Android 2.3. When designing the test plan, core function of the application are mainly filling the survey, submitting the survey, redeeming points and reviews; then additional test cases are designed because these two devices are customized by china telecomm; finally, several cases are designed for every device separately aiming at the differences introduced by different mobile OS version. According to a rough estimate, using the device system, we shorten the time for designing the whole test plan about 35%.

A defect system using software BugFree [10] has been built. The defects defined in Section III have been recorded in the defect tracking system. All team members can access the defect system. We also have received positive feedback about the convenience and the clarity by using the defect system.

We tested scenarios according to key points described in Section III. The application consumed a small quantity of local system resources, so no fatal problems were found about resources limitations. As for server testing, we found that if the service was closed unexpected and the client tried to connect the server, there was no obvious notification and the client kept waiting state. As to imitating real usage, an important question was found, that was while a survey was submitted, an incoming call failed. This phenomenon was obviously unreasonable. About charge and legal related field, the application is free; so, the test was simple and no problems were found. In user experience case, it was found that, in some survey, there were so many items that users had to turn pages for too many times if each page only for each item.

We invited six students to do usability test. As a final result, two reasonable advices were presented; first, the participants hoped to append progress bar or progress indicator in every page if the survey had many pages, so that the progress could be known at any time, and second, besides UC browser and Opera mobile web browser we used in the test, it was expected that QQ mobile browser [11], which is very popular in China, was adopted.

V. CONCLUSION AND FUTURE WORK

In this paper, a framework for QA testers to test mobile applications was proposed. This framework provides a helpful method to solve the question of heavy workload bought by mobile device's diversity and defines a specific defect system for mobile services to describe problems more accurately. Through our preliminary test practices, they are validated to be effective to shorten the time for designing test plan and preparation, and improve communication efficiency. Also, the framework suggests a set of test scenarios to be attended to particularly and highlights a mechanism for usability test. The benefit for product quality from them are proved in our test.

As future work, the framework should be applied in more testing of mobile applications. We should collect much more statistics to prove the benefit of the framework for mobile applications test. Also, we should refine and extend every part of the framework.

 ISBN: 978-1-61208-233-2

REFERENCES

[1] V. L. L. Dantas, F. G. Marinho, A. L. da Costa, and R. M. C. Andrade, "Testing Requirements for Mobile Applications", Computer and Information Sciences, 2009. ISCIS 2009. 24th International Symposium on, Sept. 2009, pp. 555-560, doi:10.1109/ISCIS.2009.5291880.

[2] D. Franke and C. Weise, "Providing a Software Quality Framework for Testing of Mobile Applications", 2011 Fourth IEEE International Conference on Software Testing, Verification and Validation, Mar. 2011, pp. 431-434, doi: 10.1109/ICST.2011.18.

[3] S. Conder and L. Darcey, "Android Wireless Application Development (2nd edition)", Addison-Wesley, 2011.

[4] S. Krug ,"Rocket surgery made easy", New Riders, 2010.

[5] N. S. Godbole, "software quality assurance principles and practice", Alpha Science Intl Ltd, 2004.

[6] C. Woody, N. Mead and D. Shoemaker, "Foundations for Software Assurance", 2012 45th Hawaii International Conference on System Sciences, Jan. 2012, pp. 5368-5374, doi: 10.1109/HICSS.2012.287.

[7] V. Roto, "Web Browsing on Mobile Phones-Characteristics of User Experience", Doctoral Dissertation, TKK Dissertations 49, Espoo 2006.

[8] R. Hoekman Jr., "Designing the Obvious: A Common Sense Approach to Web & Mobile Application Design (2nd Edition)", New Riders Press, 2010.

[9] A. Kaikkonen, T. Kallio, A. Kekäläinen, A. Kankainen, and M. Cankar, "Usability Testing of Mobile Applications: A Comparison between Laboratory and Field Testing", Journal of Usability Studies, Issue 1, Vol. 1, Nov. 2005, pp. 4-16.

[10] http://www.bugfree.org.cn/ [retrieved: September, 2012].

[11] http://mb.qq.com/ [retrieved: September, 2012].

 ISBN: 978-1-61208-233-2