Report about " Future Trends in Software Engineering Research for Mobile Apps"

Shahla Shaan Ahmed
BSSE0810

Atiq Ahammed
BSSE0817

Tulsi Chandra Das
BSSE0811

Moloy Kanti Sarkar
BSSE0834

25 August 2019

The objective of the paper is to make a reference for mobile application works. This paper provides a discussion on the current state-of-art of the apps based on software engineering phases requirements, development, testing as well as non functional requirements such as energy and security. With the increased use by end-users of smartphones and mobile apps and the creation of these mobile apps by software developers, mobile apps have become an interesting region for researchers in software engineering to examine. User reviews, numerical app rating and feature descriptions available in app stores give a new ground to the research works based on mobile applications.

Common challenges which are faced by the researchers are restriction on user review data, absence of source code and no trace of previous binary release of the application and notes. The challenges limits the outcome the research work.

Requirements

Most research work are mainly done on analyzing requirements and specificifications based on the leveraged user reviews. Iacob used linguistic rules to get the features and later summarized them for more abstract features. Two other research work compared the their extracted requirements with the actual application requirements and consider the extracted characteristics to be consistent and applicable to evolution functions of demands. A good number of studies focuses on feature analysis. For a big collection of applications, Finkelstein et al extract the set of characteristics from the release notes accessible in the app store. They discovered a slight correlation between an app's amount of features and an app's cost. Similarly other papers worked on a case study for feature prioritization and feature migration life cycles among apps. Some previous studies works with user complaints with the application. Few prioritizing algorithms are suggested for working with a group of reviews. Whether all users are equal or some users are more influential, however, it is essential to understand and therefore reviews may be more impactful to enforce. The authors of the paper give rises to a future topic on determining which features should be dropped.

Working with user review is pretty challenging because of the absence of the large number of reviews and qualitiful reviews. Most app stores restrict the number of user reviews. Another noteworthy challenge is applicability of the NLP techniques for extracting requirements from app reviews as they are not designed for analysing user review and requirements extraction. There is also another risk when users prefer the features given before they ask for it, and if the user complains about those,then it's too late.

Energy

Because energy (or battery) is a scarce resource for mobile devices, a number of research have suggested methods of measuring and saving energy from mobile apps. GreenMiner by Hindle et al is one of the first works in measuring energy that is a dedicated hardware platform that enables measurement of energy consumption of mobile devices. In other work, a per-instruction based energy model was suggested which estimates energy consumption to within 10% of the ground truth for Android apps. Similar GreenDroid tool and other tools can detect energy bugs. Li and others' empirical study make several interesting findings such as: 1) Most of the energy spent on a mobile app is idle and 2) networking is the heavier resource element. SOme empirical studies were performed on API calls and usage patterns and UI Hotspots on consuming energy.The two primary challenges in energy-related studies for mobile apps are not knowing what to measure to identify energy problems correctly and then attempting to solve developers ' problems. This is because developers can not easily access the current state-of - the-art tools. We therefore need decent power estimates. Access to a hardware to measure power or settle for software models that can be inaccurate and sampling issues after accessing it seems like a risk to this research area. Further research can be conducted on identifying practical ways in which energy usage can be improved in apps. Understanding how and when the authors' findings translate to other platforms could be another research area.

Security

Several recent studies have focused on Android apps security. A tangible line of job is to examine permissions in mobile applications to avoid vulnerabilities in safety.There are two research lines in the intersection of software engineering, mobile apps and security, however, in security. Identifying vulnerabilities in apps is the first line of work and finding malicious apps is the second line of research.

Most static analytical methods are well known to suffer from a high rate of false positives. However, this issue may be less critical for mobile apps as they tend to be smaller in size. Other job relies on information from the designers of the app, such as the description of the app.The availability of data is another challenge. Malicious code and code vulnerabilities are constantly changing (and at a high pace).When it comes to malware research, the ultimate goal is to build a sufficiently lightweight static analytics tool that can be deployed in the app store to prevent malicious apps from being uploaded to the store at first.

Android poses a danger as to how the suggested methods would work on other platforms for mobile apps. Another danger is to focus solely on reactive safety methods to fix present safety problems and not on preventive alternatives. Focusing on reactive approaches is a problem not only for mobile apps, but for all software.

Development

A lot of previous work has been performed on the demands, power, safety, testing and maintenance of mobile apps, and very little research has been done on the actual development of applications.One of the previous software engineering study articles was by Syer et al who compared Android and BlackBerry source code in three-dimensional apps, source code, code dependencies, and code churn. They discover that BlackBerry applications are bigger and depend highly on third-party libraries, while Android apps have fewer files and depend strongly on Android.

Several frameworks are accessible — Sencha, PhoneGap, and Titanium Appcelerator to name a few (some cross-platform development frameworks such as Cocos2d, Unity 3D, and Corona are specific to games).The developer must construct the app by calling only the APIs in these frameworks, and the framework generates an app for each platform at the moment of construction.Because of their design, all these frameworks adversely influence the efficiency of both the app and its user interface.There has been very little research to help developers understand the costs and benefits of the different approaches to cross-platform applications.

With the platforms evolving as quickly as they are to keep up with the competition, building a static solution for cross-platform development can be very difficult-the solutions need to evolve just as quickly.In some cases, mobile app developers may obfuse their apps, making their development study a challenge, as it would be necessary to deal with the obfuscation of the code before studying the app.

Testing

A wide range of studies have been developed to help developers of mobile apps improve testing of mobile apps. Those studies mainly focus on improving UI and system testing coverage.

One of the major challenges facing researchers in their current line of research on automated testing for mobile apps is that they can not reach a high level of code coverage. Main causes behind this are wide range and variety of inputs and partially which cannot be automatically generated. Often, because of the failure to produce inputs, automated testing tools are unable to continue a certain execution route, therefore, Nothing can be further tested along the execution path. Another challenge is that scientists often create instruments that operate on the binary app, as that is the only thing they have access to. With the increased success of multiple platforms there is now a large amount of cross-platform apps. In all the platforms the apps need to run on different hardware with different versions of the OS. Thus, even if one device tests the app, there is no assurance that it can work on another device.

The major risks in pursuing the research line above is that researchers may not have access to all the different devices and/or platforms. No effort was made to build such a cross-platform application database that researchers could analyze.

Maintenance

Because the mobile application is a youthful subarea within SE it continues mainly uncovered that mobile applications are maintained. Another line of work looked at bug reports for Android. Mobile bug reports, particularly for bugs linked to safety, are of high quality.

One of the difficulties of mobile app maintenance studies is that historical information is often lacking.

From past research it is found that mobile apps are small and have very quick release cycles. With such a quick release, maintenance can overlap with evolutionary work very much.

Monetization

In recent studies, ratings and downloads have often been very strongly correlated and ratings to be one of the key determinants in a user's purchase decision of an app. Although ratings may not be a sufficient requirement to download more, they may be a necessary requirement. The developers are also able to boost their sales with more downloads. Because mobile applications are often only monetized in app advertising.

Researchers in software engineering care more about the success of an app. For different people, success can mean different things. This may lead to more downloads, more users being led into a company outside of the mobile space and recognition by having a high rating. Success therefore is not just a fixed measure but a measure based on the framework of a number of feasible measures.

While there is tremendous potential to determine the success of an app, there are some risks too. The correlation in the information that has been collected as a cause can be easily misinterpreted. These factors must also be identified on the basis of intuition and motivation of common sense based on prior job.

Finally, because of the app's popularity and impact on small and large-scale developers, combined with the wealth of publicly available data, interesting research opportunities to be explored and a lively community built around it, we believe software engineering research for the mobile app is a good place for young people.