Abstract:
There are a plethora of challenges in the mobile app testing field. Manual testing is a costly venture, especially with the specializations in mobile applications so there is an increasing desire to move towards automated testing. Due to the diverse nature of mobile applications, it is difficult to create one framework that will allow for automated testing for all applications. Although it is advantageous and saves time to use automated testing, there will always be some applications that do not adhere to a single framework if implemented with our current tools and technologies. Some researchers proposed a solution to automate all mobile applications referred to as the CEL framework, which focuses on continuous, evolutionary, and large-scale principles. Despite presenting a solid architecture of the implementation, there are loose ends that may not encompass all applications. This position paper will analyze the challenges present in automating mobile application testing and demonstrate the need for manual tests in some cases. The most efficient testing to test all use-cases is through a combination of automation and manual tests.

I. Introduction

It is the position put forward that it is necessary to use a combination of automated and manual tests to effectively test all mobile applications. Our belief holds that that using both automated and manual testing allows one to take advantage of both techniques while minimizing the shortcomings of using one over the other. The goal of this paper is to present the necessity of using both automated and manual tests to effectively test mobile applications. Although the original paper reviewed presents the possibility of a single CEL framework for the automation of tests, we present specific use-cases that need to be solved with manual testing because of current tools in this field.

I. Survey of the literature

There are many challenges with implementing an overall framework for automated testing. Some of the issues in implementing this change is the current tools, organizational restrictions, and even personal choice. Furthermore, many mobile applications are based on user-input and event driven, which means they are dependent on a lot of external factors, shifts, and sensors. Cost of testing and time constraints are other issues with automation. Specifically, for mobile applications, the bugs found in this field are uncommon and usually only apply to this domain. [1]

Mobile applications need specialized research and testing according to research by Muccini et. al. The same testing strategies used on desktops and traditional softwares cannot be used on mobile applications because there are some major differences, such as mobile connectivity, limited resources, autonomy, and user interfaces. One of the challenges mentioned in testing mobile applications included GUI testing. Although time-consuming to test manually, there has been difficulty in making the task automatic while finding any errors. [2]

There exists tools and technologies for testing automation that can greatly impact the effectiveness of a testing process. Testing tools can automate certain tasks within the testing process. This can include test generation, test case execution, and the evaluation of test case results. Furthermore, automation testing tools can “support the production of useful testing documentation,” and in addition to the application source code, “provide a configuration management of this documentation” Two main categories of tools can be considered depending on the testing activity they support: 1. Tools for functional requirements testing, 2. Tools for non-functional requirement testing, such as performance, security, or usability testing. In the past few years, researchers have developed tools and

technologies to aid in the automated testing of mobile applications. One of the features offered by existing tools is performing black box testing with features like that of MobileTest, “a tool for the description and automatic execution of test cases for Java applications.” These test cases can be executed automatically on the actual device, rather than on device emulators “through a distributed run-time environment.” [3]

I. Analysis and Discussion

Despite advancements in technology, some mobile application testing aspects can only be achieved by manual testing. Furthermore, manual testing is used to provide quality products. Although quality assurance has significantly improved the speed of testing applications, there is still no replacement for manual testing methods. The end goal of testing is to release products of quality while reducing the overall time and cost. Automated mobile app testing is used in cases such as repeated execution and performance tests. However, in cases of accurate user interface testing, and replicating user experiences, it struggles to test more complicated use-cases. [4] Thus, the most efficient way to test mobile applications effectively is by incorporating both automated and manual testing.

Vasquez, Moran, and Posyvanyk presented a counter-argument by proposing a solution to automating tests through a comprehensive framework for mobile applications. It is based on CEL principles: continuous, evolutionary, and large-scale. Although they recognize current restrictions with automation, they believe their solution can provide a comprehensive framework to automation. Specifically, they aim to incorporate a change monitoring subsystem, a testing artifacts generation module, and a large scale execution engine. They mentioned more research was needed to implement the CEL framework in the future. However, once this was done, it would allow for fully automated solutions. [1]

We recognize that automation is an integral part of the testing process as mentioned by Memon [3] in the *Survey of Literature* section. Automated testing is useful for speeding up repeatable time-consuming tests. Most mobile app testing is repeatable and needs to be repeated but, in some cases, there is need for variation. This allows the testing of complex use cases that cannot be easily discovered. However, there are many drawbacks to automated testing.

Due to the nature of mobile application testing, device stability and test-specific issues can be seen as a problem. It is time consuming to debug these tests and it adds risks to the overall test coverage. Furthermore, “lab availability for testing on mobile devices, emulators, and desktop browsers” can pose additional challenges that impact automation stability. [3]

Additionally, most of all mobile applications these days depend on some sort of database. Automated tests greatly rely on the data within the database to produce repeatable results. However, there are problems with setting up this data. For example, there is “no dedicated database instance available” for various reasons that can be utilized solely for automated tests. This can be frustrating as manual usage can disturb the results from automated tests. Furthermore, the automated tests may corrupt the data of “other users” of that system.

A further challenge presented by automation is maximizing test automation coverage within an open source framework. On open source is problematic in keeping up with new technologies, such as image injection, face-ID and fingerprint authentication, and chatbots. As a result, this can cause an increase in manual testing being executed, and therefore, increases manual testing debt. [5]

In order to ensure that the billions of people in this world with smartphones are satisfied with their products, it is essential that all apps are running fine on all types of devices, mobile operating systems, and new and old versions of everything. [3] This requires a great amount of mobile testing. These days, the most apparent solution to cover it

all is through test automation. However, this is much easier said than done. Even though it is a process to make the mobile app testing experience simpler, there are many complications. It is these complications associated with test automation the prevent many top-quality assurance (QA) teams from increasing their productivity in order to release mobile apps faster, with much fewer defects.

There are three big mobile test automation obstacles that QA teams undergo when implementing or expanding mobile test automation:

1) Finding the right mobile test automation tool
    a. There are a variety of tools out there that claim they have test automation abilities. However, that does not mean they will fit every specific need. There are a few requirements a mobile automation tool must meet to enhance a mobile device's QA:
        i) The tool has to be easy to use (i.e. it must be easy enough to record an automated test script)
        ii) It needs to be able to run on "both native and object web properties," which would make the identification of the app components as consistent and precise as possible.
        iii) Easy connection to the cloud
        iv) It should allow more functional testing in order to give a thorough check of a mobile app before going to the users
2) Optimize Mobile Test Automation with Cross Platform Testing
    a) In order to meet the needs of the rapidly growing smartphone users, test automation tools "must be able to cover an expanding mobile test matrix" for mobile operating systems, devices, and the different versions of a mobile application. When an automation tool does not test all of this, the ability to cover the entire test area (acquire complete coverage) with just one automation script cannot be done. Furthermore, devices support various mobile networks and communication technologies, such as Bluetooth. This factor suggests that mobile applications require cross-platform development and testing [3]
3) Integrating into Existing DevOps or Testing Environment
    a) There is the possibility that when a test is created, it cannot be used because the mobile testing tool outputs scripts in testing environments that one is not familiar with. In this case, this question arises: should the tool be replaced, or the tester, or a combination of both? An automation tool should be able to resolve this issue by integrating with all major testing environments such as Selenium, Microsoft Visual Studio, Java, Pearl, Python, Ruby, etc.

Another difficult issue with mobile application testing is device compatibility. [4] If testers rely on an automation framework to test this, it might not pick up on issues such as the user running multiple apps at a time or if they receive a phone call. A user might send a text message or check their email in the middle of using an app. Many times, these issues with simple actions will not be picked up by an automated testing framework. Furthermore, many devices ask for permissions such as camera or microphone access. Making sure this works is easier in manual testing than automation testing where it may be missed. A very common test is for connection and navigation gestures, which again are more useful if explored through manual testing and allow the release of a higher quality application.

Manual mobile app testing is useful in cases of exploratory testing. Applications that require experiences, intuition, and creativity cannot be tested through automation with the current technology. Another challenge that cannot be solved through automation is usability testing. It is difficult to determine whether an application is user friendly without manual testing. [2] Ad-hoc testing is random testing, designed to find potential areas of error in a software. Normally, this is only effective when the testers are knowledgeable about the software, so it would be challenging to

automate. It is effective in finding bugs and uncovering loopholes in the application, which are normally tough to find.

Modern QA teams and software developers may have the mindset that everything needs to be automated. However, this leads to the issue of too much time being spent on automation testing and not enough time on exploratory testing. Agile and DevOps practices emphasize speed and automation, but it is not necessarily better to be creating more automated test cases if they are not actually useful in discovering the bugs in the code. Although automated tests may be increasing, the quality of the software is still at the same level, or even worse in some cases. Many bugs and issues commonly slip through to the production environments. If this happens enough in a specific mobile application, people might stop using the application. In agile testing environments, user stories are automated and do not go through manual testing. Another issue is that QAs will automate the test just to get it to pass, which doesn't allow them to focus on the big picture and focus on the development of the code. Although automation is good to use, many manual testing practices have declined because it is considered superior to manual testing. However, this is not the case. Manual testing is not always following a script and going through the steps. Instead in this case, manual testing can be seen as the exploratory aspect in testing. [2] This means testing the application's behavior through interesting scenarios. Some of the major problems that occur in automated testing include having useless tests, automating testing at the wrong layer, and missing key scenarios. Sometimes, the tools and time of testing is also incorrectly used with automation.

Particularly in cases where MVPs, or minimum viable products are being launched, it is smarter to use manual testing. Automated testing requires constant updating and changes, and the cost of keeping up with this would be more than the amount of benefit received. Test automation frameworks are useful in theory but end up causing more difficulties in many cases. For example, as the request for code development speeds up, and developers try to create code during their sprints, it becomes difficult to develop and test code, while also updating the test automation framework.

IV. CONCLUSIONS

Automating mobile application testing is useful in saving time and resources. However, there are also obstacles with implementing it, such as finding the right mobile test automation tool, optimizing mobile test automation with cross platform testing, and integrating testing into existing DevOps or testing environments. Furthermore, there are many instances where complex use-cases remain untested when there is no exploratory manual testing. Through the use of exploratory testing, QA teams are able to properly test device compatibility, usability and use intuition to discover errors in the applications. Many mistakenly believe one type of testing is superior over the other, when in reality, manual testing and automated testing enhance one another when used in combination.

References

[1] M. Linares-Vasquez, K.Moran, D. Poshyvanyk, "Continuous, Evolutionary, and Large-Scale: A New Perspective for Automated Mobile App Testing," in *ICSME'17* 2017.
[2] H. Muccini, A.D. Francesco, P. Esposito, "Software Testing of Mobile Applications: Challenges and Future Research Directions," Zurich, Switzerland, 2012. [Online]

[3] A. Memon, “Testing Android Mobile Applications: Challenges, Strategies, and Approaches,” in *Advances in Computers,* 1st ed. Academic Press, 2013, ch. 1. [Online]. Available: https://proquest-safaribooksonline-com.proxy01.its.virginia.edu/book/electrical-engineering/computer-engineering/9780124080942/chapter-1dot-testing-android-mobile-applications-challenges-strategies-and-approaches/s0015tit_html#X2ludGVybmFsX0h0bWxWaWV3P3htbGlkPTk3ODAxMjQwODA5NDIlMkZzMDAxMHRpdF9odG1sJnF1ZXJ5PUJPT0s=

[4] B. Kirubakaran, V. Karthikeyani, “Mobile application testing - Challenges and solution approach through automation,” in *ICME’13* 2013.

[5] A. I. Wasserman, “Software Engineering Issues for Mobile Application Development,” in FoSER ’10. New York, NY, 2010, pp. 397–400. [Online].