Proceedings of the 2013 International Conference on
Pattern Recognition, Informatics and Mobile Engineering (PRIME) February 21-22

Mobile Application Testing – Challenges and Solution Approach through Automation

B. Kirubakaran
Department of Computer Science and Engineering,
Manonmaniam Sundaranar University,
Tirunelveli, Tamil Nadu, India
1maxkiruba@gmail.com

Dr. V. Karthikeyani
Department of Computer Science,
Tiruvalluvar Government Arts College,
Rasipuram, Namakkal, Tamil Nadu, India
2drvkarthikeyani@gmail.com

***Abstract*—By the time this paper has been presented, the mobile app landscape will have changed. New OS versions will have been released. A bunch of new devices will have hit the market. And mobile application testing will have become that much more complex and challenging for all of us. There is no doubt that mobile applications need specific testing approaches. This paper wants to investigate new directions in research on the type of testing and skills required on mobile app testing by answering the following three research questions: *(RQ1)* How mobile applications testing are so different from traditional web applications, that require specialized testing skills and techniques?, (RQ2) What are the new challenges and future trends in mobile application testing, and (RQ3) How far automation effective in testing mobile application?. We answer those questions by analyzing the current trends in mobile application development and testing, and by proposing my view on the topic.**

Keywords-mobile applications; software testing; automation;

I. Introduction

According to the latest research from our Wireless Smartphone Strategies (WSS) service, the number of smart phones in use worldwide surpassed the 1 billion-unit mark for the first time ever in the third quarter of 2012. It has taken 16 years for the smart phone industry to reach this historic milestone but the next billion might be achieved in less than three years, by 2015 [1]. International Data Corporation (IDC) predicts that 182.7 billion mobile apps will be downloaded, by 2015. That's a 1600% increase from the 10.7 billion apps downloaded in 2010 (www.idc.com).

While mobile applications were initially developed mostly in the entertainment sector, they are now touching more critical domains namely: Retail: Location-intelligent mobile commerce, Media: Magazines & newspapers going 100% digital, Travel: Mobile bookings, check-ins, maps, deals, etc, Education: Tablets and apps in classrooms, Healthcare: Patient records, physician notes, etc, Finance: Apps for real-time trading, portfolio analysis, Social: Games and social media platforms

And mobile apps are running rampant in the B2B world as well, including spaces such as: Business Apps: CRM, ERP, HR systems, Productivity: Docs, spread sheets, presentations, Collaboration: Email, IM, publishing, etc. [2] The exponential growth of this market and the criticality of the developed (or, to be developed) systems impose an increased attention to dependability aspects of applications running on them. As demonstrated in some studies [7], [9], mobile applications are not bug free and new software engineering approaches are required to test those applications [14].

But, are mobile applications different from traditional ones, so to require different and specialized new testing techniques (RQ1)? And if so, what are the new challenges and future trends on testing mobile applications (RQ2)? and How far automation effective in testing mobile application (RQ3)?

Goal of this work is to provide an answer to those research questions. For answering RQ1, we first analyzed what a mobile application is (Section II-A). We look at mobility and context-awareness as the most peculiar characteristics of mobile applications, and based on those, define two types of mobile applications. Then, we analyze how mobile applications peculiarities drive specialized testing on mobile applications' (Section II-B). Section III helps to provide an answer to RQ2 and RQ3, by identifying challenges and solutions approach through automated testing of mobile applications. We outline the most relevant aspects in systematic testing of mobile applications, followed by a brief analysis of state-of-the art solutions, and future trends. Section IV concludes this paper.

II. Mobile Application and its Implications on Testing

What makes Mobile Applications differ from the traditional ones? Are mobile applications (so) different from traditional ones, so to require different and specialized new testing techniques? While some studies have analyzed additional requirements of mobile applications not commonly found in traditional ones [14], this section wants to investigate how mobile applications testing differs from testing traditional applications.

978-1-4673-5845-3/13/$31.00©2013 IEEE

A. *What is a Mobile Application?*

A mobile application (or mobile app) is a software application designed to run on Smartphone, tablet computers and other mobile devices and/or taking in input contextual information [7]. To improve our knowledge on the topic, it's imperative to analyze the role of mobile and context-awareness computing on mobile applications.

In mobile computing an application is considered to be mobile if it runs on an electronic device that may move (e.g., mp3 readers, digital camera, mobile phones). Satyanarayanan in [12] synthetically captures the uniqueness and conceptual difference of mobile computing through four constraints: limited resources, security and vulnerability, performance and reliability variability, and finite energy source.

In context-aware computing, instead, an application is aware of the computing environment in which it runs, and adapts/reacts according to its computing, user, physical, or time context. According to A. Dey, G. Abowd [3] and Constantin Schmidt [16] contextual information can be categorized into human factors (i.e., user, social environment, and task) and physical environments (i.e., location, infrastructure, and physical conditions). Challenges in context aware computing are represented by contextual sensing (e.g., location, time, nearby objects, orientation, and social contexts), contextual adaptation/reconfiguration, context-triggered actions, and contextual resource discovery [11].

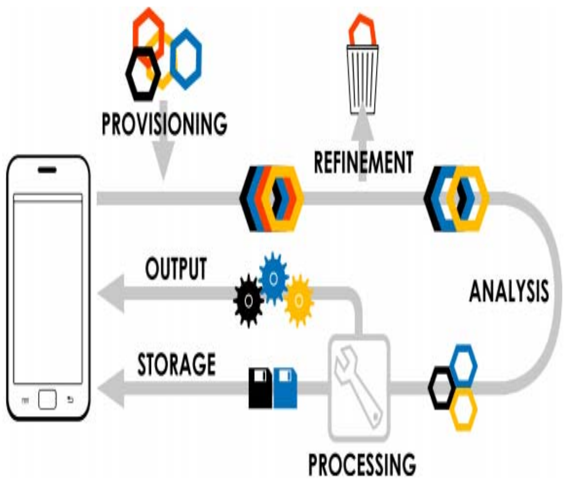

Figure 1. A Typical Context-awareness Workflow

On the whole mobile applications can be classified into two categories: Native apps, or Mobile Apps as one that is specifically designed to run on a device's operating system and machine firmware, and typically needs to be adapted for different devices. A Web app, or browser application, is one in which all or some parts of the software are downloaded from the Web each time it is run. It can usually be accessed from all Web-capable mobile devices. As a direct impact on testing, Native applications inherit the peculiarities of mobile applications (e.g., mobility, autonomy, and connectivity) [12], while Mobile Apps inherits the challenges associated to context-aware applications as well.

From a QA perspective, Henry Muccini defines Mobile apps as an application that is driven by user inputs, which runs on a mobile device with limited resources. On the other hand Web app is defined as applications that resides on server and is accessed via the Internet. It performs specified tasks potentially all the same ones as a native application for the mobile user, usually by downloading part of the application to the device for local processing each time it is used. (www.mobithinking.com/blog/what-is-a-Web-app).

B. *Mobile Applications peculiarities that drive specialized testing and Implications on Testing*

With the understanding of difference between various mobile applications, this section highlights the distinctive characteristics and their implications on Testing. The following table summarizes various peculiarities and implications discussed further of this section.

Mobile Apps Distinctive Characteristics	Implications on Testing
Connectivity	Functional, Performance, Security, Reliability testing through different networks
Convenience (Quality Design)	GUI Testing
Supported Device (Diversity of physical device and OSs)	Test Matrix based on Diversity Coverage testing
Touch Screens	Usability and Performance testing
New Programming Languages	White box and Black box testing, Byte-code analysis
Resource constraints	Functional and Performance monitoring testing
Context Awareness	Context dependent Functional testing

Figure 1. Mobile apps characteristics v Implications on Testing

C. *Characteristics of Mobile apps and Implications on Testing*

The main Characteristics are

a) Connectivity

Characteristics: mobile applications are always online as the device is constantly logged in to the mobile network that may vary in speed, reliability and security. Slow and unreliable wireless network connection with low bandwidth is a common hindrance for mobile applications.

Implications on Testing: Functional testing has to be performed in different networks and connectivity scenarios. Performance, Security and Reliability testing relies on the available connectivity type. Defects related to unreliable Wi-Fi, 3G or Bluetooth connections have been reported [8]

b) Convenience

Characteristics: convenience relates to new user interface, which needs specific guidelines for developing GUIs for mobile applications. The UI look differently depending on the physical mobile device screen resolution and dimension

Implications on Testing: GUI testing needs to be performed across various mobile devices as different devices behave differently to the same application code as claimed in [15].

c) Supported Device

Characteristics: the most difficult aspect of the testing matrix is device fragmentation due to the fact that there are hundreds of different mobile devices, with varied vendors, with different software features and hardware components. This is especially true of the Android operating system, with its seemingly countless permutations. Also the variety of mobile operating systems also poses a challenge for dev and engineering teams whose goal is to provide a consistent user experience across platforms. For example, sensors are typically calibrated differently, so that two (different) phones running the same application, in the same environment, may compute different outputs. As defined in Testing Experience Test devices – Fragmentation can be grouped in to three possible categories

Group 1, priority C: Small devices with a small CPU, RAM and low resolution. Older software versions and older browsers.

Group 2, priority B: Mid-range devices with an avg. CPU, RAM (<512 MB), good screen size and resolution. The software is not the latest.

Group 3, priority A: High-end devices with a dual/quad-core CPU, RAM (>512 MB) and a high screen resolution. Latest software versions.

Implications on Testing: Testing on varied combinations of devices with right combination of OS involves choices of testing approaches [10] as below

Manual testing vs. automated tools

In-house teams vs. outsourced partners

Guided testing vs. exploratory testing

Emulators & Simulators vs. remote access.

d) Touch Screens

Characteristics: touch screens are the main means for inputting user data into a mobile application. However, the system response time to a touch strongly depends on the device resource utilization, and may become slow in certain circumstances (e.g., entry level hardware, busy processor). Touch screens lack of responsiveness has been reported[6].

Implications on Testing: testing techniques have to be introduced to test the touch screen functioning under different circumstances (e.g., resources usages, processor load, and memory load) and within different mobile devices.

6See, for example, http://code.google.com/p/android /issues/detail?id=23382

e) New Programming Languages for Mobiles

Characteristics: programming languages of mobile applications have been designed for supporting mobility, managing resource consumption, and handling new GUIs. Objective-C, for example, uses the UIKit framework for implementing GUIs, and it is a dynamic language that can be extended at run-time. Android, instead, replaces the java.awt and java.swing APIs with new libraries, supports the bluetooth through third parties libraries, and uses Dalvik opcode compiler optimized for mobile devices [11].

Implications on Testing: traditional structural testing techniques (and associated coverage criteria) need to be revised in order to be applied to new mobile languages. Byte code analysis tools and functional testing techniques may be required to deal with binary code.

f) Resource Constraints

Characteristics: while mobile devices are becoming more and more powerful, their resources are still far away from those available on a laptop or desktop computers (e.g., the most powerful iPad can account for a modest -if compared with laptop devices- 512 MB of RAM, 64 GB of disk space, and 1 Ghz dual core CPU) [11].

Implications on Testing: mobile applications resource usage has to be continuously monitored so to avoid performance degradation and incorrect system functioning. Actions have to be taken in case of resource shortage.

g) Context Awareness

Characteristics: Mobile apps heavily rely on sensed data provided by context providers, that are sensing (like noise, light, motion, image sensors) and connectivity (like bluetooth, GPS, Wi-Fi, 3G) devices. All those devices may provide a huge amount of inputs (e.g., brightness, temperature, altitude, noise level, type of connectivity, bandwidth, neighboring devices) that vary (even unpredictably) depending on the environment and/or user actions.

Implications on Testing: testing whether the application is going to correctly work on any environment and under any contextual input is a challenge and may lead to combinatorial explosion. Context-specific test selection techniques and coverage criteria have to be produced. Bugs associated to contextual inputs are quite frequent [8].

III. Challenges in Testing Mobile Applications with Solution Approach through Automation

Software testing is an activity which is aimed for evaluating quality of a program and also for improving it, by identifying defects and problems. The testing process and its results must be repeatable and independent of the tester, i.e., consistent and unbiased. When dealing with systematic testing, different dimensions can be considered. For the purpose of this work, the focus is laid on a subset of what can be considered the most relevant aspects in systematic software testing of mobile applications. Specifically, we analyze challenges and potential future research directions on i) the type of testing process (test selection and execution), ii) the testing artifacts (structural and functional), and iii) the testing types (GUI, Device multitude, Performance, Security, Reliability and Energy and Memory testing) to be performed at different levels

Each topic is structured so to discuss what we consider the most relevant challenges first, followed by some reasoning about potential solutions and support to test automation, and the most recent state-of-the art (SOTA).

A. The Testing Process: Challenges and Solutions in Test Selection and Test Execution of Mobile Applications

The testing process consists of different activities. Those on which we focus on this paper are test selection and test execution.

- Test Selection. (Challenge): Mobile App are expected to receive inputs from different context providers (i.e., users, sensors and connectivity devices), inputs that vary from the different (and changing) contexts the mobile device can move towards. This may lead to the unpredictability and high variability of the inputs the application is potentially receiving. (Scope of testing and Automation): new testing criteria are required to provide the guidelines, rules, and strategies by which mobile test cases are selected so to maximize coverage in case of unpredictable and variable contextual inputs. Models of contextual scenarios may be used to drive the systematic selection and coverage of Mobile Apps. For example, if a tracking application making use of GPS data wants to be tested, the test suite could already include parametric inputs associated to the GPS input scenarios, e.g., the entrance/exit to/from a gallery. (SOTA): in [13] different contexts are created and used for testing purposes, but without a coverage criteria. In [8], contexts that have a higher probability to generate faults are selected. Overall, coverage criteria for context-aware inputs seem to be still missing.
- Test Execution. (Challenge): Mobile applications are, in terms of test execution, quite similar to traditional applications. In Mobile Apps, instead, contextual information leads to the challenge on how to execute test cases including rich contextual inputs. (Potentials and Automation): existing simulators are unable to simulate real-world phones with their sensors, GPS, connectivity. New capture-and-replay automated techniques can be realized for executing (as a replay) contextual inputs selected during the test selection phase. (SOTA): Android Robotium7 is an open-source tool that enables the automated and black box test execution of third parties applications. It resembles Selenium but for Android applications [11].

B. The Testing Artifacts: Challenges and Solutions in Structural and Functional Testing of Mobile Applications

Structural (code-based) or functional (model-based) testing techniques can be devised for testing mobile applications. More specifically:

- Structural. (Challenge): mobile application languages add some specific constructs for managing mobility, sensing, and energy consumption. The peculiarities of those new programming languages have to be taken into account when producing control or data flow graphs (and their respective coverage criteria) out of the mobile programming language. . (Scope of testing and Automation): New coverage criteria (and if needed, new control and data flow graphs) shall be though as a way to consider at best the new mobility, sensing, and energy constructs. In case the source code is not available, new bytecode analysis tools can be realized. (SOTA): In [6] the authors propose a structural testing approach to execute test cases on real devices. The tool JaBUTi/ME is proposed for this purpose.
- Functional. (Challenge): mobile applications functional testing requires to specify both the application and the environment it may operate in (especially in MobileApps). (Scope of Testing and Automation): state-based approaches can be particularly useful to specify the different states a mobile application may reach when different sensed data are received. We can expect to model the ranges of value an environmental variable can get, and model the impact of various environmental inputs into the application under test. We can also use state-based approaches to model different execution modes (e.g., low battery, meeting, and flying mode) the system can be. (SOTA): Android Robotium8 is an open-source tool that enables the automated and black box test execution of third parties applications. The MonkeyRunner tool 9 can run an automated functional test for Android applications. MobileTest [19] adopts a sensitive-event based approach for the automatic black box testing of software running on mobile devices.

C. Types of Testing: Challenges and Solutions in Mobile Applications Testing

Following are the different types of testing performed in Mobile application testing.

- GUI testing. (Challenge): two are the main challenges we foresee in GUI testing of mobile applications: i) to test whether different devices provide an adequate rendering of data, and ii) to test whether native applications are correctly displayed on different devices. (Scope of testing and Automation) an idea can be to automatically execute scripts that are captured during the user interaction and replayed and modified even on different devices. It is important to make this task automatic, so to avoid to manually interact with the GUI that is a time consuming task. (SOTA): a few testing approaches have been proposed for capturing GUI failures. In [15] the authors propose a technique for detecting GUI bugs by automatically generating test cases, feeding the application with random events, instrumenting the VM, and producing log/trace files and analyzing them post-run. The Android Monkey tool13 provides features for stress testing of mobile applications user interfaces
- Performance and reliability testing. (Challenge): performance and reliability of mobile applications strongly depends on the mobile device resources, on the device operational mode, on the connectivity quality and variability, and other contextual

information. (Scope of testing and Automation): new techniques for performance and reliability analysis have to explicitly consider characteristics related to (changing) contexts and different devices. Run-time analysis techniques can also be adopted to monitor the resources and connectivity state and prevent performance degradation. (SOTA): Berardinelli et al. in [20] introduce a framework for analyzing the performance of context-aware mobile software systems

- Security testing. (Challenge): security is particularly relevant due to the mobility of the device into networks with different security levels. A trojan might in fact have access to personal date, private networks, and private contextual information (e.g., where the application user is located). Moreover, the rich contextual information presents real privacy concerns. The democratization of mobile application publishers also increases the number of apps store the final user does not know anything about. (Scope of Testing and Automation): since mobility towards different networks represent a peculiarity of mobile applications, traditional security testing approaches shall be revised so to keep in consideration contextual factors, that shall be simulated so to check which data is transmitted from the mobile device. (SOTA): in [21] the authors analyze the threats Android applications pose to the security and privacy of enterprise and propose several approaches for defending enterprises against security risks.

- Device multitude testing. (Challenge): testing an application on a multitude of mobile devices is certainly an important challenge, especially in the Android O.S., where different mobile phones provide different features and hardware components, and phone manufacturers may deeply customize the O.S. Considering that as far as today there are around 130 different mobile phones running Android, seven versions of the Android OS, and assuming two firm wares per device, this will count to about 1800 different combinations. (Scope of Testing and Automation): the current "test on many devices" practice has to be replaced by (cost) effective automated techniques. Commonalities and variabilities in hardware devices and operating systems may be used for a systematic testing of the cell phone product lines. A different approach could consider to release (free of cost) instrumented beta versions of the application to be tested, have them running on a multitude of devices, and collect, store, and analyze run-time data and failures. (SOTA): A service provided by keynotedeviceanywhere.com enables users to upload their applications on their web-site and test them on various devices as per the combinations

- Memory and Energy testing. (Challenge): memory leaks may preempt the (limited) memory resource, and active processes (of killed applications) may reduce the device (battery) autonomy. (Potentials and Automation): metrics for estimating energy and memory consumption could be devised so to automatically predict memory leaks and energy outage. (SOTA): the iOS Instruments tool enables memory leaks analysis. However such analysis is not exhaustive. Some development guidelines explain how to avoid null pointer exceptions due to silent processes, but approaches and tools for testing those type of failures are still missing. A model driven engineering tool for modeling mobile software architectures successively used to generate synthetic emulation code to estimate power consumption properties. As per [10] propose a methodology to select user level test cases for performing energy cost evaluation of smartphone applications.

D. *Concluding Remarks*

The need of automation in testing mobile applications is exacerbated by two orthogonal aspects: i) Cost of testing: the common perception on mobile applications is that they must be cheap, and cost much less than traditional applications. On the other side, they must be performant, reliable and correct. Automation is certainly among the most important means for keeping the cost of testing low, while guaranteeing and adequate degree of dependability. ii) Testing through the layers: current bugs are due to interoperability problems that exist today among the application, application framework, operating system, and hardware (sensoring) layers (as also remarked in [9]). Bugs reports show how certain types of failures do not depend on buggy applications, but more on operating systems malfunctioning, or sensored data lack of precision. This remark highlights the need of testing automation towards all the different layers, and able to clearly separate application-level failures from application framework or OS failures.

Solutions that may enable cost-effective testing of mobile applications include outsourcing, cloud- and crow-based testing. We can expect that companies will start offering as-a-service software testing services, providing specialized testing skills and laboratories to make thorough testing of critical mobile applications affordable.

www.keynotedeviceanywhere.com, for example, has already launched a service to verify mobile applications on a variety of devices. Cloud- and crow based testing strategies may also enable a distributed, cost effective way of testing mobile applications on a multitude of devices.

IV. CONCLUSION

This short paper has tried to provide an overview on what testing mobile applications is and can be in the next few years. Given the first research question (RQ1) are mobile applications (so) different from traditional ones, so to require different and specialized new testing techniques? and based on what discussed in Section II and III the natural answer seems to be yes, they are. About (RQ2) what are the new challenges and solution approach on testing mobile applications?, the challenges seems to be many, related to the contextual and mobility nature of mobile applications.

Performance, security, reliability, and energy are strongly affected by the variability of the environment where the mobile device moves towards. As far as concern (RQ3) which is the role automation may play in testing mobile applications?, some potentials for automation have been outlined, being aware that a much deeper and mature study shall be conducted. In this line, we will be contacting colleagues with expertise on specific topics discussed in this paper, for making this paper a more through repository of knowledge and a reference for future research on mobile application testing.

ACKNOWLEDGMENT

The author wants to thank Dr. M. Jerald for sharing some of the results, and the anonymous reviewers for the useful insights.

REFERENCES

[1] http://blogs.strategyanalytics.com/WDS/post/2012/10/17/Worldwide-Smartphone-Population-Tops-1-Billion-in-Q3- 2012.aspx.

[2] http://www.utest.com/ext-v4/g/crowdsource-your-mobile-app-testing?ls=Banner%20Ad&cc=Pd&mc=Display-SQE-Mobile_Testing-CPA-Oct2012.

[3] Dey and G. Abowd, "Towards a better understanding of context and context-awareness," in CHI 2000 workshop on the what, who, where, when, and how of context-awareness, vol. 4. Citeseer, pp. 1–6, 2000.

[4] Cooperstock, Jeremy R., Koichiro Tanikoshi, Garry Beirne, Tracy Narine and William Buxton, "Evolution of a reactive environment," In the Proceedings of the 1995 ACM Conference on Human Factors in Computing Systems (CHI '95), pp.170-177, 1995.

[5] http://www.snet.tu-berlin.de/fileadmin/fg220/courses/WS1011/snet-project/context-aware-computing_schmidt.pdf

[6] I. Wasserman, "Software Engineering Issues for Mobile Application Development," in Proceedings of the FSE/SDP workshop on Future of software engineering research, ser. FoSER '10. New York, NY, USA: ACM, pp. 397–400, 2010. [Online]. Available: http://doi.acm.org/10.1145/1882362.1882443

[7] G. Chen and D. Kotz, "A Survey of Context-Aware Mobile Computing Research," Hanover, NH, USA, Tech. Rep., 2000.

[8] "GoogleCode Android open issues," http://code.google.com/p/android/issues/list.

[9] Martin Kropp, Pamela Morales, "Automated GUI Testing on the Android Platform", Institute of Mobile and Distributed Systems, University of Applied Sciences Northwestern, Switzerland: pp.1-2. [online]. Available: http://www.fhnw.ch/technik/imvs/publikationen/vortraege/android-testing

[10] "The Essential Guide to Mobile App Testing", http://www.utest.com/landing-blog/essential-guide-mobile-app-testing.

[11] "Software testing of mobile applications: Challenges and future research directions," http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=6228987.

[12] M. Satyanarayanan, "Fundamental Challenges in Mobile Computing," in Proceedings of the fifteenth annual ACMsymposium on Principles of distributed computing, ser. PODC '96. New York, NY, USA: ACM, pp. 1–7, 1996. [Online].Available: http://doi.acm.org/10.1145/248052.248053

[13] M. Sama, "Context-Driven Testing and Model-Checking for Embedded Contextq-Aware and Adaptive Systems," in ISSTA 2008 poster, 2008.

[14] I. Wasserman, "Software Engineering Issues for Mobile Application Development," in Proceedings of the FSE/SDP workshop on Future of software engineering research, ser. FoSER '10. New York, NY, USA: ACM, 2010, pp. 397–400. [Online]. Available: http://doi.acm.org/10.1145/1882362.1882443 http://visual.ly/mobile-application-testing-process.

[15] Hu and I. Neamtiu, "Automating GUI Testing for Android Applications," in Proc. of the 6th Int. Workshop on Automation of Software Test, ser. AST '11, pp. 77–83, 2011.

[16] Schmidt, "Implicit human computer interaction through context," Personal and Ubiquitous Computing, pp. 191–199, 2000.

[17] B. Schilit, N. Adams, and R. Want, "Context-Aware Computing Applications," in Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications. Washington, DC, USA: IEEE Computer Society, pp. 85–90, 1994. [Online]. Available: http://dl.acm.org/citation.cfm?id=1439278.1440041

[18] M. E. Delamaro, A. M. R. Vincenzi, and J. C. Maldonado, "A strategy to perform coverage testing of mobile applications," in Proc. of the 2006 Int. Workshop on Automation of Software Test, ser. AST '06, pp. 118–124, 2006.

[19] J. Bo, L. Xiang, and G. Xiaopeng, "MobileTest: A Tool Supporting Automatic Black Box Test for Software on Smart Mobile Devices," in Proc. of the Second Int. Workshop on Automation of Software Test, ser. AST '07, pp.8, 2007.

[20] L. Berardinelli, V. Cortellessa, and A. Di Marco, "Performance modeling and analysis of context-aware mobile software systems," in Fundamental Approaches to Software Engineering, ser. LNCS, D. Rosenblum and G. Taentzer, Eds.Springer Berlin / Heidelberg.

[21] X. Wei, L. Gomez, I. Neamtiu, and F. Michalis, "Malicious android applications in the enterprise: What do they do and how do we fix it?" in ICDE Workshop on Secure Data Management on Smartphones and Mobiles – SDMSM 2012.