Empirical Software Engineering (2019) 24:3249–3254
https://doi.org/10.1007/s10664-019-09776-9

EDITORIAL

Guest Editorial: Special Issue on Software Engineering for Mobile Applications

Sebastiano Panichella[1] · Fabio Palomba[2] · David Lo[3] · Meiyappan Nagappan[4]

Published online: 5 November 2019

As Andreessen stated "software is eating the world" (Andreessen 2011). Most of todays industries, from engineering, manufacturing, logistics to health, are run on enterprise software applications and can efficiently automate the analysis and manipulation of several, *heterogeneous* types of *data*. One of the most prominent examples of such software diffusion is represented by the widespread adoption of mobile applications. Indeed, during the recent years, the Global App Economy experienced unprecedented growth, driven by the increasing usage of apps and by the greater adoption of mobile devices (e.g., smartphone) around the globe. This mobile application market, which is expected in few years to double in size to $101 billion (Annie 2016), represents an attractive *opportunity* for software developers interested to build high quality and successful software applications. In such a competitive market, both "software quality" (Scherr et al. n.d.; Catolino 2018; Noei et al. 2017; Grano et al. 2017; Xia et al. n.d.; Syer et al. 2015) and overall "user experience and satisfaction" (Panichella et al. 2015; Di Sorbo et al. 2016; Grano et al. 2018) play a paramount role in the success of applications (Annie 2016; Tian et al. 2015). Thus, mobile developers interested in maximizing apps' revenue need to efficiently monitor and understand user experience and (perceived) software quality of their mobile applications, this with the help of appropriate tools and datasets (Grano et al. 2017; Panichella et al. 2015; Panichella n.d.).

✉ Sebastiano Panichella
panc@zhaw.ch

Fabio Palomba
palomba@ifi.uzh.ch

David Lo
davidlo@smu.edu.sg

Meiyappan Nagappan
mei.nagappan@uwaterloo.ca

[1] Zurich University of Applied Science, Zurich, Switzerland

[2] University of Zurich, Zurich, Switzerland

[3] Singapore Management University, Singapore, Singapore

[4] University of Waterloo, Waterloo, Canada

Given the relevance of software engineering research in mobile (Nagappan and Shihab 2016; Al-Subaihin et al. 2015) we decided in 2018 to organize a special issue on this topic. The special issue focused on software engineering research for mobile applications with studies having a strong empirical component. Specifically, we looked at studies with results obtained through any empirical approach, e.g., qualitative (involving developers), quantitative (analyzing industrial or open source data), or experimental. Moreover, we focused on innovative papers that address maintenance, testing, or monetization strategies for mobile applications, providing new ways to handle these problems or addressing them in a more unified manner, discussing benefits, limitation and costs of provided solutions. For instance, approaches, techniques and empirical studies related to innovative solutions that support developers achieve higher levels of *"software quality"* and overall *"user experience and satisfaction"* were considered of high interested for the special issue. Finally, we were also interested in experience reports reviewing software engineering practices in the context of mobile applications, e.g., studies that explore how the GUI testing strategies that have been proposed in the last years are used in practical settings, in different App Store Platforms, or on the variety of data that is created in industrial development projects.

For this special issue, we received a total of 22 submissions. All these submissions went through a rigorous peer review process, involving at least three external reviewers. Thus, relying on the reviewers' comments the guest editors made a decision about each manuscript. After a rigorous peer review cycle, out of 22 submissions, 13 papers were accepted for the special issue, 9 rejected (including one desk-rejection). The accepted papers concerned different and complementary topics: (i) the design of smart city mobile applications; (ii) the usage of text-based mobile apps similarity measurement techniques; (iii) the identification of best and bad practices related to performance and energy efficiency of mobile apps; (iv) the design of automated machine translation mechanism for app localization; (v) the analysis of repackaged apps; (vi) the characterization and detection of fake reviews in app stores; (vii) the investigation of app quality written in Kotlin; (viii) the characterization of logging practices in apps; (ix) the characterization and detection of performance anti-patterns in iOS apps; (x) the design of an evolutionary fuzzing algorithm to test proprietary Android OS system services; (xi) mining nonfunctional requirements from user reviews of mobile apps; (xii) an empirically elicited catalog of code smells for the presentation layer of Android apps; and (xiii) the mining and storing in a comprehensive database of resource leaks in Android apps.

The following papers were reviewed and accepted:

- *"Designing Smart City Mobile Applications: An Initial Grounded Theory"*. This study investigates the design process for mobile applications in the context of smart cities, which is very challenging since they need to operate within the power, processor, and capacity limitations of the mobile device. The authors conducted a multi-case study with 9 applications from 4 different development groups to build a grounded theory. The resulting theory offers explanations for how software engineering teams design mobile apps for smart cities. We think that the software engineering community benefits by this knowledge, as it will serve as a basis to further understand the phenomena and advances towards more effective design and development process definitions.
- *"Empirical Comparison of Text-Based Mobile Apps Similarity Measurement Techniques"*. This work presents an empirical study on analyzing how different text representation techniques perform when used for mobile app clustering. The authors use several state-of-the-art well-known techniques such as VSM and TF-IDF to achieve their goals.

Moreover, they investigated relevant research questions to solve this problem from multiple aspects: quantitative quality by silhouette score, qualitative quality by human judgment, and time consumption. This paper proposes several interesting directions on the mobile domain. In the mobile era, we do need to organize well the sources in the repository and the results of this article provide us some inspirations.

- *"Towards Understanding and Detecting Fake Reviews in App Stores"*. This study investigates characteristics of fake reviews and their providers, as well as how well those reviews can be detected. App stores allow users to provide feedback and positive feedback improves download and sales figures; these in turn create a market for fake reviews. To shed more light to the fake review phenomenon, the authors compared 62 million reviews from the Apple App Store and 60,000 fake reviews, as well as contacted 43 fake review providers through disguised questionnaires. The paper reports distinguishing characteristics between fake and regular reviews, as well as a summary of strategies of and offers made by fake review providers. Based on the distinguishing characteristics, the authors developed a classifier that can detect fake reviews with a high AUC/ROC value of 98%. This is a timely paper as fake reviews has become a serious problem, and this paper nicely characterizes this phenomenon as well as provides a machine learning solution to identify those fake reviews.
- *"An Empirical Study on Quality of Android Applications written in Kotlin language"*. This study first investigates the degree of adoption of Kotlin language, which was officially supported since 2017, in a collection of 2167 open-source Android applications. Next, it compares the quality of Android applications – measured based on occurrences of 10 code smells – before and after the introduction of Kotlin code. The paper reports that 11.26% of the applications include Kotlin code, and for most of them, the quality of the application after the introduction of Kotlin code improves. This study is both timely and important as Kotlin is newly introduced and findings reported here support the value of the new language in improving code quality.
- *"Studying the Characteristics of Logging Practices in Mobile Apps: A Case Study on F-Droid"*. This paper describes a case study on logging practices of 1444 open source Android apps in the F-Droid repository. The study first identifies differences between logging practices in Android apps and server/desktop applications. The study also uncovers that in 35.4% of the cases, there are inconsistencies between the chosen logging level and the developers' rationale of using logs, which may result in preventing useful information to be logged or performance overhead. Finally, the study investigates the performance overhead of logging in Android apps. This study provides new insights and paves the way to potentially many future studies that can provide systematic guidance as well as tool support to help mobile app developers in their logging practices.
- *"DroidLeaks: A Comprehensive Database of Resource Leaks in Android Apps"*. This paper first reports the definition of DROIDLEAKS, a publicly available database of 292 resource leaks found on 32 Android apps. To build such a dataset, the authors mined 124,215 code revisions and then performed an automated filtering procedure followed by a manual validation. Furthermore, to understand the nature of these leaks, the paper reports an empirical study aimed at understanding their characteristics as well as the common patterns of resource management mistakes made by developers. Finally, the authors compared eight resource leak detectors, providing a detailed analysis of strengths and weaknesses of each of them. This paper provides novel insights on the nature of resource leaks and the limitations of existing approaches for automatically detecting them.

- *"iPerfDetector: Characterizing and Detecting Performance Anti-patterns in iOS Applications"*. This work presents an empirical study on performance issues in iOS apps written in Swift language. The authors manually investigated 225 performance issues of four open-source iOS apps, finding that most of them are related to inefficient UI design, memory issues, and inefficient thread handling. Then, the paper proposes an approach, called IPERFDETECTOR, which can identify performance issues using static analysis. The authors empirically validated the approach on 11 apps and found that fixing the issues discovered by IPERFDETECTOR developers can improve the performance of their apps by up to 80%. Thus, this paper provides a usable tool that can be directly exploited by practitioners to monitor the performance of iOS applications.
- *"Catalog of Energy Patterns for Mobile Applications"*. This paper presents an empirical study aimed at classifying common practices followed by developers when improving energy efficiency. To this aim, the authors inspected commits, issues, and pull requests of 1027 Android and 756 iOS apps. Furhermore, the paper also analyzes the differences between Android and iOS devices and shows that the Android community is generally more energy-aware. This paper is intended to be relevant for practitioners, who can use the proposed catalog to learn best practices to improve energy efficiency of mobile apps. This article appears in Volume 24, Issue 4: https://link.springer.com/article/10.1007/s10664-019-09682-0.
- *"Domain-specific Machine Translation with Recurrent Neural Network for Software Localization"*. This paper proposes a neural-network based translation model specifically designed for mobile application text translation.

  To build this model, the authors collected human-translated bilingual sentence pairs inside different Android applications and customized the original RNN encoder-decoder neural machine translation model by adding categorical information addressing the domain-specific rare word problem which is common phenomenon in software text. This novel approach has been later evaluated in an empirical study involving Android apps, showing that is outperforms the general machine translation tool. This paper provides an automated solution that can be exploited by practitioners as well as a baseline model that researchers can use to further study the problem of machine translation.

- *"An Empirical Catalog of Code Smells for the Presentation Layer of Android Apps"*. This work aims at defining a novel set of code smells appearing in the presentation layer of Android applications. The paper reports on a three-step study involving 246 Android developers where they have been inquired on the problems they feel as more problematic when implementing the GUI of mobile apps. Furthermore, the authors devised an automated tool able to automatically identify the 20 code smells belonging to the defined catalog. This tool has been later used to study the diffusion of code smells in 619 Android apps. This paper provides a tool that can used by practitioners and researchers to spot design issues in mobile apps and further study the characteristics of the defined smells, respectively.
- *"Empirical Study of Android Repackaged Applications"*. This paper presents an empirical study involving more than 15,000 apps and aimed at providing insights into the factors that drive the diffusion of repackaged apps. The paper examines the motivations of developers who publish repackaged apps as well as those of users who download them. Moreover, the paper investigates the factors that determine which apps are chosen for repackaging and the ways in which the apps are modified during the repackaging process. Finding that adware is particularly prevalent in repackaged apps, this paper is intended to

stimulate researchers in further investigate the problem and possibly come up with automated solutions to deal with it.

– *"Evolutionary Fuzzing of Android OS Vendor System Services"*. To test proprietary Android OS vendor system services, this paper proposes a new platform named CHIZPURFLE that performs coverage-guided fuzzing based on evolutionary algorithms. The platform addresses an important problem as there are more than a hundred manufacturer partners that extend Android "vanilla" OS with new system services. These extensions potentially expose Android devices to reliability and security concerns. CHIZPURFLE is designed to address these concerns and comes with a key feature that allows for profiling coverage on unmodified Android device. CHIZPURFLE has been evaluated on 3 high-end commercial Android smartphones, and the results demonstrate that it is more efficient than blind fuzzing.
– *"Mining Non-Functional Requirements from App Store Reviews"*. In the past there has been considerable research on mining the reviews from mobile apps to determine the presence of bug reports or feature requests.

  However, Non-Functional Requirements (NFRs) like security or performance have not been mined from the reviews. Since, NFRs are just as important as functional requirements, in this paper, the authors propose a two-phase approach to mine NFRs from the app reviews. They carry out this study on both iOS and Android apps. In the first pahse, they qualitatively analyze reviews from iOS apps. Their analysis shows that there is at least one NFR in 40% of the 6,000 reviews. In the second phase they build an optimized dictionary-based multi-label classification approach that can extract the NFRs automatically from the reviews. In their experiment with 1,100 reviews from both iOS and Android apps, they find that they can achieve an average precision of 70% and an average recall of 86%.

In conclusion, we received interesting papers from the research community. We believe that the proposed approaches open important directions for future research in the mobile domain, and we hope researchers in the field gain much from this special section on *Software Engineering for Mobile Applications*.

Handling this special issue was a great experience. As guest editors, we express our gratitude to both reviewers and authors for making this such a high quality special section on Software Engineering for Mobile Applications. We are also grateful to the great and constant support received by the Editors-in-Chief of the journal, Tom Zimmermann and Robert Feldt, which was critical for handling this special issue.

References

Al-Subaihin AA, Finkelstein A, Harman M, Jia Y, Martin WJ, Sarro F, Zhang Y (2015) DeMobile@SIGSOFT FSE, ACM, pp. 1–2

Andreessen M (2011) Why software is eating the world. (https://www.wsj.com/articles/SB10001424053111903480904576512250915629460. August 2011)

Annie A (2016) App annie reveals future of the app economy: $101 billion by 2020; china to surpass u.s. this year

Catolino G (2018) MOBILESoft@ICSE, ACM, pp. 43–44

Cruz L, Abreu R (2019) Catalog of energy patterns for mobile applications. Empir Softw Eng 24(4):2209–2235

Di Sorbo A, Panichella S, Alexandru CV, Shimagaki J, Visaggio CA, Can-fora G, Gall HC (2016) Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, ACM, New York, NY, USA, FSE 2016, pp. 499–510. https://doi.org/10.1145/2950290.2950299

Grano G, Sorbo AD, Mercaldo F, Visaggio CA, Canfora G, Panichella S (2017) WAMA@ESEC/SIGSOFT FSE, ACM, pp. 8–11
Grano G, Ciurumelea A, Panichella S, Palomba F, Gall HC (2018) SANER, IEEE Computer Society, pp. 72–83
Nagappan M, Shihab E (2016) Perspectives on Data Science for Software Engineering, Academic Press, pp. 47–49
Noei E, Syer MD, Zou Y, Hassan AE, Keivanloo I (2017) Empir Softw Eng 22(6):3088
Panichella S. (n.d.) VST@SANER, IEEE, pp. 1–5
Panichella S, Sorbo AD, Guzman E, Visaggio CA, Canfora G, Gall HC (2015) ICSME, IEEE Computer Society, pp. 281–290
Scherr SA, Elberzhager F, Meyer S (n.d.) SWQD, Lecture Notes in Business Information Processing, vol. 338, Springer, Lecture Notes in Business Information Processing, vol. 338, pp. 45–56
Syer MD, Nagappan M, Adams B, Hassan AE (2015) Softw Qual J 23(3):485
Tian Y, Nagappan M, Lo D, Hassan AE (2015) 2015 IEEE International Conference on Software Maintenance and Evolution, ICSME 2015, Bremen, Germany, September 29–October 1, 2015, pp. 301–310. https://doi.org/10.1109/ICSM.2015.7332476
Xia X, Shihab E, Kamei Y, Lo D, Wang X (n.d.) ESEM, ACM, pp. 29:1–29:10