How Developers Micro-Optimize Android Apps
(Journal-First Abstract)*

Mario Linares-Vásquez 1, Christopher Vendome2, Michele Tufano2, Denys Poshyvanyk2
1Universidad de los Andes, Bogotá, Colombia
2College of William and Mary, Williamsburg, VA, USA

m.linaresv@uniandes.edu.co, {cvendome,mtufano,denys}@cs.wm.edu

Abstract—Optimizing mobile apps early on in the development cycle is supposed to be a key strategy for obtaining higher user rankings, more downloads, and higher retention. However, little research has been done with respect to identifying and understanding actual optimization practices performed by developers. In this paper, we present the results of three empirical studies aimed at investigating practices of Android developers towards improving apps performance, by means of micro-optimizations.

I. OVERVIEW

Every software developer has heard the famous quote by Sir Tony Hoare (promoted by Donald Knuth) that “premature optimization is the root of all evil”. In fact, this quote can be easily taken to an extreme by developers, who tend to postpone optimizing their applications until the very end of the software development cycle. However, with the advent of mobile platforms, optimizing the quality of mobile apps early is becoming a key strategy for obtaining higher user rankings, more downloads (and revenue), and higher retention.

Designers of mobile platforms outline various guidelines aimed at helping developers optimize their mobile apps. Moreover, tools for debugging and memory analysis are available to help developers detect optimization opportunities at run-time. Additionally, a number of static analysis tools, such as Lint, PMD, FindBugs, PerfChecker [2], and Relda [3] have been designed to aid developers in optimizing their apps, by detecting micro-optimization opportunities.

Some of the tools for detecting micro-optimizations opportunities have been widely adopted for Java and C/C++ systems. However, the effectiveness of the warnings provided by these tools has been previously questioned in the context of fault detection in C/C++ and Java systems [4], [5]. Moreover, little research has been done into understanding whether mobile developers perform micro-optimizations, given the fact that mobile apps are more prone to performance bugs [2].

In this paper, we present the results of three empirical studies aimed at understanding the usage of micro-optimizations in the wild, the reasons, and their impact when optimizing Android apps. First, we measured the persistence of micro-optimization opportunities in change histories of open source Android apps; in particular, we mined the change histories of 3,513 Android apps hosted on GitHub to identify the most frequent micro-optimization opportunities in 297K+ snapshots of these apps (at commit level granularity) and to understand if (and when) developers implement these micro-optimization opportunities. Second, we analyze the effectiveness of micro-optimizations suggested by two static analysis tools (i.e., PMD and LINT) on the resource consumption of Android apps by means of an in-depth analysis into whether implementing micro-optimizations can help reduce memory and CPU usage. Finally, we investigated current practices of Android developers for improving the performance of their apps by conducting a survey involving 389 participants to understand the state of practice with respect to micro-optimizing apps and their reasons.

Our findings suggest that although open source Android apps have a significant number of micro-optimization opportunities throughout their change histories, developers tend to disregard them. Regarding the value of micro-optimizations, we found that, from the analyzed micro-optimizations, removing unused resources is the only one that can actually have a large and significant improvement on the memory usage in eight open-source Android apps. Also, the lack of micro-optimization implementations might be explained by our survey results that indicate that (i) most of the developers do not know about micro-optimizations, (ii) they consider the apps too simple to require optimizations, (iii) they do not consider micro-optimizations as a worthwhile investment of their time, or (iv) they do not believe that implementing those optimizations will have any impact on the performance of Android apps.

These results contribute to understanding the nature of mobile apps and the need for specific practices that consider the context of apps; the theoretical assumption that practices with certain success in non-mobile apps can be transferred to mobile apps with the same results is not totally valid as in the case of micro-optimizations, which require specific conditions that are not common in mobile apps (e.g., large number of String operations or large number of objects instantiation in loops).

REFERENCES