End Users' Perception of Hybrid Mobile Apps in the Google Play Store

Ivano Malavolta*, Stefano Ruberto*, Tommaso Soru†, Valerio Terragni‡
*Gran Sasso Science Institute, L'Aquila, Italy - {ivano.malavolta,stefano.ruberto}@gssi.infn.it
†University of Leipzig, Leipzig, Germany - tsoru@informatik.uni-leipzig.de
‡The Hong Kong University of Science and Technology, Hong Kong, China - vterragni@cse.ust.hk

Abstract—**Today millions of mobile apps are downloaded and used all over the world. Mobile apps are distributed via different app stores, such as the Google Play Store, the Apple App Store, the Windows Phone Store. One of the most intriguing challenges in mobile apps development is its fragmentation with respect to mobile platforms (e.g., Android, Apple iOS, Windows Phone). Recently, companies like IBM and Adobe and a growing community of developers advocate *hybrid mobile apps* development as a possible solution to mobile platforms fragmentation. Hybrid mobile apps are consistent across platforms and built on web standards.**

In this paper, we present an empirical investigation into mobile hybrid apps. Our goal is to identify and analyse the traits and distinctions of publicly available hybrid mobile apps from end users' perspective. The study has been conducted by mining 11,917 free apps and 3,041,315 reviews from the Google Play Store, and analyzing them from the end users' perception perspective. The results of this study build an objective and reproducible snapshot about how hybrid mobile development is performing "in the wild" in real projects, thus establishing a base for future methods and techniques for developing hybrid mobile apps.

Index Terms—**Empirical software engineering; app store analysis; hybrid apps; Android; end-user perception;**

I. Introduction

As one billion smartphones will be sold this year, people will rely more and more on mobile apps for activities like purchasing products, messaging, etc. [36]. One of the main factors driving mobile's success is mobile apps usage (which alone makes up a majority of total digital media engagement at 52% [7]). Indeed, the mobile apps market now counts more than two million apps, downloaded billions of times per year from a number of dedicated app stores (with Google Play Store and Apple App Store as market dominators [7]).

However, programming languages and tools for developing mobile apps are platform-specific, as code written for one mobile platform (e.g., the Java code of an Android app) cannot be used on another (e.g., the Objective-C code of an Apple iOS app) [1], making the development and maintenance of native apps for multiple platforms one of the major technical challenges affecting the mobile development community [26].

In this context, *hybrid mobile apps* allow developers to use standardized web technologies such as HTML5, and distribute them in the various app stores via cross-platform wrappers and tools [24], [37]. If on one side hybrid mobile apps give numerous benefits, such as cross-platform portability, the reuse of existing knowledge of web developers, simpler and less expensive development processes [1], on the other side they suffer from a number of shortcomings such as restricted access to hardware features, variations on user experience, decrease in performance [13]. Today there is a strong debate about benefits and drawbacks in hybrid app development, with some form of limited evidence mainly coming from ad-hoc case studies and in-the-lab experiments [30], [22], [8], [13].

In this paper, we present an empirical study about the traits and distinctions of hybrid mobile apps from the end user's perspectives. The purpose of this work is exploratory: we aim at studying hybrid mobile apps in their natural setting and letting the findings emerge from the observations [39]. More specifically, the study has been conducted by (i) mining the binaries of 11,917 free apps from the Google Play Store, (ii) collecting their corresponding 3,041,315 user reviews from the store, and (iii) analysing them in terms of end users' perceived differences. In this context, directly mining the Google Play Store has been an invaluable instrument since we have been able to capture information about the apps within their real-life context.

In a previous work [29], we analysed hybrid mobile apps by mainly considering the developers' point of view, thus focussing on technical aspects, such as the used hybrid development frameworks, the use of 3rd-party web libraries, their integration to the Android platform, etc. Based on the observation that common end users do not actually have the skills, the technical background, or even the willingness to distinguish between a hybrid and a native mobile app, end users just expect the mobile app to properly work on their device (e.g., without delays, with few bugs, with a natural user experience), independently of the development framework, tool, or libraries used to implement it [19]. Under this perspective, in this paper we extend the previous work by focussing on the *end user perception* of hybrid mobile apps with respect to native ones.

The *main findings* of our study are the following: (i) hybrid development frameworks are perceived as better suited for data-intensive mobile apps, whereas they perform poorly when dealing with low-level, platform-specific features, (iii) end users value hybrid and native apps similarly, (iv) in some categories, end users perceive native apps better than hybrid apps with respect to performance and the presence of bugs.

The rest of the paper is organized as follows. In Section II introduces hybrid mobile apps, then Section III and Section IV present (i) the experimental design of our study and (ii) how we extracted and validated the data, respectively. Section V discusses the results of our study, whereas its threats to validity are discussed in Section VI. Finally, in Section VII discusses related works and Section VIII closes the paper.

II. Hybrid Mobile Apps

Mobile apps consist of binary executable files that are downloaded directly to the end user's device and stored locally [1]. When distributed via app stores, mobile apps can be of two types: native apps or hybrid apps.

Native apps are developed directly atop the services provided by their underlying mobile platform. Those services are exposed via a dedicated Application Programming Interface (API) with methods related to communication and messaging, graphics, location, security, etc. [17]. Native apps can interact with the platform API only via platform-specific programming languages (e.g., Java for Android and Objective-C for iOS).

Differently from native apps, hybrid mobile apps are developed by using standard web technologies (i.e., HTML5, CSS3, and JavaScript) and all service requests to the Platform API are mirrored by a cross-platform JavaScript API. In this context, an hybrid development framework (e.g., Apache Cordova) can be defined as a software component that allows developers to create a cross-platform web-based mobile app by providing (i) a native wrapper for containing the web-based code, and (ii) a generic JavaScript API that bridges all the service requests from the web-based code to the correspoding platform API. Thanks to the native wrapper, an hybrid mobile app can be packaged, deployed, and distributed for any supported mobile platform, like Android, iOS, or Windows Phone [1]. Among the various advantages already discussed in Section I, hybrid development frameworks help in managing one of the most recognised issues in mobile app development: portability [38]. Indeed, they allow developers to create a single mobile app using web standards, and to consistently distribute it across multiple mobile platforms with (minimal to) no changes.

III. Design of the Study

In this section we will present the research questions driving our study, and its investigation plan (i.e., its object, context, and dependent variables).

A. Research Questions

We formulate the goal of this research by using the Goal-Question-Metric perspectives (i.e., purpose, issue, object, viewpoint [11]). Table I shows the result of the above mentioned formulation.

TABLE I
Goal of this research

Purpose	Identify and analyse
Quality focus	the traits and distinctions
Object	of hybrid mobile apps
Context	in the Google Play Store
Viewpoint	from the end users' viewpoint.

In the following we present the research questions we translated from the above mentioned goal:

- **What is the difference between hybrid and native mobile apps as perceived by end users?**
  - **RQ1**: What is the difference in the *perceived value* between hybrid and native mobile apps?
  - **RQ2**: What is the difference in the *perceived performance* between hybrid and native mobile apps?
  - **RQ3**: What is the difference in the *perceived bugginess* between hybrid and native mobile apps?
  - **RQ4**: What is the difference in the *initial download overhead* between hybrid and native mobile apps?

Basically, we focus on the *end user perception* of hybrid mobile apps with respect to native ones. In this context, we identified the four main concerns that an end user may have with respect to a mobile app: its value, its performance, the presence of bugs, its initial download overhead. These concerns come from the current state of the practice, as we are continuously performing informal interviews with developers and end users[1].

B. Study planning

The **context** of our study consists of the free Android apps distributed in the Google Play Store. We decided to analyse mobile apps in the Google Play Store because of its large market share. Also, as of the third quarter of 2014, the number of downloads from the Google play Store is higher than those from other app stores[14]. Moreover, thanks to previous efforts from other researchers and developers [28], [21], downloading app binaries and apps metadata (e.g., user ratings, current version, requested permissions) from the Google Play Store is relatively straightforward.

Objects of our study are 11,917 free Android apps and their 3,041,315 user reviews automatically extracted from the Google Play Store.

In Table II we compactly show the **dependent variables** of our study, together with their corresponding research questions and scale types.

TABLE II
Dependent variables of the study

Variable name	Research question	Scale type
rating	RQ1	Ratio
reviewsPolarity	RQ1	Ratio (pair)
reviewsCount	RQ1	Ratio
performance	RQ2	Ratio (pair)
bugginess	RQ3	Ratio
size	RQ4	Ratio

Since we are considering hybrid development frameworks *from the end users' viewpoint*, we will consider the reviews and ratings provided by the end users of each mobile app. This is in line with recent research trends studying aspects related to word-of-mouth of specific products and services [34], specially in the fields of book [40], movie sales [16], and finance [12],

[1]We are working with industry partners and one of the authors is a mobile applications developer with around thirty projects in his portfolio [25].

[10]. Under this perspective, metrics such as end user ratings, reviews sentiment, and reviews scores with respect to specific aspects of the mobile app (e.g., bugginess), reflect users perceived value of the mobile app itself. In the following we will go through the dependent variables we identified for answering all the questions of this study:

• *rating*: this variable is estimated as the average rating provided by the users the mobile app as coming from the 5-stars ratings in the Google Play Store. The *rating* variable is defined as a real number in the range between 1 and 5.

• *reviewsPolarity*: it represents the polarity of sentiments of end users towards the mobile app. By building on the definition provided by Asur and Huberman [9], the reviews polarity P_a of a mobile app a is defined as:

$$P_a = \frac{pos_a - neg_a}{pos_a + neg_a} \qquad (1)$$

where pos_a is the number of end user reviews with positive sentiments, and neg_a is the number of end user reviews with negative sentiments. Section IV-A (point 5) provides the details on how we compute the sentiment of a single review.

• *reviewsCount*: based on the fact that in principles high-quality mobile apps tend to get more reviews in its app lifecycle [15], this variable represents the number of reviews of the mobile app provided by end users. Values of this variable belong to the set of natural numbers.

• *performance*: it represents the perceived performance level of the app in terms of, e.g., fast UI, quick tasks execution, etc. This variable is defined similarly to *reviewsPolarity*, where the pos_a and neg_a auxiliary functions are computed as the number of end user reviews mentioning good or bad performance of the app, respectively (Section IV-A (point 5) provides the details on this).

• *bugginess*: this variable represents a score related to the perceived presence of bugs in the mobile app. This value is estimated as the number of user reviews signalling the presence of bugs or failures in the app, normalized with respect to the total number of reviews of the app.

• *size*: file size in kilobytes of the app APK file (Android PacKage).

IV. Data Extraction, Validation, and Analysis

To allow easy replication and verification of our study, we provide to interested researchers a complete replication package. The replication package is publicly available[2] and contains all the data extracted for this study from the Google Play Store.

A. Data Extraction

Our data extraction process is composed of three main steps. Basing the discussion on Figure 1, in the following we will go through each of them.

1. Apps identification and classification. The first step of our study consists in the identification of our target population. Also, this step involves the classification of the identified apps with respect to their development strategy (i.e., native apps versus hybrid apps). In this context, we reuse an already existing dataset we produced in a previous work in which hybrid mobile apps have been analysed by mainly focussing on technical aspects, rather than specifically on end users' perception [29]. The dataset considers the top 500 most popular free apps for each category of the Google Play Store, as of November 23, 2014. This kind of selection is mainly due to the fact that performing a mere random selection of apps across the whole Google Play Store, may have resulted in a population with a large number of fake or malicious apps with few reviews [27], thus potentially leading us either to partial or misleading results. By following the guidelines in [39, §10.2], from the 27 categories of the Google Play Store, we exclude the *Widget* and *Live Wallpaper* categories because they are redundant as they are aggregations of apps belonging to other categories. Also, some apps have been removed from the dataset either because they were not available to download at the time of writing or because they have been encoded in a way that reverse engineering them is not possible. The result of this step is a list of 11,917 app IDs, each of them classified according to the identified 25 categories of the Google Play Store. Table III presents a summary of the selected apps and categories, their complete list is available in our replication package.

[2]http://cs.gssi.infn.it/ms_2015

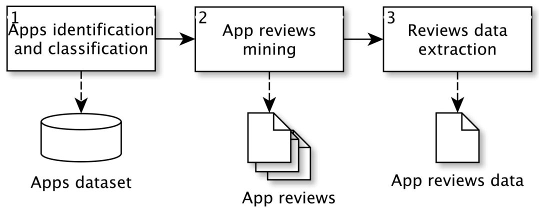

1 Apps identification and classification
2 App reviews mining
3 Reviews data extraction
Apps dataset
App reviews
App reviews data

Fig. 1. Data extraction process.

The classification of mobile apps with respect to their hybrid nature has been performed via a data-extraction tool we developed in the context of the previous work [29]. The tool is publicly available on GitHub [5] and we are actively maintaining it. Our APK data extraction tool is able to automatically get a series of information about a mobile app by analyzing the various resources contained into its APK file. In this paper, we focus on the ability of the tool to distinguish whether a mobile app is native or hybrid, and to extract its file size (see the *size* dependent variable of our study). The tool has been developed in Java and it is based on two open-source third-party libraries, that are: android-apktool [2] to decode APK files and dex2jar [4] to obtain a Jar archive from an APK file, and so making it more easy to inspect.

2. App reviews mining. We extracted and stored end user reviews of each selected app. Collecting all reviews of each app is not practical since very popular apps have millions of reviews, so we limited our data extraction on the most *helpful* ones. The *helpfulness* score of a review is provided by the Google Play Store using a free voting system (a thumbs up/thumbs down option). This score is a reasonable metric of review quality. Google Play Store presents reviews in pages with a variable number depending on their length. We ordered the reviews according to their helpfulness score and collected up to 50 of these pages. We collected a total of 3,041,315 reviews, the average number of reviews per app is 255 with a median of 132.

3. Reviews data extraction. We performed the review analysis in order to quantify variables in the second half of table II. When a review is classified as relevant to a variable, the corresponding value is updated.

To evaluate the reviews we adopted a vector space model for document representation, thus following a classical Information Retrieval approach for document indexing and searching [33]. As our aim was to build a model well-suited for keyword search, we chose a cosine similarity measure. In this paper, the cosine similarity is the dot product of two *tf-idf* vectors representing the review and the *set* of keywords of interest. These sets are textual representations of the variable semantics we then estimated. The cosine similarity thus measures the relevance of each review with respect to the different variables. The review analysis has been articulated in three main steps:

- Keywords selection: We asked two domain experts to extract keywords from 300 reviews, in order to represent the variables; As keywords are the typical variables representation in the text of the reviews, 300 reviews have been read by humans to select the significant words and expressions.
- Construction of the vectors and *cosine similarity* calculation;
- Evaluation of the reviews.

In the last step we considered relevant reviews having a cosine similarity above an experimental threshold. Whenever the variable is determined by a tuple, only the highest correspondence in the tuple is updated.

The only exception to this algorithm is the *rating* variable that is computed by averaging the rate in any single review.

In our pipeline, we used Apache Mahout [3] to compute *tf-idf* vectors and the cosine similarity. The use of a scalable software was of central importance to handle data about millions of reviews. Since our variable representatives were defined beforehand, we preferred an approach based on raw cosine similarity over unsupervised learning methods (e.g., crisp and fuzzy K-Means for clustering). In fact, while these clustering algorithms update the centroid at each iteration, our approach builds up the cluster from a given set of centroids.

B. Data Validation

In this context we aim at verifying if the extracted data is complete, reasonable and within acceptable boundaries [39]. Our validation process consists in the following checks.

1. assertion-checking [32]. We ensure that the values of our extracted variables belong to their *value domains* (i.e., set of permissible values). For example, for each app_i we performed the following assertions:

- app_i*.rating* $\in [20, 100]$
- app_i*.reviewsPolarity* $\in [-1, 1]$
- app_i*.reviewsCount* ≥ 0
- app_i*.performance* $\in [-1, 1]$
- app_i*.bugginess* $\in [0, 1]$
- app_i*.size* > 0

2. consistency-checking [32]. We also ensure that the value of one variable is logically compatible with that of some other variable.

Moreover, we performed a series of qualitative analyses to check the correctness of the extracted data. First, in order to validate the correlation among user-generated feedback, we computed the Pearson coefficient between the app average rating and the polarity of its reviews. We obtained a value of $p = 0.8$ by considering only apps having at least $k = 30$ reviews with some polarity value associated to them.

We then performed a manual evaluation on a subset of reviews to check how correctly the unsupervised analysis on the reviews has been performed. On a random test set containing 100 reviews, 87% of them were classified correctly.

V. Results

Table III shows the distribution of hybrid mobile apps in the various domain categories of the store. Overall, we have identified 445 hybrid mobile apps in our dataset, counting for a 3.73%. On one side this result clearly shows that hybrid mobile apps are still far away from being widely used in the Android apps ecosystem; on the other side, recalling that our study considers the top-500 apps within 25 Google Play categories, it shows that hybrid mobile apps are not completely neglected by top Android developers; this result may be encouraging for the future growth of hybrid development practice.

Moreover, we can notice that hybrid apps are more present in categories such as *Finance*, *Medical*, *Transportation*, *Business*, *Lifestyle*, *Social*. Interestingly, these categories share a common trait: they are all heavily based on the so-called *data-intensive mobile apps* [18]. Basically, a data-intensive mobile app differs from other mobile apps for their: (i) focus on browsing collections of data and basic interactions with data items, (ii) focus on information organization and ease of navigation, (iii) support of one-to-one content delivery, (iv) simpler transactional requirements, in most cases limited to high-performance read-only access and standard write operations, and (v) support for delivering content to multiple devices. As a confirmation, the categories containing the lowest number of hybrid mobile apps are not data-intensive, like *Photography*, *Music & Audio*, *Tools*, *Game*, and *Personalization*. We can observe that the latter categories require a closer interaction with the Android platform (e.g., photo-based apps for manipulating photos, music apps for playing songs from the device's music library, tools for launching background and system tasks, games for their performance requirements, personalization apps for customizing the standard menus and features of the device, etc.); because of their very cross-platform nature, hybrid development frameworks suffer from the lack of these capabilities, often falling back to platform-specific plugins and add-ons. This results is a clear indicator

TABLE III
HYBRID MOBILE APPS AND FRAMEWORKS DISTRIBUTION IN THE GOOGLE PLAY STORE

	Total	**Native**	**Hybrid** (%)	**Apache Cordova**	**Appcelerator Titanium**	**PhoneGap**	**Sencha**	**Kivy**	**Rho Mobile**	**IUI**	**Enyo**
Finance	463	410	53 (11.45)	23	29	1					
Medical	490	445	45 (9.18)	24	11	2	6		2		
Transportation	438	404	34 (7.76)	18	11	2	2			1	
Travel & Local	484	450	34 (7.02)	23	5	4	2				
Health & Fitness	352	331	21 (5.97)	15	6						
Libraries & Demo	418	410	8 (5.97)	6	1			1			
Business	488	459	29 (5.94)	17	3	3	6				
Lifestyle	497	468	29 (5.84)	12	11	5	1				
Social	491	465	26 (5.30)	16	3	5	2				
Sports	497	473	24 (4.83)	15	6	2	1				
Shopping	487	464	23 (4.72)	12	6	3	1		1		
Education	493	473	20 (4.06)	14	1	4	1				
Book & References	472	457	15 (3.18)	12	2						1
Communication	487	471	16 (3.29)	13		1				2	
Entertainment	487	472	15 (3.08)	10	2	3					
News & Magazines	491	478	13 (2.65)	2	11						
Comics	465	456	9 (1.94)	7				2			
Weather	495	488	7 (1.41)	4	3						
Media & Video	483	477	6 (1.24)	4	1	1					
Productivity	495	489	6 (1.21)	3	1	1		1			
Photography	494	489	5 (1.01)	4			1				
Music & Audio	477	473	4 (0.84)	1	3						
Tools	489	486	3 (0.61)	3							
Game	492	491	1 (0.20)	1							
Personalization	493	492	1 (0.20)	1							
ALL	11,917	11,470	445 (3.73)	258	116	37	23	4	3	3	1

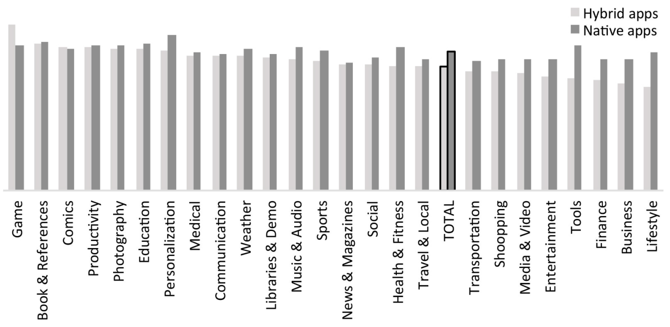
of a future area of work for developers and vendors of hybrid development frameworks.

Following the structure of research questions in Section III-A, in the following we discuss the results of our study with respect to end users' perspective.

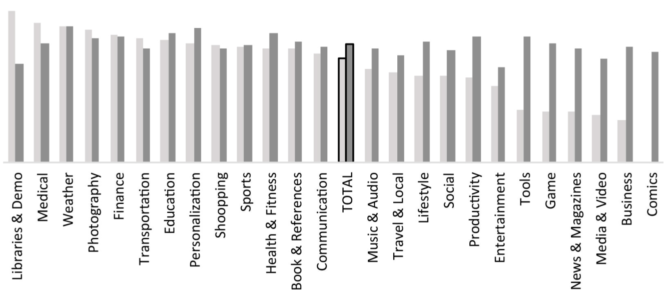
Perceived value (RQ1). As proxies to the perceived value of a mobile app we previously defined three variables, that are: *rating*, *reviewsPolarity*, and *reviewsCount*. Figures 2(a), 2(b), and 2(c) shows their average in separate charts. Values in those charts in Figure 2 focus on the difference among the values of the analyzed variables, their actual absolute value is shown here only for reference. The average of end user *ratings* for both hybrid and native apps is 3.75 and 3.35, respectively. This result does not come as a surprise because, as suggested by Hu et al. [23], end users suffer from purchasing bias, i.e., they are more likely to view their acquired product more positively since they committed the time (and money) to purchase it. More interestingly, hybrid apps and native apps are performing equally with respect to end users' star-rating across all categories, with neglectable differences. When considering the *polarity* of the sentiment of end users' reviews, we find again a certain balance between hybrid and native apps, with a slight advantage of native apps ($\sim$77.93 points against $\sim$68.60 points over the classified reviews). Nevertheless, in this chart we can also note that there is a number of categories in which the difference between hybrid and native apps is more evident (see the first six categories from the right in the chart); some of those categories, such as *Tools*, *Game*, *News & Magazines*, *Media & Video*, *Comics* are either not data-intensive or require advanced multimedia capabilities. For what concerns *app review counts*, Figure 2(c)(column TOTAL) shows that there is a relevant difference between the number of reviews of native apps with respect to hybrid apps. Indeed, in our dataset, native apps have been reviewed in average 6.5 times more than hybrid apps. By following the theory saying that end users who are reviewing a product are only doing so when they are either incredibly satisfied or dissatisfied [23], we can interpret this result as an indication of the fact that hybrid mobile apps are neither perceived as too satisfying nor dissatisfying.

Together, the three variables we discussed above let us conclude that, regardless of some exceptions (mainly concerning games and those apps that do not have a data-intensive nature), to use a hybrid development framework or to develop an app natively is not a key discriminator with respect to end users' perception of the app.

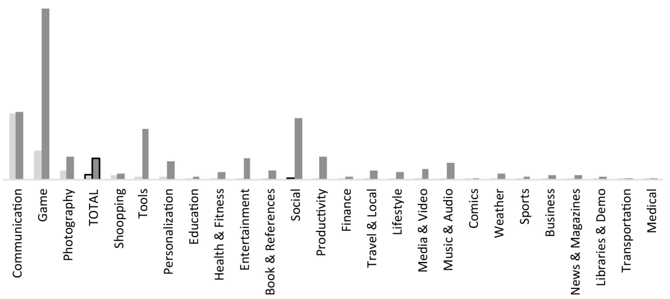
Perceived performance level (RQ2). Figure 2(d) shows the values of the *performanceLevel* variable across categories. Overall, we can notice a certain balance with respect to the difference between the perceived performance level of hybrid and native apps, as it sums up to $\sim$11.30 points in favor of native apps. Interestingly, there is great variance of the perceived level of performance across categories. Indeed, on one side we can note that *Book & References* hybrid apps are even reviewed positively, and there are less reviews signaling bad performance of hybrid apps in the *Travel & Local*, *Communication*, and *Media & Video* categories. On the other side, there are more negative reviews with respect to the



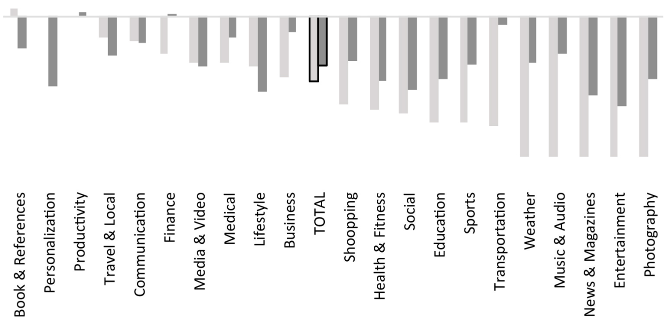
(a) Average of the ratings as provided by end users



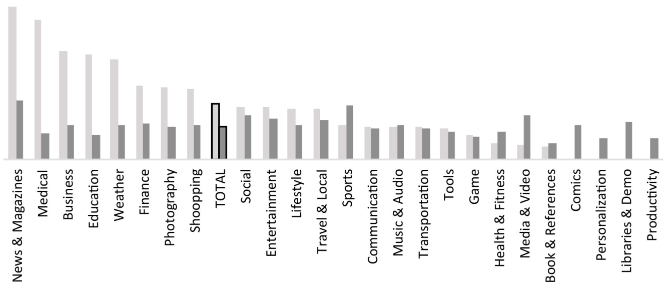
(b) Polarity of sentiment of end users



(c) Average review count



(d) Perceived performance level



(e) Perceived bugginess

Fig. 2. Hybrid and native apps as perceived by end users

performance of hybrid apps in the categories in the right-hand side of the figure, like *Shopping*, *Health & Fitness*, *Transportation*, *Photography*, etc. By looking at the collected data, we could not find any relevant correlation or interpretation between the apps' performance level and each category.

Perceived bugginess (RQ3). By referring to Figure 2(e), the difference between the perceived bugginess of hybrid and native apps sums up to $\sim$18.42 points with a higher value for hybrid apps, revealing the highest unbalance between the two development strategies in our study. In this case the advantage of native apps over hybrid apps may be rooted to the absence of full-fledged web testing frameworks specialized for hybrid apps, such as those provided by native apps IDEs like Eclipse and Android Studio. By analysing the single categories, we can see that the *News & Magazines*, *Medical*, *Business*, *Education*, and *Weather* share a large number of signalled bugs, whereas in other categories bugs have been signalled only for native apps, like in *Comics*, *Personalization*, *Libraries & Demo*, *Productivity*. According to the observed data, these differences seem to do not follow any relevant trend.

Initial download size (RQ4). We computed the median of the values of the *size* variable both for all hybrid and native apps, the result is: 6,586 kilobytes for hybrid apps and 4,625 kilobytes for native apps. These values are both in line with the average size of Android apps, irrespectively of their development strategy [6].

VI. Threats to Validity

In this section we discuss the threats to validity of our study following common guidelines [39].

Threats to external validity mainly concern the generalization of our results that relate to the representativeness of the apps considered in this work. We reduced this threat by considering a large data set, larger than recent empirical studies on the Google Play Store (e.g., [28], [35]). A random selection of all apps in the Google Play Store is likely to select poor quality apps with a small number of downloads and user reviews. By considering the most popular free apps per category we increased the chance to include the apps with the best (both current or expected) user base. In fact, free apps represent 75% of all Google Play Store apps and they are downloaded more often [20]. In addition, top apps are expected to have a high number of users because they are ranked by Google using a combination of number of downloads and aggregate user ratings. Moreover, top apps gain more exposure in the store comparing with lower ranked apps which could lead to a faster increase in the number of downloads.

Threats to construct validity refer to the degree to which the metrics used in this work actually represents the constructs in the real world. Under this perspective, we are assuming that if a review contains determinate keywords is describing an app perceived value with respect to specific aspects (e.g., bugginess, performance). However, the usage of those keywords might not be used in the intended context. We addressed this issue by estimating the keyword effectiveness for polarity values, as showed in Section IV-B. As opposite keyword sets (i.e., positive and negative) might not necessarily yield the same absolute effectiveness, the polarity or the perceived performance metrics might be skewed. This makes absolute values unreliable, however it does not affect comparisons among apps.

Reliability validity threats concern the possibility of replicating this study. We mitigated this possible threat by releasing the replication package.

Threats to conclusion validity concern the relation between the treatment and the outcome. Google Play Store categories have been considered as homogeneous entities during the analysis, however this is not the case. For example, categories like *Libraries & Demo* contain misc apps with many different functionalities. Also, the other categories have an important amount of heterogeneity. In this condition, generalizing considerations to a whole category presents some risk and is not completely appropriate.

VII. Related Work

We consider two types of research efforts as related to our study, namely: (i) studies considering app stores as their source of information for software engineering research, and (ii) studies focussing on hybrid development frameworks for cross-platform mobile app development.

A. App Store Analysis

In the last years app store analysis for research purposes is getting more and more traction. For example, the authors of [28] empirically confirmed the relationship between the success of 7,097 free Android apps and the stability and fault-proneness of the used platform APIs. In this work, the authors extracted (i) the success of a mobile app as the average (mean) rating provided by its end users, (ii) the changes performed in Android APIs by counting the number of method signature changes over time, and the number and type of exceptions raised by those methods, and (iii) the bug fixes in the Android APIs by checking the bug-fixing commits in the official Android Git commits.

The CHABADA tool [21], which is able to identify malicious app by checking the mismatch between their descriptions and their actual behaviour, has been applied on more than 22,500 Android applications mined from the Google Play Store. In this work, the data extracted from the Google Play Store include: the APK files of the apps, app names, their unique ID, and the app descriptions. The result of this analysis is the identification of the app features declared by their developers, which are in turn compared to their usage of the Android platform APIs; the latter has been computed by using a well-knwon technique for Android bytecode static analysis.

The correlation between apps launch times and their commercial success is investigated in [15], where empirical data has been collected from 3,535 apps mined from the Apple app store. In this work, the data extracted by the app store consists of the name, description, release date, price, category, and developer name of each app; in addition, the authors collected the review counts, review texts, review sentiments (analyzed via the SentiStrength tool) of each app every day during a four-months time span. The main result obtained by this study is that apps released on a day closer to weekends tend to more popular than other apps, and that users are far less sensitive to the price factor around the holiday season.

The above mentioned approaches share some similarities with our study. However, differently from those papers, our study is more focussed on web-based hybrid mobile apps (see Section II and it focusses on end users' perspectives. The size of our dataset is aligned with that of other related research.

B. Analysis of hybrid mobile apps

Research studies analysing hybrid mobile apps are emerging only recently. An observational study to provide a guide to choose the right technology for implementing a mobile app is presented in [30]. The resulting decision framework takes into consideration five dimensions: user needs, device features, development technologies, supported platforms, and development approaches. Similarly to our study, this work has an observational goal, and takes into consideration also hybrid development frameworks; however, the method proposed in [30] has been empirically tested on four case studies only (ours involves the analysis of 11,917 apps), and it aims at addressing only technical issues.

An experiment on evaluating the execution time and performance overhead between a PhoneGap-based hybrid mobile app with respect to an identical native application is reported in [13]. The results of the benchmark show that in 7 out of 8 cases, the hybrid app implementation was slower than the native one; however, the authors also noted that for general-purpose business applications, this performance issue can be considered as a slight one. Similarly to the previous described paper, the work in [13] presents a single in-the-lab experiment, with limited external validity.

An in-the-lab study about hybrid mobile apps with respect to developers' needs (e.g., used programming language, debugger, extensibility with native code, etc.) and user expectations (mainly focussing on performance issues such as app package size, required RAM, etc.) is presented in [31]. They extracted data on a (non-exhaustive) set of hybrid development frameworks from vendor documentation. Even though it is not limited to a single hybrid development framework, this work suffers from the same external validity of the previously described two approachers.

Heitkotter et al. evaluated mobile web apps, hybrid apps, and native apps with respect of a set of common criteria [22], such as license and cost, supported platforms, application speed, scalability, etc. The whole study is based on the authors' own research and experiences and on opinions from experienced developers. Again, even though the study presents interesting findings, it presents a single in-the-lab experiment, with limited external validity.

In summary, differently from past research, this paper presents the first study for characterizing hybrid apps that (i) has a solid empirical strategy, (ii) is based on a large dataset comprising 11,917 apps and 3,041,315 user reviews, and (iii) analyses hybrid apps in their actual context of use. Also, our study is one of the first investigations into hybrid mobile apps from end users' viewpoint.

VIII. Conclusion and Future Work

This paper presents detailed analysis of end users' perception about hybrid apps in their actual context of use (i.e., the Google Play Store). The results of our study show that there is still room for working on hybrid development frameworks, especially for supporting platform-specific features, and for improving their performance and testing practices. Hopefully, the above mentioned results will shed light on the current traits and distinctions of hybrid apps today, thus impacting future research, methods, and techniques for developing and managing hybrid mobile apps.

As future work, we are planning to extend our study by performing a study with a wider focus that considers at least the three most popular app stores – Apple iTunes, Google Play, and the Windows Phone stores – to compare how hybrid mobile apps perform differently on different platforms. Also, we will design and conduct a survey targeting hybrid mobile app developers with a focus on practitioners' perceived strengths, limitations and needs associated to existing hybrid development frameworks.

References

[1] Native, web or hybrid mobile-app development. *White paper, IBM Corporation*, April 2012. Document Number: WSW14182USEN.

[2] android-apktool, 2015. http://code.google.com/p/android-apktool.

[3] Apache Mahout, 2015. http://mahout.apache.org.

[4] dex2jar, 2015. http://code.google.com/p/dex2jar.

[5] Gabriele Martini, 2015. ApkCategoryChecker - http://github.com/GabMar/ApkCategoryChecker.

[6] Aapo Markkanen. Findings from Mobile Application File-size Research, 2012. ABI Research market report. Code: IN-1014787. http://www.abiresearch.com/market-research/product/1014787-findings-from-mobile-application-file-size.

[7] Adam Lella, Andrew Lipsman. The U.S. Mobile App Report, 2014. comsCore white paper.

[8] S. Amatya and A. Kurti. Cross-platform mobile development: challenges and opportunities. In *ICT Innovations 2013*, pages 219–229. Springer, 2014.

[9] S. Asur and B. A. Huberman. Predicting the future with social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 1, pages 492–499. IEEE, 2010.

[10] M. Bagnoli, M. D. Beneish, and S. G. Watts. Whisper forecasts of quarterly earnings per share. *Journal of Accounting and Economics*, 28(1):27–50, 1999.

[11] V. R. Basili, G. Caldiera, and H. D. Rombach. The goal question metric approach. In *Encyclopedia of Software Engineering*. Wiley, 1994.

[12] H. Chen. Business and market intelligence 2.0. *IEEE Intelligent Systems*, 25(1):68–71, 2010.

[13] L. Corral, A. Sillitti, and G. Succi. Mobile multiplatform development: An experiment for performance analysis. *Procedia Computer Science*, 10:736–743, 2012.

[14] Dan Crawley. VentureBeat report, 2014. http://venturebeat.com/2014/10/15/google-play-downloads-60-percent.

[15] D. Datta and S. Kajanan. Do app launch times impact their subsequent commercial success? an analytical approach. In *Cloud Computing and Big Data (CloudCom-Asia), 2013 International Conference on*, pages 205–210. IEEE, 2013.

[16] C. Dellarocas, X. M. Zhang, and N. F. Awad. Exploring the value of online product reviews in forecasting sales: The case of motion pictures. *Journal of Interactive marketing*, 21(4):23–45, 2007.

[17] B. Fling. *Mobile design and development: Practical concepts and techniques for creating mobile sites and Web apps*. O'Reilly Media, Inc., 2009.

[18] M. Franzago, I. Malavolta, and H. Muccini. Towards a collaborative framework for the design and development of data-intensive mobile applications. In *Proceedings of the 1st International Conference on Mobile Software Engineering and Systems*, MOBILESoft 2014, pages 58–61, New York, NY, USA, 2014. ACM.

[19] M. Franzago, I. Malavolta, and H. Muccini. Stakeholders, viewpoints and languages of a modelling framework for the design and development of data-intensive mobile apps. *CoRR*, abs/1502.04014, 2015.

[20] I. Gartner. Gartner says free apps will account for nearly 90 percent of total mobile app store downloads in 2012, 2012. http://www.gartner.com/newsroom/id/2153215.

[21] A. Gorla, I. Tavecchia, F. Gross, and A. Zeller. Checking app behavior against app descriptions. In *ICSE'14: Proceedings of the 36th International Conference on Software Engineering*, 2014.

[22] H. Heitkötter, S. Hanschke, and T. A. Majchrzak. Evaluating cross-platform development approaches for mobile applications. In *Web Information Systems and Technologies*, pages 120–138. Springer, 2013.

[23] N. Hu, J. Zhang, and P. A. Pavlou. Overcoming the j-shaped distribution of product reviews. *Communications of the ACM*, 52(10):144–147, 2009.

[24] M. Irvine, J. Maddocks, et al. *Enabling Mobile Apps with IBM Worklight Application Center*. IBM Redbooks, 2013.

[25] Ivano Malavolta. Online Portfolio, 2015. http://www.ivanomalavolta.com/portfolio.

[26] M. E. Joorabchi, A. Mesbah, and P. Kruchten. Real challenges in mobile app development. In *Empirical Software Engineering and Measurement, 2013*, pages 15–24, 2013.

[27] I. Juniper Networks. Third annual mobile threats report, 2014. http://www.juniper.net/us/en/local/pdf/additional-resources/3rd-jnpr-mobile-threats-report-exec-summary.pdf.

[28] M. Linares-Vásquez, G. Bavota, C. Bernal-Cárdenas, M. Di Penta, R. Oliveto, and D. Poshyvanyk. Api change and fault proneness: A threat to the success of android apps. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2013, pages 477–487, New York, NY, USA, 2013. ACM.

[29] I. Malavolta, S. Ruberto, T. Soru, and V. Terragni. Hybrid mobile apps in the google play store: An exploratory investigation. In *Proceedings of the 2nd International Conference on Mobile Software Engineering and Systems*, page To appear. ACM, 2015.

[30] E. Masi, G. Cantone, M. Mastrofini, G. Calavaro, and P. Subiaco. Mobile apps development: A framework for technology decision making. In *Mobile Computing, Applications, and Services*, pages 64–79. Springer, 2013.

[31] J. Ohrt and V. Turau. Cross-platform development tools for smartphone applications. *Computer*, 45(9):0072–79, 2012.

[32] J. Rosenberg. Statistical methods and measurement. In *Guide to Advanced Empirical Software Engineering*, pages 155–184. Springer, 2008.

[33] G. Salton, A. Wong, and C.-S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[34] J. Surowiecki. *The wisdom of crowds*. Random House LLC, 2005.

[35] S. E. S. Taba, I. Keivanloo, Y. Zou, J. Ng, and T. Ng. An exploratory study on the relation between user interface complexity and the perceived quality of android applications.

[36] University of Alabama at Birminghams Online Masters in Management Information Systems. The Future of Mobile Application, 2014. http://businessdegrees.uab.edu/resources/infographic/the-future-of-mobile-application/.

[37] J. M. Wargo. *PhoneGap Essentials: Building Cross-Platform Mobile Apps*. Addison-Wesley, 2012.

[38] A. I. Wasserman. Software Engineering Issues for Mobile Application Development. In *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research*, pages 397–400, 2010.

[39] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering*. Computer Science. Springer, 2012.

[40] Z. Zhang, X. Li, and Y. Chen. Deciphering word-of-mouth in social media: Text-based metrics of consumer reviews. *ACM Trans. Manage. Inf. Syst.*, 3(1):5:1–5:23, Apr. 2012.