




Implications on the Migration from Ionic to Android

Maria Caulo¹, Rita Francese², Giuseppe Scanniello³(✉) ,
and Genoveffa Tortora²

¹ Potenza, Italy

² University of Salerno, Fisciano, Italy
{francese,tortora}@unisa.it

³ University of Basilicata, Potenza, Italy
giuseppe.scanniello@unibas.it

Abstract. In our past research, we presented an approach to migrate apps implemented by a cross-platform technology (*i.e.*, Ionic-Cordova-Angular) toward a native platform (*i.e.*, Android). We also conducted a study to assess if there was a difference in the user experience and in the affective reactions of end-users when they used the original version of an app and its migrated version. Since we were also interested to study the perspective of developers, we successively conducted a controlled experiment to study possible differences, *e.g.*, in terms of source code comprehension and affective reactions, when developers dealt with the original and migrated versions of a given app. In this paper, we present and discuss implications from both these studies and discuss them from both researchers' and practitioners' perspectives. For example, one of the most important takeaway results from the practitioners' perspective is: it is worthy to develop an app by using a cross-platform technology (*e.g.*, for time-to-market reasons) and then to assess if this app is ready for the market; if this happens, its migration to a native technology is a good option so letting the app penetrate more the market.

Keywords: Android · Ionic · Migration · User experience

1 Introduction

Migration means transferring an application to a new target environment holding the same features as the original application [6]. Migrating applications is relevant to consolidate past knowledge and to preserve past investments [12]. We can conjecture that the use of a migrated application should not affect how the end-user perceives it as compared with its original version, in terms of performances and User Interface (UI) interaction. This is to say that one of the success factors in the migration is that the end-user does not perceive any difference when using the original and migrated apps.

M. Caulo—Independent Researcher.

© Springer Nature Switzerland AG 2021

L. Ardito et al. (Eds.): PROFES 2021, LNCS 13126, pp. 3–19, 2021.

https://doi.org/10.1007/978-3-030-91452-3_1

To reduce the development cost and time-to-market many mobile-cross-platform development technologies have been proposed [14]. Their main advantage is that apps are developed once and delivered for a number of hardware/-software platforms. Cross-platform development can be also adopted for rapid prototyping. For example, start-uppers often have to release their mobile app in a very short time on many platforms (*e.g.*, iOS, Android) and they have neither time nor money and so cross-platform development represents the only possible solution. The results from an industrial survey [15] indicated that cross-platform development is largely adopted because it is less risky than native development. Respondents in this survey also thought that a cross-platform app should be preferred when not much money can be invested in native development. Once the value of cross-platform apps has been assessed with real users (*e.g.*, through beta-testing), these apps could be re-implemented or migrated towards native platforms (*e.g.*, Android or iOS). As an example, a Stack Overflow user asks some suggestions on how to substitute an Ionic app with a native Android one in the Google Play store, because he is “*planning to start a startup and currently not in a position to afford individual development for various platforms.*”[2]. Further motivations for migrating to a native platform are represented by cross-platform development downsides [26,33], such as: (*i*) cross-platform frameworks often provide a worse user experience, (*ii*) they provide only limited access to native APIs, and (*iii*) developers must rely on the continuous development of the cross-platform frameworks to adapt changes of the changing native APIs.

In [9], we presented an approach to migrate apps implemented by cross-platform technology (*i.e.*, Ionic-Cordova-Angular) toward a native platform (*i.e.*, Android). We conducted a user study to investigate if there was a difference in the user experience¹ (UX) when using the cross-platform app version and the migrated one. We also took into account users’ affective reactions². Since we were also interested to study the perspective of developers, we successively conducted a controlled experiment to study possible differences when developers dealt with the source code of the original and migrated apps. The participants were novice developers (*i.e.*, graduate students), who were asked to comprehend, identify, and fix faults in the source code of the original and migrated apps. The results of such an experiment were presented in [8].

To summarize, we present a combined discussion of the results of our research previously presented in [8] and [9]. This allowed us to derive new results—from both the end-user’s and developer’s perspectives—that improved our body of knowledge on the relevance of approaches to migrate cross-platform apps toward a native platform considering researchers’ and practitioners’ perspectives.

¹ In the ISO 9241-210 [18], the user experience is defined as “a person’s perceptions and responses that result from the use or anticipated use of a product, system or service”. One of the most important components (*i.e.*, Usability, Adaptability, Desirability, and Value) of the user experience is usability.

² Affect is a concept used in psychology to describe the experience of feeling or emotion.

2 Related Work

Our study concerns the comparison between cross-platform and native apps. Comparing our study with those focused on performances (*i.e.*, [4, 11]), we differ from them on: (*i*) specific technologies considered (*i.e.*, design); (*ii*) aim of the study; and (*iii*) method applied. When considering those studies which compare cross-platform and native apps by means of user tests (*i.e.*, [3, 28, 30]), we differ from the studies of Malvolta *et al.* [28] and Noei *et al.* [30] for the design (*e.g.*, metrics and attributes to evaluate apps), the method used (*e.g.*, we do not mine opinions on the Google Play Store), and, also, the aims (*i.e.*, we do not only compare two versions of a given app but also evaluate our migration approach when applied on a real case). The research of Angulo and Ferre [3] might seem the most similar to that conducted in our user study: they also compared native and cross-platform tools used to deploy the same app by means of user tests. But our study differs from theirs in substantial ways. First, the aim: we do not only investigate end-users' opinions on the two versions of the app (*i.e.*, UX), but also measure their affective states (*i.e.*, Pleasure, Arousal, Dominance, and Liking). Second, the design: we have one experimental object in two versions: an Android app developed with Ionic-Cordova-Angular technologies and the same app implemented in native Android, while they have one experimental object in four versions: the native ones (Android and iOS) and the cross-platform ones (Titanium Android, and Titanium iOS). Then, the involved participants used their own smartphone, while we made the participants use the same smartphone; moreover, we did not make the participants execute specific tasks, while they did. The GUI of the native and the Titanium version (in both the cases of iOS and Android) significantly differ from one another, while in our case the GUIs are very similar, in a way that it was hard for the participants to distinguish one version from the other. Finally, the cross-platform technology they adopted was Titanium, while we use Ionic-Cordova-Angular technologies. As a consequence of all these differences, their conclusions involve several comparisons among Android and iOS (both native and Titanium), while in our case we focus on the differences between the native Android version and the Ionic-Cordova-Angular Android version. Concerning the developers' perspective, we do not focus on the selection of the most convenient cross-platform technology to suggest to developers, as made in [13, 14, 16, 17, 33], as well as we do not investigate the challenges concerning the developing phase, but we compare affective reactions that developers feel while executing three tasks on existing source code of the two aforementioned versions of the same app and also measure differences on the correctness of the results of their tasks. The work by Que *et al.* [32] might seem the most similar to that conducted in our controlled experiment but the most important differences concern the method used to conduct their research. Indeed, they did not execute an experiment with practitioners to make comparisons between cross-platform and native technologies in terms of ease of coding, debug/test, and distribution stage, but they make comparisons in theory.

3 Overall Assessment

The main RQ we investigated in our integrated study follows:

How does the migration of mobile applications from the Ionic Framework to Native code impact the end-user's and developer's experience/satisfaction?

The rationale behind this RQ is that: when migrating a cross-platform app towards a native platform, it is advisable that the UX and affective reactions improve when end-users deal with the migrated version of the app, and the maintenance and the evolution are easier for the migrated version of the app (or at least comparable). To this end, we conducted two empirical investigations: (i) a user study and (ii) a controlled experiment. The former aimed at comparing the affective reactions of end-users and their UX when using a cross-platform app and its migrated version. The latter aimed at studying if there is a difference when comprehending source code and performing fault fixing tasks on a cross-platform app and its migrated version.

To conduct both the investigations, we followed the guidelines by Wohlin *et al.* [40] and Juristo and Moreno [20]. We reported these studies on the basis of the guidelines suggested by Jedlitschka *et al.* [19]. In this paper, we limited ourselves to the presentation of the main aspects of both studies. The interested reader can find more details in our technical report (TR), available at bit.ly/3xIjjSJ.

3.1 User Study

In this section, we present the design and the results of our user study. The goal of this study was to evaluate the difference (if any) between apps developed by using cross-platform and native technologies from the perspective of the end-users. We compared the original version of an app, namely Movies-app [1], with that migrated to Android (see TR for details) in terms of affective reactions (*i.e.*, Pleasure, Arousal, Dominance, and Liking) of end-users, as well as the UX that they reached. If we observe a difference in favor of the Android migrated version of Movies-app with respect to these two aspects, we can speculate that the migration from Ionic-Cordova-Angular to Android impacts the end-user's experience.

Experimental Units. Initially, 19 people accepted to take part in the study, while 18 actually participated. The background of the participants can be summarized as follows: 12 people had a Bachelor Degree (ten in Computer Science and two in Mathematics); four people had a Master's Degree (three in Computer Engineering and one in Mathematics); one had a Ph.D. in Computer Science and one had a Scientific High School Diploma. Except for the latter participant, all the others were graduate students at the University of Basilicata.

Experimental Material and Tasks. The experimental objects (as already mentioned) consisted of the two versions of Movies-app: the original one and

the migrated one. The experiment referred to a version of Movies-App which was available on GitHub. It is worth mentioning that the official Ionic website has recently adopted Movies-App as a demo app.³ We asked the participants to freely use both versions of Movies-app.

We collected affective reactions by requiring participants to filling in the SAM [5] questionnaire. It considered the following dimensions evaluated on a nine-point scale for Pleasure, Arousal, and Dominance. As Koelstra *et al.* [25] did, we included the Liking dimension.

As for UX, we relied on the 26 statements by Laugwitz *et al.* [27]. These authors defined these statements to evaluate the quality of interactive products (*e.g.*, software). Each statement is made of two adjectives that describe some opposite qualities of products (*e.g.*, annoying and enjoyable). According to their objectives, these statements are grouped into the following six categories: Attractiveness, Perspicuity, Efficiency, Dependability, Stimulation, and Novelty.

Experiment Design. We opted for a *factorial crossover* [38] design. The number of periods (*i.e.*, Order) and treatments (*i.e.*, Technology) is the same and the treatment is applied only once [38]. We randomly assigned the participants into two groups, G₁ and G₂, both made of nine members. Each participant used both versions of the app. Participants in G₁ firstly used the Android version and then the Ionic-Cordova-Angular one, while vice-versa for G₂.

Hypotheses and Variables. We considered two independent variables: *Technology* and *Order*. The first indicates the technology used to implement the app. Therefore, Technology is a categorical variable with two values: Android and Ionic (abbreviating Ionic-Cordova-Angular). The Order variable indicates the order in which a participant used the version of the app (also known as *sequence* in the literature).

To measure affective reactions, we used four dependent variables (one for each dimension of SAM plus Liking). To measure UX, we used six dependent variables, one for each of the six categories of UEQ (User Experience Questionnaire), *e.g.*, Attractiveness. To obtain a single value for each category we summed the scores of each statement in that category. This practice to aggregate scores from single statements is widespread [39]. We formulated and tested the following parameterized null hypothesis.

- H_{0X} : *There is no statistically significant difference between the Android and Ionic Apps with respect to X.*

Where X is one of the dependent variables (*e.g.*, Liking).

Procedure. We performed the following sequential steps.

1. We invited Ph.D. and Master’s students in Computer Science and Mathematics at the University of Basilicata and students enrolled in the course of

³ <https://ionicacademy.com/ionic-4-app-api-calls/>.

Advanced Software Engineering of the Master’s Degree in Computer Engineering from the same University. We also invited people working in the Software Engineering Laboratory at the University of Basilicata. They had to fill in a pre-questionnaire to gather demographic information. This design choice allowed us to have participants with heterogeneous backgrounds.

2. We randomly split the participants into two groups: G_1 and G_2 .
3. The study session took place under controlled conditions in a research laboratory.
4. Depending on the group, each participant freely used a version of the Movies-app and then filled in the SAM+Liking questionnaire (first) and UEQ (later).
5. Each participant freely used the other version of the Movies-app and then filled in the SAM+Liking questionnaire (first) and UEQ (later).

All the participants used the same smartphone⁴ when using both the versions of Movies-app.

Analysis Procedure. To test the null hypothesis, we used the ANOVA Type Statistic (ATS) [7]. It is used (*e.g.*, in Medicine) to analyze data from rating scales in factorial designs [22]. We built ATS models as follows:

$$X \sim \text{Technology} + \text{Order} + \text{Technology} : \text{Order}. \quad (1)$$

Where the dependent variable is X and Technology and Order are the manipulated ones. Technology:Order indicates the interaction between Technology and Order. This model allows determining if Technology, Order, and Technology:Order had statistically significant effects on a given dependent variable X. In the case of a statistically significant effect of a factor, we planned to use Cliff’s δ effect size. It is conceived to be used with ordinal variables [10] and assumes the following values: *negligible* if $|\delta| < 0.147$, *small* if $0.147 \leq |\delta| < 0.33$, *medium* if $0.33 \leq |\delta| < 0.474$, or *large* if $|\delta| \geq 0.474$ [34].

To verify if an effect is statistically significant, we fixed (as customary) α to 0.05. That is, we admit 5% chance of a Type-I-error occurring [40]. If a p-value is less than 0.05, we deemed the effect as statistically significant.

3.2 User Study Results

Android vs. Ionic with Respect to Affective Reactions of End-Users.

Median values seem to suggest that the participants in both G_1 and G_2 obtained more positive affective reactions when dealing with the migrated version of Movies-app. The smallest difference between Ionic and Android is for the Pleasure dimension. The median values are seven and 6.5, without considering the participants’ distributions between the groups, respectively. As for G_1 the median values are the same, *i.e.*, 7, whatever is the Technology. As for Liking,

⁴ Umidigi A3, a Dual-Sim smartphone equipped with Android 8.1.0, 5.5” screen with 720×1440 resolution points, 3300mAh capacity battery, 2 GB RAM, 16 GB of expandable memory, MediaTek MT6739 processor.

Table 1. Median values for Attractiveness, Perspicuity, Efficiency, Dependability, Stimulation, and Novelty.

	Android						Ionic					
	Attract.	Perspicuity	Efficiency	Depend.	Stimulation	Novelty	Attract.	Perspicuity	Efficiency	Depend.	Stimulation	Novelty
G ₁	33	27	23	23	19	18	28	24	17	18	18	16
G ₂	23	25	23	21	18	14	28	26	18	22	18	12
Total	32.5	26.5	23	22.5	19	17	28	24.5	17.5	19	18	14

the difference between the two versions of Movies-app (without considering the participants' distributions between the groups) is very clear: eight for Android and six for Ionic. The median values for G₁ are eight for Android and seven for Ionic, while the median values for G₂ are six for Android and five for Ionic.

The results of our statistic inference suggest a statistically significant difference with respect to Liking. Therefore, we can reject $H0_{Liking}$ ($p - value = 0.0494$) with a *medium* effect size (0.383), and then we can postulate that the participants liked more the migrated version of Movies-app than its original version. As for Liking, we also observed a significant interaction between the two independent variables. This interaction is significant also for Pleasure. This means that there is a combined positive effect of Technology and Order for these two dependent variables.

We observed that there is a slight preference for the app migrated to the Android platform, although this is statistically significant only for the Liking dimension with a medium effect size.

Android vs. Ionic with Respect to the UX. In Table 1, we report the median values for the dependent variables measuring the UX grouped by G₁ and G₂. The median values by grouping observations only considering Technology are also shown (*i.e.*, Total row). The median values seem to suggest that the participants, in general, expressed more positive UX when dealing with the migrated version of Movies-app. The smallest difference between the two versions of this app can be observed for the category Stimulation. Descriptive statistics also show the following pattern: the participants in G₁ (those administered first with the Android version of Movies-app) were more positive with respect to the migrated version of the app as compared with the participants in G₂ (those administered first with the Ionic-Cordova-Angular version). The only exception is Efficiency since the values are 23 for both groups.

The results our statistic inference indicate a statistical significant difference with respect to Efficiency ($p - value = 0.0004$) with a *large* effect size (0.67). Therefore, we can assert that the participant found the Android version of the app to be more efficient than its original version since we were able to reject the null hypothesis $H0_{Efficiency}$. We also observed a significant interaction between Technology and Order for Novelty. This means that there is a combined positive effect of these two independent variables.

The UX is better for the app migrated to the Android platform although the effect of Technology is significant only for Efficiency, where the size of the effect is large.

3.3 Controlled Experiment

If through the controlled experiment we observe a difference in the source code comprehensibility and fault identification and fixing when dealing with the two versions of Movies-app, we can speculate that the migration from Ionic to that platform impacts the developers’ experience. We were also interested in assessing how developers perceive source code comprehension tasks and fault identification and fixing tasks. Therefore, we also focused on both affective reactions. A positive (or negative) effect of a technology (Ionic-Cordova-Angular vs Android) with respect to affective reactions might imply that a developer is more (or less) effective when performing these kinds of tasks.

Experimental Units. The participants were 39 students of the “Enterprise Mobile Applications Development” course at the University of Salerno (Italy). This course focused on the study of Ionic-Cordova-Angular technologies. The average age of participants was 24. At the time of the experiment, participants were 39 months (on average) experienced with programming and ten months (on average) experienced with mobile programming, in particular. They passed the programming exams with a rating of 27.4/30 on average. The participants before the “Enterprise Mobile Applications Development” course passed the “Mobile Development” course, which was focused on Android.

Experimental Material. We used the source code of two versions of Movies-app. Its code is not very complex and it is small enough to allow good control over participants. The problem domain of this app can be considered familiar to the participants. The reader can find further details in our TR. We used the SAM [5] questionnaire to gather affective reactions of participants when accomplishing comprehension and fault identification and fixing tasks. We also included the Liking dimension in addition to those considered in the SAM questionnaire.

Tasks. We asked the participants to perform three tasks in the following order:

1. *Comprehension Task.* We defined a comprehension questionnaire composed of six questions that admitted open answers. The questions of this questionnaire were the same for both the groups of participants; those administered with the source code of the Ionic-Cordova-Angular version of Movies-app and those administered with the source code of its migrated version.
2. *Fault Identification.* Similar to Scanniello *et al.* [35], we seeded (four) faults in the source code of the two versions of the app. We asked the participants to fix these faults providing them with a fault report for each seeded one. The bug report was the same independently from the app version. We seeded faults by applying mutation operators (*i.e.*, predefined program modification rules) by Kim *et al.* [23]. We asked the participants to document where they believed each fault was in the source code. It is worth mentioning that we seeded faults in the source code which the participants did not have to analyze to answer the

questions of the comprehension questionnaire. However, we could not prevent that the participants could have analyzed the faulty source code during the comprehension tasks. This threat to conclusion validity equally affects fault identification (and fixing) results for both the versions of Movies-app.

3. *Fault Fixing*. Participants had to fix the faults they identified. We asked them to work with a fault at a time. Faults do not interfere with one another.
4. *Post questionnaire*. It included a SAM+Liking questionnaire for each task the participants accomplished.

Hypotheses and Variables. As made for the user study, we considered Technology as the independent variable (or manipulated factor). This variable indicates the technology with which the app was implemented. As for the source-code comprehension task, we used *Comprehension* as the dependent variable. It measures the correctness of understanding of a participant given a version of Movies-app by analyzing the answers provided to the comprehension questionnaire. We used an approach based on that by Kamsties *et al.* [21] that computes the number of correct responses to the questions of that questionnaire. The dependent variable *Comprehension* assumes values between zero and six (*i.e.*, the number of questions in the comprehension questionnaire). A value close to six indicates that a participant comprehended the source code very well. A fault is successfully identified if the participant correctly marked the source code where the fault was seeded. We named the variable counting the faults correctly identified as *Correctness of Fault Identification*. This variable assumes values between zero and four (*i.e.*, the number of faults). The higher the value the better it is. As for the fault fixing task, we defined the variable: *Correctness of Fault Fixing*. It counts the number of seeded faults the participants correctly fixed in the source code of the experimental object. Also, *Correctness of Fault Fixing* assumes values between zero and four (*i.e.*, the number of faults). The higher the value the better it is.

As for affective reactions, we considered four dependent variables (one for each dimension of SAM plus the Liking one) for each kind of task the participants performed: comprehension, fault identification, and fault fixing.

We tested the following parametrized null hypothesis.

- $H1_X$: *There is no statistically significant difference between the participants who were administered with the cross-platform and the native versions of Movies-app with respect to X (*i.e.*, one of the considered dependent variables).*

Experiment Design. We used the *one factor with two treatments* design [40]. We randomly divided the participants into two groups: Ionic (*i.e.*, control group) and Android (*i.e.*, treatment group). The participants in the first group were asked to accomplish only the experiment tasks on the version of Movies-app implemented by using Ionic-Cordova-Angular technology. The participants in the second group were asked to accomplish the tasks only on the version of such app migrated to Android. The participants in the Ionic group were 20, while those in the Android one were 19.

Procedure. The experimental procedure included the following sequential steps.

1. We invited all the students and asked them to fill in the pre-questionnaire to gather their demographic information.
2. We randomly split the participants into two groups: Ionic and Android.
3. The experiment session took place under controlled conditions in a laboratory at the University of Salerno. All the used PCs had the same (Hardware/Software) configuration.
4. The participant performed the comprehension task by answering the questions of the comprehension questionnaire.
5. We asked the participants to deal with each fault at a time. The participants could pass to the next fault only when they either fixed the previous fault or were aware that they could not identify/fix it.
6. Participants filled in the post-questionnaire by rating affective reactions.
7. Participants compressed and archived their version of the app with the source code they modified. We then collected all those versions.

Analysis Procedure. We carried out the following steps:

- We undertook the descriptive statistics.
- To test the null hypotheses, we planned to use either an unpaired t-test or the Mann-Whitney U test [29]. Unlike the t-test, the Mann-Whitney U test does not require the assumption of normal distributions. To study the normality of data, we use the Shapiro-Wilk W test [36]. Regarding this test, a p-value lower than a fixed α indicates that data are not normally distributed. In the case of a statistically significant effect of Technology, we planned to compute effect size (Cohen’s d or Cliff’s δ) to measure the magnitude of such a difference. We applied a non-parametric statistical analysis (Mann-Whitney U test [29]) when considering affective reaction. If any statistically significant difference is found, we measure its extent through Cliff’s δ .

As we did for the data analysis of the user study, we fixed α to 0.05 to verify if an effect is statistically significant.

3.4 Controlled Experiment Results

In Table 2, we report the descriptive statistics and the results of the statistical tests performed (*i.e.*, p-values).

Android vs. Ionic with Respect to Source-Code Comprehension. As for *Comprehension*, descriptive statistics (Table 2) do not show a huge difference in the source-code comprehension the participants achieved in the Ionic and Android groups. Descriptive statistics indicate that the participants in the Ionic group answered the questions of the comprehension questionnaire better: the mean and median values are 0.625 and 0.667, respectively; while the mean and

Table 2. Descriptive Statistics of Comprehension and Correctness of Fault Identification and Fixing.

Technology	Comprehension				Correctness of fault identification				Correctness of fault fixing			
	Mean	Std. Dev	Median	p-value	Mean	Std. Dev	Median	p-value	Mean	Std. Dev	Median	p-value
Android	0.5	0.266	0.5	0.109	0.882	0.255	1	0.971	0.829	0.289	1	0.935
Ionic	0.625	0.152	0.667		0.9	0.189	1		0.850	0.235	1	

median values for the Android group are both 0.5. The results of the Shapiro-Wilk W test suggest that data were not normally distributed in the Ionic group (p -value = 0.007). For such a reason, we performed the Mann-Whitney U test. The returned p -value is 0.109, *i.e.*, there is no statistically significant difference between the comprehension that the participants in the two groups achieved.

The results of the Mann-Whitney U test do not allow us to reject the null hypothesis, thus we could not observe a statistically significant difference in the comprehensibility of the source code written in either Android or Ionic-Cordova-Angular technologies.

Android vs. Ionic with Respect to Fault Identification. Descriptive statistics (Table 2) suggest that all the participants achieved high correctness in the identification of the faults. The mean values for the *Correctness of Fault Identification* are 0.882 and 0.9 for Android and Ionic, respectively. The results of the Shapiro-Wilk W test show that data did not follow a normal distribution: the p -values are $1.294e-06$ for Android and $1.422e-06$ for Ionic. The results of the Mann-Whitney U test do not indicate any statistically significant difference between the data in the two groups since the p -value is 0.971.

The results of the statistical inference do not allow us to reject the null hypothesis, thus, also in this case, we could not observe a statistically significant difference in the identification of faults in the source code written in either Android or Ionic-Cordova-Angular technologies.

Android vs. Ionic with Respect to Fault Fixing. As we could suppose, for *Correctness of Fault Fixing* we observed a pattern similar to *Correctness of Fault Identification*. The participants in the groups achieved high correctness in the fixing of the faults in both the versions of Movies-app (see Table 2). Data were not normally distributed since the Shapiro-Wilk W test returned $2.04e-05$ and $2.656e-05$ as the p -values for the Android and Ionic groups, respectively. Then, we applied the Mann-Whitney U test and we obtained 0.935 as the p -value.

We did not observe a statistically significant difference in the fixing of faults in the source code written in either Android or Ionic-Cordova-Angular technologies.

Android vs. Ionic with Respect to the Affective Reactions of Developers. As Sullivan and Artino [37] suggest, we used median values and frequencies as descriptive statistics of the dependent variables PLS_K , ARS_K , DOM_K , and LIK_K (where K is the kind of task, *i.e.*, source code comprehension and fault identification and fixing). In Table 3, we report the median values and

Table 3. Median values and statistical test results for the affective reactions of the participants in the study.

Technology	PLS_{Comp}	ARS_{Comp}	DOM_{Comp}	LIK_{Comp}	PLS_{Ident}	ARS_{Ident}	DOM_{Ident}	LIK_{Ident}	PLS_{Fix}	ARS_{Fix}	DOM_{Fix}	LIK_{Fix}
Android	7	7	8	7	8	7	8	8	8	7	8	8
Ionic	7.5	6.5	9	8	8	7.5	9	8	8	7.5	8	8.5
p-value	0.988	0.503	0.106	0.326	0.352	0.626	0.206	0.912	0.538	0.966	0.768	0.59

the p-values of the statistical test performed. As for the Comprehension task, there is not a huge difference between the dependent variables (*i.e.*, PLS_{Comp} , ARS_{Comp} , DOM_{Comp} , and LIK_{Comp}) in the two groups. However, the medians for the Ionic group were always higher than the Android group ones, except for ARS_{Comp} . The Mann-Whitney U test returned p-values higher than 0.05 for all the dependent variables, hence there is no statistically significant difference between the affective reactions of the two groups.

Also for the Fault Identification task, there is not a huge difference between the two groups. Medians of dependent variables of the Ionic group were always greater or equal to the Android group ones. The Mann-Whitney U test returned p-values higher than 0.05 for all the dependent variables signifying that there is no statistically significant difference between the affective reactions of both the groups. The analysis of the data from the Fault Fixing task allowed identifying the same pattern as we identified for the Fault Identification task.

The affective reactions of developers in the two groups do not show a statistically significant difference when dealing with a cross-platform app and its migrated version to comprehend source code and identify and fix faults in that code.

4 Overall Discussion

In this section, we discuss implications and future extensions related to the results of both our studies. We conclude this section by discussing the threats that could have affected the validity of the obtained results.

4.1 Implications and Future Extensions

We delineate a number of practical implications from the researcher and the practitioner perspectives. We also suggest possible future directions for research. We believe that one of the values of our paper concerns the body of knowledge on the relevance of approaches to migrate cross-platform apps toward a native platform and how to use this body of knowledge for future research.

- End-users’ opinions are in favor of the Android (migrated) version of Movies-app in terms of Liking and Efficiency. However, UEQ statements measuring how generally appealing the two apps are (*i.e.*, Attractiveness, Perspicuity, Stimulation, and Novelty) seem not to highlight any preference for either technology. Thus, we conjecture that the Liking dimension might be influenced by the performance of the apps perceived by end-users. This point has implications for the practitioner, who could invest initial efforts in the development

- of apps through a cross-platform technology (*e.g.*, for time-to-market or prototyping reasons) and then she could decide to migrate such apps towards a native technology to have a better UX and let the app affirm in the market.
- Overall results of the controlled experiment suggest that novice developers did not find a huge difference between the two studied technologies (Ionic-Cordova-Angular and Android) in terms of the source-code comprehension and the correctness of fault identification and fixing (although a slight difference in the size of the two versions of the app). Furthermore, the affective reactions of developers seem not to be affected when performing tasks as well as the difficulty perceived to accomplish them (see TR). This outcome might be relevant to the practitioner. In particular, our study seems to support one of the main results by Francese *et al.* [15]; *i.e.*, cross-platform development is valuable when an app has to be run in different hardware/software platforms. That is, native technology should be preferred in all the other cases.
 - Outcomes of the controlled experiment also suggest future research on the design and the implementation of native and cross-platform apps. However, our experiment is based on tasks concerning existing source code, but there could be found a difference in the use of these technologies when conducting implementation tasks. This point is of interest to the researcher.
 - The researcher could be interested to study if the shown outcomes hold also for apps developed by cross-platform technologies different from those considered in our research (*i.e.*, Ionic-Cordova-Angular) when migrating them towards a native platform (*e.g.*, Android as we did, but also iOS or others). An experimental approach similar to that used (user study and a controlled experiment) could be adopted.
 - The adoption of a migration approach should also consider the cost for its application. This aspect, not considered in this paper, is of particular interest for the practitioner. In fact, it could be crucial knowing whether it is less costly to migrate an apps developed by cross-platform technology to a native platform rather than to re-engineer it. Obviously, this aspect is also relevant for the researcher because she could define predictive models to quantify costs and benefits to migrate or re-engineer cross-platform app. Our research contributes to justify further work on this subject.
 - The experiment object is of a specific kind of apps, *i.e.*, the entertainment universe. The researcher and the practitioner could be interested in studying whether our results also hold for different kinds of apps. It could be also of interest for the researcher to study whether our outcomes scale to applications more complex and larger than that studied.
 - The diffusion of a new technology/method is made easier when empirical evaluations are performed and their results show that such a technology/method solves actual issues [31]. The results of our investigations suggest that migrating cross-platform apps towards a native platform matters from the practitioner perspective. This outcome could increase the diffusion of such a kind of migration approach and the definition of new ones. These points are clearly relevant for both the practitioner and the researcher.

4.2 Threats to Validity

Internal Validity. A possible threat to Internal Validity is voluntary participation (*selection threat*). We embedded the experiment in a University course and did not consider experiment scores to grade participants in that course.

To deal with *threat of diffusion or treatment imitations*, we monitored participants and asked back material to prevent them from exchanging information in the controlled experiment. We also prevented the diffusion of the experimental material by gathering it at the end of the tasks from all participants.

Another threat might be *resentful demoralization*—participants assigned to a less desirable treatment might not perform as well as they normally would. This kind of task is present only in the controlled experiment.

Construct Validity. A possible threat is concerned to the *threat of hypotheses guessing* could be present. Although the participants were not informed about our goals, they might guess them and change their behavior accordingly. To deal with this kind of threat we did not disclose the goals to the participants.

To mitigate *evaluation apprehension threat*, we reassured participants in the controlled experiments that their data were treated anonymously. We also asked the participants to sign a consent form to use their data.

Conclusion Validity. We involved participants with a different background in the user study and then the *threat of random heterogeneity of participants* could be present. In the controlled experiment, participants followed the same course, underwent the same training, and had a similar background.

Reliability of measures is another threat to conclusion validity. We used well-known and widely used measures in both studies.

Low statistical power refers to the ability of the test to reveal a true pattern in the data. If the power is low, there is a risk that an erroneous conclusion is drawn from data. To partially deal with this kind of threat we used robust and sensitive statistical tests [24].

External Validity. The participants in the controlled experiment were graduate students. This could pose some threats to the generalizability of the results to the population of professional developers (*threat of interaction of selection and treatment*). The studied technology is relatively novel and then we can speculate that participants are more experienced than many professionals.

The experimental object might affect external validity (*threat of the interaction of setting and treatment*). In particular, Movies-app could be not representative of the universe of real-world apps.

5 Conclusion

Results suggest that migrating cross-platform apps developed by Ionic-Cordova-Angular towards Android matters from both the developers' and end-users' perspectives. This main outcome allows formulating the most important practical takeaway message from our research: it is worthy to develop an app by using a

cross-platform technology (*e.g.*, for time-to-market reasons) and then to assess if this app is ready for the market; if this happens its migration to a native technology is a good option to provide better support to the UX so letting the app penetrate more the app market.

References

1. Movies-App. <https://github.com/okode/movies-app>
2. The Stackoverflow comment. <https://stackoverflow.com/questions/34986098/migrating-from-hybrid-app-to-native-app-at-later-point-of-time>
3. Angulo, E., Ferre, X.: A case study on cross-platform development frameworks for mobile applications and UX. In: Proceedings of HCI (2014)
4. Bjørn-Hansen, A., Rieger, C., Grønli, T.-M., Majchrzak, T.A., Ghinea, G.: An empirical investigation of performance overhead in cross-platform mobile development frameworks. *Empir. Softw. Eng.* **25**(4), 2997–3040 (2020). <https://doi.org/10.1007/s10664-020-09827-6>
5. Bradley, M.M., Lang, P.J.: Measuring emotion: the self-assessment manikin and the semantic differential. *J. Behav. Ther. Exp. Psychiatry* **25**(1), 49–59 (1994)
6. Brodie, M.L., Stonebraker, M.: Legacy Information Systems Migration: Gateways, Interfaces, and the Incremental Approach (1995)
7. Brunner, E., Dette, H., Munk, A.: Box-type approximations in nonparametric factorial designs. *J. Am. Stat. Assoc.* **92**(440), 1494–1502 (1997)
8. Caulo, M., Francese, R., Scanniello, G., Spera, A.: Dealing with comprehension and bugs in native and cross-platform apps: a controlled experiment. In: Franch, X., Männistö, T., Martínez-Fernández, S. (eds.) PROFES 2019. LNCS, vol. 11915, pp. 677–693. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-35333-9_53
9. Caulo, M., Francese, R., Scanniello, G., Spera, A.: Does the migration of cross-platform apps towards the android platform matter? An approach and a user study. In: Franch, X., Männistö, T., Martínez-Fernández, S. (eds.) PROFES 2019. LNCS, vol. 11915, pp. 120–136. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-35333-9_9
10. Cliff, N.: Ordinal methods for behavioral data analysis (1996). <https://books.google.it/books?id=bIJFvgAACAAJ>
11. Corral, L., Sillitti, A., Succi, G.: Mobile multiplatform development: an experiment for performance analysis. *Proc. Comput. Sci.* **10**, 736–743 (2012)
12. De Lucia, A., Francese, R., Scanniello, G., Tortora, G.: Developing legacy system migration methods and tools for technology transfer. *Softw. Pract. Exp.* **38**, 1333–1364 (2008)
13. El-Kassas, W.S., Abdullah, B.A., Yousef, A.H., Wahba, A.: ICPMD: integrated cross-platform mobile development solution. In: International Conference on Computer Engineering and Systems, pp. 307–317 (2014)
14. El-Kassas, W.S., Abdullah, B.A., Yousef, A.H., Wahba, A.M.: Taxonomy of cross-platform mobile applications development approaches. *Ain Shams Eng. J.* **8**(2), 163–190 (2017)
15. Francese, R., Gravino, C., Risi, M., Scanniello, G., Tortora, G.: Mobile app development and management: results from a qualitative investigation. In: Proceedings of International Conference on Mobile Software Engineering and Systems, pp. 133–143 (2017)

16. Heitkötter, H., Hanschke, S., Majchrzak, T.A.: Evaluating cross-platform development approaches for mobile applications. In: Cordeiro, J., Krempels, K.-H. (eds.) WEBIST 2012. LNBIP, vol. 140, pp. 120–138. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36608-6_8
17. Heitkötter, H., Kuchen, H., Majchrzak, T.A.: Extending a model-driven cross-platform development approach for business apps. *Sci. Comput. Program.* **97**, 31–36 (2015)
18. Ergonomics of human system interaction - Part 210: Human-centered design for interactive systems. Standard, International Organization for Standardization (2009)
19. Jedlitschka, A., Ciolkowski, M., Pfahl, D.: Reporting experiments in software engineering. In: Shull, F., Singer, J., Sjøberg, D.I.K. (eds.) *Guide to Advanced Empirical Software Engineering*, pp. 201–228. Springer, London (2008). https://doi.org/10.1007/978-1-84800-044-5_8
20. Juristo, N., Moreno, A.: *Basics of Software Engineering Experimentation*. Springer, Heidelberg (2001). <https://doi.org/10.1007/978-1-4757-3304-4>
21. Kamsties, E., von Knethen, A., Reussner, R.: A controlled experiment to evaluate how styles affect the understandability of requirements specifications. *Inf. Soft. Technol.* **45**(14), 955–965 (2003)
22. Kaptein, M., Nass, C., Markopoulos, P.: Powerful and consistent analysis of likert-type ratingscales, vol. 4, pp. 2391–2394 (2010)
23. Kim, S., Clark, J.A., McDermid, J.A.: *The rigorous generation of java mutation operators using hazop technical report* (1999)
24. Kitchenham, B., et al.: Robust statistical methods for empirical software engineering. *Empir. Softw. Eng.* **22**(2), 579–630 (2016)
25. Koelstra, S., et al.: DEAP: a database for emotion analysis using physiological signals. *IEEE Trans. Affect. Comput.* **3**(1), 18–31 (2012)
26. Latif, M., Lakhri, Y., Nfaoui, E.H., Es-Sbai, N.: Cross platform approach for mobile application development: A survey. In: *Proceedings of International Conference on Information Technology for Organizations Development*, pp. 1–5 (2016)
27. Laugwitz, B., Held, T., Schrepp, M.: Construction and evaluation of a user experience questionnaire. In: Holzinger, A. (ed.) *USAB 2008. LNCS*, vol. 5298, pp. 63–76. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89350-9_6
28. Malavolta, I., Ruberto, S., Soru, T., Terragni, V.: End users' perception of hybrid mobile apps in the google play store. In: *Proceedings of International Conference on Mobile Services*, pp. 25–32 (2015)
29. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Statist.* **18**(1), 50–60 (1947)
30. Noei, E., Syer, M.D., Zou, Y., Hassan, A.E., Keivanloo, I.: A study of the relation of mobile device attributes with the user-perceived quality of android apps. In: *Proceedings International Conference on Software Analysis, Evolution and Reengineering*, p. 469 (2018)
31. Pflieger, S.L., Menezes, W.: Marketing technology to software practitioners. *IEEE Softw.* **17**(1), 27–33 (2000)
32. Que, P., Guo, X., Zhu, M.: A comprehensive comparison between hybrid and native app paradigms. In: *Proceedings of International Conference on Computational Intelligence and Communication Networks*, pp. 611–614 (2016)
33. Rieger, C., Majchrzak, T.A.: Towards the definitive evaluation framework for cross-platform app development approaches. *J. Syst. Softw.* **153**, 175–199 (2019)

34. Romano, J., Kromrey, J.: Appropriate statistics for ordinal level data: should we really be using t-test and Cohen's d for evaluating group differences on the NSSE and other surveys? (2006)
35. Scanniello, G., Risi, M., Tramontana, P., Romano, S.: Fixing faults in C and java source code: abbreviated vs. full-word identifier names. *ACM Trans. Softw. Eng. Methodol.* **26**(2), 6:1–6:43 (2017)
36. Shapiro, S., Wilk, M.: An analysis of variance test for normality. *Biometrika* **52**(3–4), 591–611 (1965)
37. Sullivan, G., Artino, A.: Analyzing and interpreting data from Likert-type scales. *J. Grad. Med. Educ.* **5**, 541–2 (2013)
38. Vegas, S., Apa, C., Juristo, N.: Crossover designs in software engineering experiments: benefits and perils. *IEEE Trans. Softw. Eng.* **42**(2), 120–135 (2016)
39. Watson, D., Clark, L.A., Tellegen, A.: Development and validation of brief measures of positive and negative affect: the PANAS scales. *J. Pers. Soc. Psychol.* **54**(6), 1063 (1988)
40. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B., Wesslén, A.: *Experimentation in Software Engineering*. Springer, Heidelberg (2012). <https://doi.org/10.1007/978-3-642-29044-2>