Self-Adaptation in Mobile Apps: a Systematic Literature Study

Eoin Martino Grua, Ivano Malavolta, Patricia Lago
Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
Email: {e.m.grua | i.malavolta | p.lago}@vu.nl

***Abstract*—With their increase, smartphones have become more integral components of our lives but due to their mobile nature it is not possible to develop a mobile application the same way another software system would be built. In order to always provide the full service, a mobile application needs to be able to detect and deal with changes of context it may be presented with. A suitable method to achieve this goal is self-adaptation. However, as of today it is difficult to have a clear view of existing research on self-adaptation in the context of mobile applications.**

In this paper, we apply the systematic literature review methodology on selected peer-reviewed papers focusing on self-adaptability in the context of mobile applications. Out of 607 potentially relevant studies, we select 44 primary studies via carefully-defined exclusion and inclusion criteria. We use known modelling dimensions for self-adaptive software systems as our classification framework, which we apply to all selected primary studies. From the synthesized data we obtained, we produce an overview of the state of the art. The results of this study give a solid foundation to plan for future research and practice on engineering self-adaptive mobile applications.

I. Introduction

Since the announcement of the iPhone in 2007 and the sale of Android based smartphones, the number of mobile applications has been increasing and so is the number of mobile users [20], [39].

With their increase, smartphones have become more integral components of our lives but do to their mobile nature it is not possible to develop a mobile application the same way another software system would be built. In order to always provide the full service, a mobile application needs to be able to detect and deal with changes of context it may be presented with. A suitable method to achieve this goal is self-adaptation [36].

While there is a lot of work that has been done in the field of self-adaptation [47], to the best of our knowledge there is no published literature review that explores self-adaptation in the specific context of mobile applications. Within this context we can identify several questions related to the most common goals that the self-adaptive systems are aiming to achieve, what kind of changes can trigger adaptation processes, how is it achieved in current published work and what would be the outcomes and effects of the adaptation to the mobile application. Unveiling the above mentioned aspects will give a better understanding of the current landscape of self-adaptation for mobile applications.

In this study we aim to fill the knowledge gap present with self-adaptive systems in the context of mobile applications. To do so, we apply the systematic literature review methodology [17] and target peer-reviewed papers focusing on self-adaptability in the context of mobile applications. Out of 607 potentially relevant studies, we select 44 primary studies via carefully-defined selection criteria. We then utilize and customize known modelling dimensions for self-adaptive software systems [2] and use them as our classification framework, which we apply to all selected primary studies.

Obtained **results** reveal that the most common sources of change are hardware (which includes the battery of the device) and the internet connectivity. Most analyzed approaches perform the self-adaptation in an autonomous manner and adaptation happens within the application itself, with sometimes the use of the backend (*e.g.,* cloud offloading systems). Furthermore, in all primary studies adaptation is event triggered and performed in a best-effort manner, without a strict guarantee on the duration of the self-adaptation process. Most of the approaches are not specific to any application domain, with a lack of case study evaluation.

The **main contributions** of this study are:

- an up-to-date *systematic review* of the literature on self-adaptation in the context of mobile applications;
- a customized *classification framework* for understanding, classifying, and comparing approaches for self-adaptation in the context of mobile apps;
- a *discussion* of the main implications of this study, the application domains covered by the literature so far, and future research challenges;
- a *replication package* including the research protocol, raw data, and analysis scripts for independent replication and verification of this study.

The **target audience** of this paper includes: researchers working in the field of self-adaptation and want to have better insight of the literature when specifically dealing with mobile applications, researchers and mobile application developers looking to implement self-adaptation in their system but do not have prior experience in the field and need a guide to understand what has been done so far.

The rest of the paper is organized as follows. In Section II we give background information on self-adaptation in mobile applications. Section III explains the study design, whereas its results are reported in Section IV. In section V we provide a discussion of the emerging results, followed by section VI in which we present threats to validity. Section VII presents related work and lastly we close the paper in section VIII.

II. Self-Adaptation in Mobile Applications

Figure 1 shows an overview of a **mobile-enabled system**. We use the entities shown in the figure to settle with the terminology used throughout the paper. Mobile apps consist of binary executable files that are downloaded directly into the user's device and stored locally [25]. Mobile apps are developed directly atop the services provided by their underlying mobile platform [26]. Platform services are exposed via a dedicated Application Programming Interface (API) and provide functionalities related to communication and messaging, graphics, location, security, etc. [9]. Moreover, the platform API abstracts and provides access to the hardware components of the device such as its proximity sensor, GPS, accelerometer, battery, networking devices, and so on. Apps can also communicate with other apps installed on the device via a dedicated event-based communication system (*e.g.,* Android intents and broadcast receivers[1]). Smart objects such as fitness trackers, smart headphones, external sensors, and smartwatches can be connected to the mobile device either via short-range communication protocols (*e.g.,* bluetooth) or by passing through the Internet.

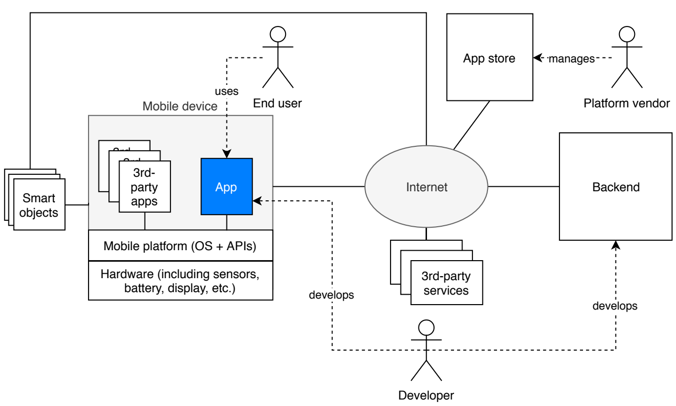


Fig. 1: Overview of a mobile-enabled system

The vast majority of mobile apps send and receive data to their remote backends in order to persist data across usage sessions, share data across apps instances, etc. The communication between the app and its backend is usually performed in a RESTful fashion via the HTTP protocol [1], [22]. Similarly, apps can also communicate with 3rd-party services, for example for authentication via Facebook, accessing the mapping services of Google Maps, sharing data to social networks. Mobile apps are distributed via dedicated app stores, such as the Google Play Store for Android apps and the Apple app store for iOS apps. App stores are managed by platform vendors like Google and Apple [9].

Self-adaptation can happen in any part of a mobile-based system (*e.g.,* in the app itself, in the backend, in a smart object) and can be applied to different levels of the technology stack of computing systems (*e.g.,*, at the hardware level, at the platform level, in the business logic of the app). In the context of this study, a self-adaptive system is defined as a system that can autonomously handle changes and uncertainties in its environment, the system itself and its goals [41]. A self-adaptive system is internally composed of two parts: one has the responsibility of performing the business capabilities of the system (*i.e.,* the operations for which the system is built), whereas the second part interacts with the first one and is responsible for the adaptation process [41]. As an example of a self-adaptive mobile application we take the one described by Moghaddam et al. [28]. The work describes a framework built to enhance energy efficiency in mobile apps. The framework was built with the MAPE model functionalities and consists of a scheduler that is in charge of allocating resources in real-time. In this particular implementation case the authors focus on the network scheduling strategies.

[1] https://developer.android.com/guide/components/broadcasts

III. Study Design

In this section we present the design of this study. We firstly present the research questions (Section III-A) and then we explain our search and selection process in Section III-B. We report on the data extraction process and the framework used to classify the information extracted from our primary studies in Section III-C. Lastly, we explain how we synthesized the main findings from the extracted data in Section III-D.

The *replication package* is publicly available to researchers interested in replicating and independently verifying the study [13]. The replication package includes the raw data of the search and selection phases of the study, the raw data extracted from each primary study, and the full list of all primary studies.

A. Goal and Research Questions

Below we show the formalization of the goal of this study according to the Goal-Question-Metric approach [4].

Purpose	Identify, classify, and evaluate
Issue	the characteristics
Object	of existing approaches for self-adaptation in mobile apps
Viewpoint	from the researcher's and practitioner's point of view.

By building on the modeling dimensions for self-adaptive software systems proposed by Andersson *et al.* [2], we can elicit the following research questions targeted by our study.

RQ1 *What are the goals of self-adaptation in the context of mobile apps?*
Answering this question we aim to identify the characteristics of the goals that self-adaptation should achieve in the context of mobile apps. As an example, the the self-adaptation mechanism proposed in the primary study by Moghaddam *et al.* [28] has the main goal of *reducing the energy consumption* of the application in order to prolong the smartphone's battery life.

RQ2 *What are the changes triggering the self-adaptation in the context of mobile apps?*
By answering this question we want to gain insight on the characteristics of the changes triggering self-adaptation in mobile apps. For example, referring back

to the previously mentioned study by Moghaddam *et al.*, one of the possible sources of a change is the event in which a new application requests to transfer data since it requires adaptation on resource scheduling within the whole mobile device.

RQ3 *What are the mechanisms used for self-adaptation in the context of mobile apps?*
Answering this question will allow us to better understand the characteristics of the mechanisms for self-adaptation within the context of mobile applications. For example, in the case of Moghaddam *et al.*, the mechanism for adaptation is *structural* since adaptation involves the reconfiguration of the overall architecture of the whole system.

RQ4 *What are the effects of self-adaptation in the context of mobile apps?*
By answering this question we will gain better understanding of what are the effects of self-adaptation upon mobile-enabled systems. In this context, a dimension for judging the effect of self-adaptation is by understanding its criticality, *i.e.,* the impact that the self-adaptation process would have on the mobile application in case said adaptation fails. For example, returning to the primary study by Moghaddam *et al.* the criticality of the self-adaptation process is *harmless* since the mobile app is able to function even if the adaptation fails (the only downfall would be the continuation of the current use of energy, instead of reducing it).

The research questions shape the whole study, with a special influence on (i) search and selection of primary studies, (ii) data extraction, and (iii) data synthesis.

B. Search and Selection

As shown in Figure 2, the search and selection process of this study has been designed as a multi-stage process, so to have full control over the studies being considered during the various stages.

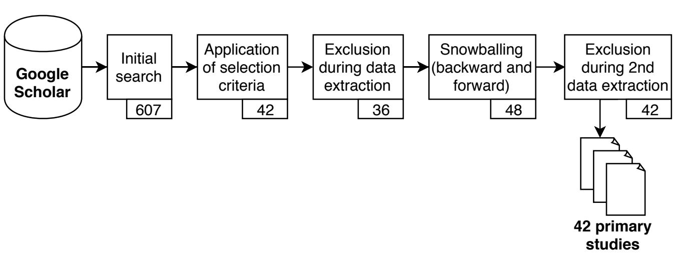


Fig. 2: The search and selection process of this study

Initial search. In this stage we perform an automated search on *Google Scholar*, which at the time of writing is one of the largest and most complete databases and indexing systems for scientific literature. We use such a data source for the following main reasons: (i) it provides the highest number of potentially relevant studies compared to other four relevant libraries (Scopus, ACM Digital Library, IEEE Explore, and Web of Science), (ii) as reported in [44], the adoption of this indexer has proved to be a sound choice to identify the initial set of literature studies for the snowballing process, (iii) the query results can be automatically extracted from the indexer. The query we use to perform the initial search is provided in Listing 1

```
( adaptive OR "self-adaptation" OR "self-adaptive")
AND ( android OR ios OR mobile )
AND ( apps OR applications OR application ))
```

Listing 1: Search string used for the automatic search

In order to cover as much potentially relevant studies as possible, we kept our search string as generic as possible and considered exclusively the object of our research. Indeed, the search string can be divided into three main components, one for each line of the listing, where the first component captures self-adaptive systems, the second captures the mobile nature of the targeted approaches, and the third one is about apps and applications. The search string has been tested by executing pilot searches on Google Scholar. In order to keep the results of this initial search as focused as possible, the query has been applied to the title of the targeted studies. The considered timeframe ranges from 2007[2] and ends at the time in which the query has been executed (*i.e.,* November 2018).

Application of selection criteria. In this stage we consider all the 607 studies resulting from the initial search and filtered them according to a set of well-defined inclusion and exclusion criteria. In this stage it is crucial to select studies objectively and in a cost-effective manner, so we apply the adaptive reading technique [34], as the full-text reading of clearly excluded studies is not necessary. In the following we report the inclusion and exclusion criteria of this study.

Inclusion criteria:
1) The study focuses on self-adaptability, as defined in [2].
2) The study focusses on mobile applications, as defined in Section II.

Exclusion criteria:
1) Secondary or tertiary studies (*e.g.,* systematic literature reviews, surveys, etc.).
2) Studies in the form of editorials, tutorial, poster papers, because they do not provide enough information.
3) Studies that have not been published in English language.
4) Duplicate papers or extensions of already included papers.
5) Studies that have not been peer reviewed.
6) Papers that are not available, as we cannot inspect them.

Each paper is included as primary study if it satisfies *all* inclusion criteria, and it is discarded if it meets *any* exclusion criterion. The definition of the above mentioned criteria has been incrementally refined and tested by two researchers by considering a set of pilot studies. It is important to note that we excluded secondary studies because of the first exclusion criterion, but we discuss them in our related work section (see Section VII).

[2]The first announcement about the existence of mobile apps as defined in Section II has been done in the well-known keynote where Steve jobs firstly launched the iPhone in 2007 [3].

Attribute	*Possible values*	*Definition*
Goals - goals are objectives the system under consideration should achieve		
Quality requirement	functional suitability (FUN), performance efficiency (PERF), compatibility (COMP), usability (US), reliability (REL), security (SEC), maintainability (MAINT), portability (PORT), energy (EN), any or all of the possibilities (?)	The system/software quality the goal is aiming to achieve
Evolution	static (S), dynamic (D)	Whether the goals can change within the lifetime of the system
Flexibility	rigid (R), constrained (C), unconstrained (D)	Whether the goals are flexible in the way they are expressed
Duration	temporary (T), persistent (P)	Validity of a goal thoroughout the system lifetime
Multiplicity	single (S), multiple (M)	How many goals are there?
Change - Change is the cause of the adaptation		
Source	APP, 3rd party app (3A), mobile platform (PLAT), hardware (HW), smart objects (SMARTO), end user (USER), backend (BACK), 3rd party services (3S), developer (DEV), app store (STORE), platform vendor (VENDOR), INTERNET	Where is the source of the change?
Frequency	rare (R), frequent (F)	How often a particular change occurs?
Anticipation	foreseen (FN), foreseeable (FE), unforeseen (UN)	Whether the change can be predicted
Mechanisms - what is the reaction of the system towards change		
Type	parametric (P), structural (S)	whether adaptation is related to the parameters of The system components or to the structure of the system
Autonomy	autonomous (AU), human assisted (H)	What is the degree of the outside intervention during adaptation
Organization	centralized (C), decentralized (D)	whether the adaptation is done by a single Component or distributed amongst several components
Scope	APP, 3rd party app (3A), mobile platform (PLAT), hardware (HW), smart objects (SMARTO), end user (USER), backend (BACK), 3rd party services (3S), developer (DEV), app store (STORE), platform vendor (VENDOR), INTERNET	Where in the system is the adaptation localized
Duration	short (S), medium (M), long (L)	How long the adaptation lasts
Timeliness	best effort (B), guaranteed (G)	Whether the time period for performing self-adaptation can be guaranteed
Triggering	event-triggered (E), time-trigger (T)	Whether the change that triggers adaptation is associated with an event or a time slot
Effects - What is the impact of adaptation upon the system		
Criticality	harmless (H), mission-critical (M), safety-critical (S)	Impact upon the system in case the self-adaptation fails
Predictability	non-deterministic (N), deterministic (D)	Whether the consequences of the adaptation can be predictable
Overhead	insignificant (I), reasonable (R), failure (F)	The impact of system adaptation upon the quality of services of the system
Resilience	resilient (R), semi-resilient (S), vulnerable (V)	The persistence of service delivery that can justifiably be trusted, when facing changes

TABLE I: Classification framework utilized for the data extraction

Exclusion during data extraction. When going through each primary study in detail for extracting information, we agreed that 6 analysed studies were semantically out of the scope of this research and we excluded them, leading to a set of 36 potentially relevant studies.

Snowballing. To reduce potential bias introduced with the use of our selected search string we also carry out a snowballing process[12]. The main goal of this stage is to enlarge the set of potentially relevant studies by considering each study selected in the previous stages, and focusing on those papers either citing and cited by it. More technically, we perform a closed recursive backward and forward snowballing activity [45]. In both backward and forward snowballing the initial screening of additional studies is based on their title only, whereas the final decision about their inclusion into the set of primary studies is based on their full text and on the selection criteria discussed above. Duplicates have been removed at each iteration of the snowballing activity.

Exclusion during 2nd data extraction. In this phase we extract data from the 12 additional papers resulting from the snowballing activity and agree that 6 of them are semantically out of the scope of this research and we exclude them. This final check leads to the final set of 42 primary studies, which are then analyzed in details for answering our research questions.

C. Data Extraction

In this section we present how we perform the data extraction on the selected primary studies. The main goal of this phase is to collect data from each primary study, so to be able to suitably compare them in the subsequent data synthesis phase. The data extraction phase is executed collaboratively by two of the authors of this study. In order to have a rigorous data extraction process and to ease the management of the extracted data, a well-structured classification framework has been designed upfront.

As anticipated, we build our classification framework on the modeling dimensions for self-adaptive software systems presented by Andersson *et al.* [2]. In order to better fit the framework to the characteristics of mobile applications, we *customize* the modeling dimensions for self-adaptive systems presented in [2]. The customization of the classification framework has been performed as follows: (i) firstly we selected a subset of 10 pilot studies from the 42 primary studies, (ii) then two researchers independently extracted the data from the 10 pilot studies by using the original version of the self-adaptation modeling dimensions proposed in [2], (iii) the two researchers then discussed the results of the data extraction, with a special focus on too generic/abstract attributes, those attributes which did not fully fit with the characteristics of the primary studies, attributes whose values were redundant, (iv) based on the discussion, the self-adaptive modeling dimensions have been customized into the final version of the classification framework, and lastly (iv) the final version of the classification framework has been applied to all 42 primary studies.

The customized classification framework is presented in Table I. Specifically, as part of the Goals dimension, we add an attribute called *quality requirement* so to keep track of the system/software quality requirement that self-adaptation aims to achieve [14] (*e.g.,* security, usability, functional suitability, etc.). Here we are not considering how the (potentially multiple) goals of self-adaptation are related to each other, *i.e.,* the *dependency* attribute, as this attribute resulted to be too fine grained for the objective of our study. For the Changes dimension, we extend the attribute *source* so to directly map it to the main elements of mobile-enabled systems as they are depicted in Figure 1. Also, we are not considering the attribute *type*, which originally was distinguishing between functional or non-functional changes, since we notice that in our primary studies it was strictly contained by the *quality requirement* attribute. For the Mechanisms dimension, the only change we made is related to the extension of the attribute *scope*, which we are also mapping to the main elements of Figure 1. Lastly, we reuse the Effects dimension just as defined in the original framework presented by Andersson *et al.*

D. Data Synthesis

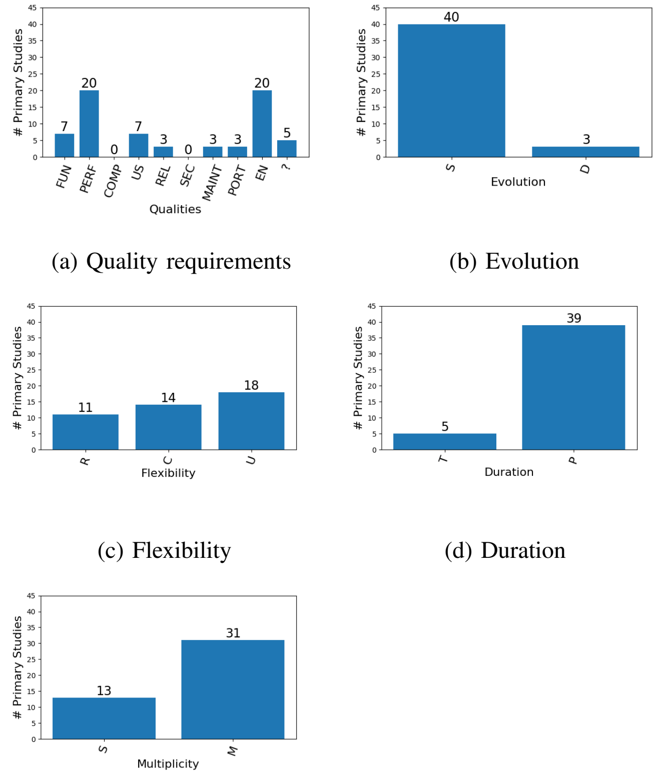
The data synthesis activity involves collating and summarizing the data extracted from the primary studies [17] with the main goal of understanding, analyzing, and classifying current research on self-adaptation in the context of mobile applications. Specifically, we performed a combination of *content analysis* (for categorizing and coding the studies under broad thematic categories) and *narrative synthesis* (for explaining in details and interpreting the findings coming from the content analysis). This phase is performed by all of the authors of this study.

IV. Results

In this section we report the results in the context of all the research questions of our study. On a technical note, some of the plots (*e.g.,* subfigure 3a) contain the '?' bin. The meaning of the symbol is that the examined primary studies in that bin are configurable by developers/users so to fit a variable number of the bins of that category, hence we have classified it as a separate category.

A. Goals of Self-Adaptation (RQ1)

Figure 3 shows the distributions of the goals characteristics across the primary studies. As shown in Figure 3a, the most common **quality** attributes are *performance efficiency* and



(a) Quality requirements
(b) Evolution
(c) Flexibility
(d) Duration
(e) Multiplicity

Fig. 3: Characteristics of the goals of self-adaptation

energy. This is an expected result since both performance and energy are fundamental aspects of the user experience in mobile applications, potentially impacting the app user ratings and reviews which, unless properly addressed, can negatively contribute to the app's success [33], [15]. Furthermore, it is interesting to note that self-adaptation for either *compatibility* or *security* is never mentioned in our primary studies, thus unveiling two potentially fruitful research gaps to be filled by researchers in the future.

For what concerns **evolution** (see Figure 3b), we observe that the vast majority of primary studies presents a system with *static*ally-defined goals, whereas only 3 primary studies present an approach in which goals can evolve during the execution of the system. For example, in MAsCOT [32] developers can configure at any time the self-adaptation objectives and trade-offs (*e.g.,* acceptable latency vs available CPU power) via an XML-based dynamic decision network, which is then used at run-time by the system for deciding whether computation should be executed on the mobile device or in the cloud.

The **flexibility** dimension is quite fragmented (see Figure 3c), where we can see that primary studies are pretty evenly distributed among *rigid*, *constrained*, and *unconstrained* goals, with a slight tendency towards *unconstrained* goals.

If we examine the **duration** attribute in Figure 3d, we observe that most of the primary studies support goals with a *persistent* validity, as opposed to only 5 studies supporting *temporary* goals. The approach presented in [8] is an example of study dealing with temporary goals; the purpose of the approach is to allow developers to develop mobile applications in a declarative manner; then it will be the responsibility of the system to adapt the application to the device on which it is deployed. Since the adaptation goal is limited to the deployment phase, we can consider it as temporary.

Lastly, if we focus on goals **multiplicity**, in figure 3e we can observe that the majority of primary studies have *multiple* goals. This result has to do with most systems employing self-adaptation not only to optimize the system in terms of a single dimension (*e.g.,* to reduce battery consumption), but they focus on the trade-off among different dimensions and types of resources, such as Internet connectivity, CPU usage, user experience, etc.

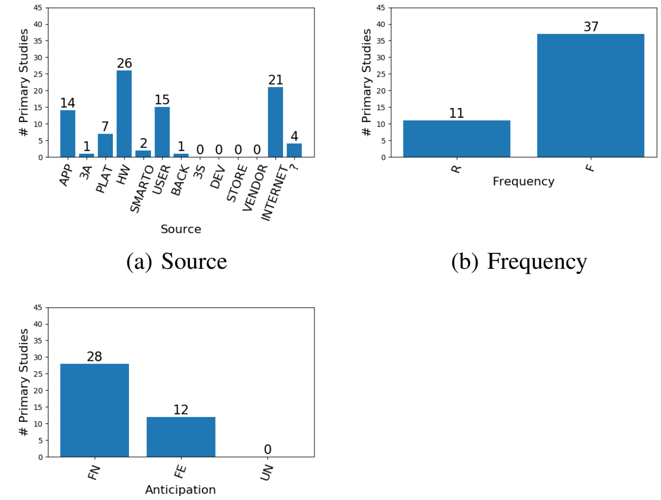
B. Changes Triggering Self-Adaptation (RQ2)

In Figure 4a we can notice that the most common **source** of changes is the *hardware* of the device, with close second being *Internet* connectivity. This result can be due to the vary nature of mobile applications being deployed on smartphones; as such, a common concern for developers is to optimize the utilization of the hardware the application is installed on and to provide the best service at all times (*e.g.,* to do not consume too much battery at runtime or to react to sensors faults). Furthermore, we can explain the fact that Internet is the second most common source of changes in the primary studies because of a significant number of studies dealing with cloud offloading. In those cases, the system decides to offload computation to the cloud depending on the available bandwidth (among other parameters) and adapts its behaviour accordingly. As a last remark on this attribute, we can observe that none of the primary studies consider as a source of change the following entities of mobile-enabled systems: third-party services, developers, app store, and platform vendor. Among them, it comes as a surprise that no primary study considers third-party services as a source of changes. Indeed, it is quite common that third-party services change their provided APIs (*e.g.,* Facebook changing the signature of its GraphAPI[3] endpoint for sharing a link) and it may be interesting to investigate on how self-adaptation techniques can help in automatically keeping the calling apps as reliable as possible, despite those (potentially unforeseen) changes.

The second examined attribute is **frequency**, where we observe that the most common type of changes are *frequent* (see Figure 4a). This result was expected, especially after having observed that the two most common sources of changes are hardware and Internet availability and that by their own nature their status can drastically change in a matter of nanoseconds.

For what concerns the **anticipation** of changes, as shown in Figure 4c, changes are mostly *foreseen*, followed by fewer

[3]https://developers.facebook.com/docs/graph-api



(a) Source

(b) Frequency

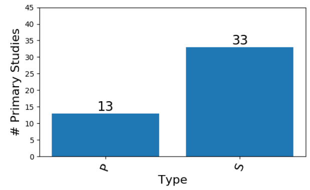
(c) Anticipation

Fig. 4: Characteristics of the changes triggering the self-adaptation process

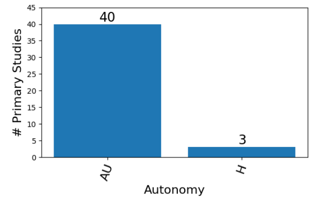
occurrences of approaches supporting *foreseeable* changes. It is important to note that in self-adaptive systems *foreseen* changes are known at design time and considered as expected to occur during the normal operation of the system [19]; examples of foreseeable changes we encountered in the primary studies include: a drop of available bandwidth [10], the user reaching its home address [27], the mobile device getting in proximity of a smart object [7], the user starting to drive a car [38]. Differently, changes are *foreseeable* when they are not known at design time, but they can be resolved at runtime and there is a plan for managing them during the execution of the system [19]. Examples of foreseeable changes include: the backend of the app has a failure [5], the GPS sensors of the mobile device produces incorrect data [8], etc. Finally, it is interesting to note that no primary study considers *unforeseen* changes, *i.e.,* drastic changes that have been not planned for and that are unknown until their first occurrence [19]. Unforeseen changes are extremely challenging to be managed due to their intrinsic level of uncertainty, both at design time and runtime. We speculate that investigating on how to incorporate them into self-adaptive mobile-enabled systems will be a scientifically challenging research area of the future.

C. RQ3 – Mechanisms for Self-Adaptation (RQ3)

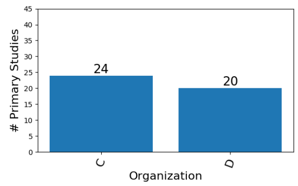
The **type** of self-adaptation supported in more than half of the primary studies is *structural* (see Figure 5a), *i.e.,* the adaptation involves structural changes in software architecture of the system [2]. This result can be explained by the high number of primary studies focussing on communication middleware, generic frameworks and meta-approaches which allow the self-adaptation process to reconfigure the architecture of the system at run-time and are, therefore,



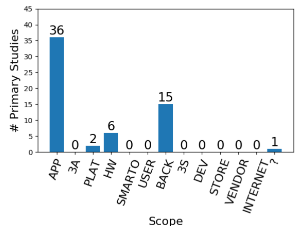
(a) Type



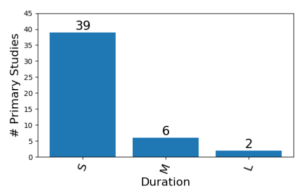
(b) Autonomy



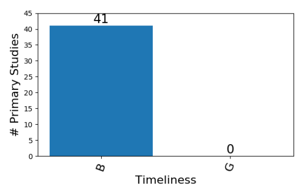
(c) Organization



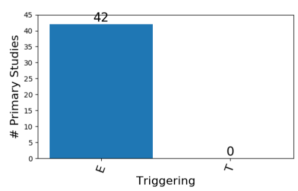
(d) Scope



(e) Duration



(f) Timeliness



(g) Triggering

Fig. 5: Characteristics of the mechanisms for self-adaptation

structural in nature. In 13 primary studies the self-adaptation is *parametric*, *i.e.,* the adaptation process involves only the policy files and configuration of specific components of the system, without changing its overall organization [2]. Examples of primary studies supporting structural self-adaptation include approaches for autonomously adapting the bitrate of the video streaming to the app [29], approaches for sending personalized notifications to the user which automatically adapt to his/her stress levels [21], etc.

For what concerns **autonomy**, we can observe that nearly all approaches are *autonomous* (see Figure 5b). This result is most likely influenced by the definition of self-adaptation that we have used throughout this study, in which the system has to be able to self-adapt and therefore can only have minimal human assistance in the process. Nonetheless, we have found 3 primary studies presenting *human assisted* approaches. For example, the approach presented in [37] aims at improving the user experience of the app with the use of an adaptive user interface. This approach is human aided because it needs the user to participate in a brief test in order to determine where the elements of the user interface should be positioned and their dimensions in the graphical layout of the app. This is obviously in contrast with a fully automated system that would monitor user actions in background and adapt accordingly, without requiring the initial test performed explicitly by the user.

The **organization** attribute is almost evenly split among our primary studies (see Figure 5c). This implies that we have nearly just as many primary studies where the adaptation is *centralized* into a single component as systems where adaptation is *distributed* among several components. An example of centralized self-adaptation is [40], where there is a single software component which is in charge of adapting the user interface of the app to the preferences, knowledge, and skills of the user. An example of distributed self-adaptation is presented in [10], where both the app and its backend are involved in the code offloading process.

Considering our mobile-enabled system in Figure 1 and examining the **scope** of self-adaptation, we notice that the vast majority of the self-adaptation mechanisms are executed in the application itself (see Figure 5d). Quite a few approaches also have mechanisms executing in the back-end of the app, where a significant part is due to the fact that the primary studies deal with cloud offloading (such as the previously given example). Furthermore, we have observed that some studies have self-adaptation mechanisms executing in the hardware and the platform of the mobile device. This result is a confirmation that in a mobile-enabled system the intelligent entities are either the app, its backend, or the software stack on which the app is running on the client side.

Regarding the **duration** of the mechanism (see Figure 5e), we notice that the vast majority of the mechanisms have a *short* duration, followed by 6 approaches having a *medium* duration, and only 2 approaches having a long duration. As an example of a long duration mechanism, the authors of [11] monitor user actions and apply a set of algorithms to perform self-adaptation for adapting the items in the graphical menus of the app according to the usage patterns exhibited by the user. In this way, the app is enhanced with a transformable and movable menu component with adaptable and adaptive features, which improves the overall efficiency of the user when using the app. Due to the fact that the app is constantly learning the usage patterns of the user, the mechanism presented in [11] can be considered as having a long adaptation duration.

As shown in Figure 5f, the **timeliness** attribute falls fully in the *best-effort* bin (*i.e.,* the time for executing the adaptation is not guaranteed) and there are no primary studies proposing a *guaranteed* time for the self-adaptation process. This result is extremely interesting since in some application domains (*e.g.,* emergency-related, post-disaster apps) it may be a strict requirement to know and respect upper bounds on the execution time of the self-adaptation process, thus guaranteeing the timeliness associated with self-adaptation.

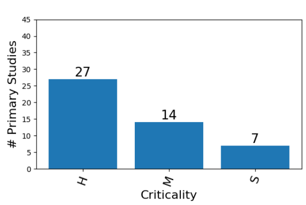
For what concerns self-adaptation **triggering** (see figure 5g), all the primary studies are based on *event triggered* mechanisms. This result is quite expected, given that mobile apps are mostly front-end software reacting either to user- or system-generated events.

D. RQ4 – Effects of Self-Adaptation (RQ4)

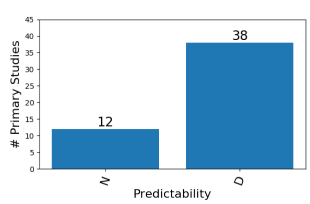
Starting with **criticality** (see Figure 6a), we observe that the majority of primary studies describe a self-adaptation process that has a *harmless* impact on the system in the case such adaptation were to fail. Fourteen and seven studies can have *mission-critical* and *safety-critical* consequences, respectively. Furthermore, we can also observe that safety-critical consequences are highly dependent on the domain in which the approach is being applied. This is logical, as, for example, a video-streaming adaptation process would most likely not run the risk of hurting its users in case of a failed adaptation.

In terms of **predictability**, the majority of the self-adaptation approaches are *deterministic* (see Figure 6b), followed by twelve approaches with non-deterministic effects. This means that in the majority of the cases the users of self-adaptive approaches know the possible states of the mobile app (and of the overall system) after adaptation.

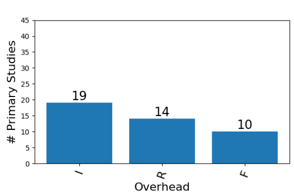
Examining the **overhead** attribute (see Figure 6c), we observe a nearly even distribution among all possible values, ranging from *insignificant*, to *reasonable*, and finally to system *failure* overhead. The relatively high number of primary studies whose approaches may lead to system *failure*s is mainly due to the relatively high number of approaches cloud offloading (or other network-related mechanisms), in which the apps' functionalities have to stop being provided



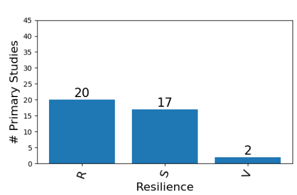
(a) Criticality



(b) Predictability



(c) Overhead



(d) Resilience

Fig. 6: Characteristics of the effects of the self-adaptation process

whilst the work is being offloaded and in some cases until its results have been obtained from the cloud.

Finally, under **resilience** there is a nearly even number of *resilient* and *semi-resilient* approaches, with hardly any being *vulnerable* (see Figure 6d). Resilience is defined as the persistence of service delivery that can justifiably be trusted, when facing changes [19]. As examples of approaches falling under the *vulnerable* category, four cases of faults and failures of context-aware adaptive applications are presented in [38]. One of those cases is related to an app supporting a so-called "meeting profile", which was autonomously applied whenever the app infers that the user is in a meeting with a colleague (based on the device's calendar, the current time, and on Bluetooth discovering another person in the room); however, the approach was falling into an adaptation cycle between the office and the meeting profiles since both of their conditions were triggered whenever a meeting was held in the office, leading to inconsistencies in the behaviour of the app.

V. DISCUSSION

In this section we present the research implications that we derived from our results (Section V-A), followed by an overview at the application domains we encountered in our primary studies (Section V-B) and the main challenges reported in the examined literature (Section V-C).

A. Research Implications

We conducted this systematic literature study to gain insight on self-adaptation in the context of mobile apps. To analyse our data we formulated four research questions (see section III-A). Here we report on the main findings related to our answers.

In our findings related to the **goals** of self-adaptation (*RQ1*) we observed that the vast majority of approaches have persistent and static goals. This implies that *the majority of the identified self-adaptive approaches are relatively rigid in their objectives*, unveiling a certain research gap involving approaches where goals can change at runtime depending on the ever-evolving context in which mobile apps are used today. Furthermore, the most frequently pursued goals are related to technical quality properties of mobile-enabled systems, such as performance and energy consumption. Interestingly, *there are few approaches targeting non-technical goals*, such as promoting user behavioural change and lifestyle improvement. Researchers in the field can direct their future studies towards filling this identified gap in the state of the art of self-adaptive mobile-enabled systems. For developers working on self-adaptive apps with static goals, the set of primary studies could represent a valuable source of knowledge.

In analyzing the **changes** that trigger self-adaptation (*RQ2*), we observe that the majority of approaches adapt due to changes within the hardware of the mobile device or its Internet connectivity. This result unveils a very interesting research gap related to potential *self-adaptive approaches which can adapt their behaviour or structural configuration to changes occurring in third-party applications running on the mobile* *device, in third-party services running in the cloud, or in smart objects surrounding the user of the app*. The above gap points to a potential unexplored market: developers looking for innovative self-adaptive apps could consider developing apps that adapt due to their third-party services, *e.g.,* to provide a better user experience, as opposed to competing apps that use these services without self-adaptation.

When discussing the **adaptation mechanisms** (*RQ3*), we observe some interesting findings as well. Firstly, the majority of approaches perform the adaptation in an autonomous manner, and therefore do not need a human in the loop or any other forms of human assistance to accomplish their adaptation. Moreover, in most cases the adaptation is performed by the application itself, with only sometimes requiring the help of the backend (such as in the case of cloud offloading applications). Secondly, all of the analysed adaptations are event-triggered and perform in a best-effort manner; therefore they do not have a guarantee on the duration of the self-adaptation process. This can be justified by the overwhelming short and medium duration of the adaptation processes we have studied, which by their nature are nearly impossible to guarantee in their timeliness. Nevertheless, an interesting research direction is about self-adaptive approaches where *formal and rigorous reasoning* plays a central role into making self-adaptation feasible also for apps belonging to critical domains (*e.g.,* energy, defense, transportation). Some works into this direction are starting to emerge in other fields, such as for the Internet of Things [42], [31], but *the application of formal reasoning in the context of self-adaptive mobile applications still seems to be an under-explored research area*.

Lastly, we need to discuss the **effects** of the self-adaptation process (answering *RQ4*). However, before doing so we must note that the reported information on effects was challenging to collect as it was rarely explicitly stated within the primary studies and, most of the time, had to be deduced by the description of the self-adaptation process and the analysed software system. *We advice researchers working on self-adaptive software systems to pay special attention to the effects dimension of self-adaptation*, so to provide a clear and complete overview of their proposed solutions. Having disclosed this, we nonetheless have a few noticeable results with the most prominent finding being that most of the analysed approaches had predictable consequences to their self-adaptation approach. Furthermore, the majority of analysed adaptation mechanisms had a harmless effect on the app in case the adaptation failed. However, we also found cases in which this effect was mission-critical or even safety-critical, and have noticed that there seems to be a strong link between a system being mission- or safety-critical and its application domain. This seems reasonable as *e.g.,* failed adaptation in the health domain is more likely to be safety-critical as opposed to a video streaming application, which poses no threat to the safety of the user in case of failed adaptation. This emerging link between domain and level of criticality, should be considered carefully by any developer working on self-adapting apps in mission/safety-critical domains.

B. Application Domains

When analyzing the primary studies we also traced the application domains in which the self-adaptive mobile apps have been applied. Of the cited application domains, *health* is the most popular with 7 primary studies. Other application domains in which self-adaptive mobile apps have been applied include: tourism, e-learning, mapping, education, science, conferencing, e-commerce, social networks, smart city, art, video streaming, image manipulation, and emergency management. Overall, such a high number of application domains hints to the general applicability of self-adaptive mechanisms, provided that the context and specifics of the application domain are taken into consideration (*e.g.,* apps should respect the intrinsic privacy-related concerns of domains like health and e-learning).

On the other hand, 22 primary studies do not mention any particular application domain. This indicates that a substantial amount of research has focused on self-adaptation mechanisms regardless of their application domain. This could be due to such mechanisms being broad enough to be applied in general. In the future, it might be interesting for researchers to investigate if there exist categories of self-adaptation techniques that are application-specific and others that are general-purpose.

C. Emerging Challenges

By extracting meaningful paragraphs from ‘future work’ and ‘challenges’ sections of our primary studies and then analyzing them, we have managed to find some common points of interest both for researchers and practitioners. Specifically:

- 12 primary studies mention the need or to further improve the implementation of their approach in order to reduce bias or eliminate potentially wrongful assumptions and of these studies 4 specifically mention only having implemented a research prototype;
- 9 of our primary studies mention the need of performing a more robust evaluation of their proposed approach;
- 4 primary studies mention the need to test their proposed approach on case studies as it was only tried in simulation or just theoretically;
- 3 primary studies mention the need for performing an in-depth comparison between their approach and the ones proposed by other researchers.

The information we have extracted seems to be reinforcing our previously given observation in subsection V-B. It would seem as a significant number of primary studies are working on a more theoretical level, therefore in a state of still needing further improvement and testing on practical scenarios and real-world applications. From this, we would therefore suggest that future research effort should be devoted not only to the improvement of the existing theoretical underpinnings of self-adaptation for mobile apps, but also to its application in real-world, realistic scenarios, at best by applying empirical case studies in industrial settings [46].

VI. Threats To Validity

The following reports on the potential threats to validity of this study according to [46].

Internal Validity. We mitigated internal threats to validity by using already established modelling dimensions [2] as our classification framework. For the validity of the synthesis of the collected data, we utilized well assessed descriptive statistics in order to minimize potential threats.

External Validity. In our study the main external threat to validity may come from our primary studies not being representative of the whole research on self-adaptation in the context of mobile applications. In order to mitigate such risk, we employed a search strategy including of both automatic search as well as backward-forward snowballing of the selected primary studies found with the automatic search. Furthermore, we chose to consider only peer-reviewed papers and excluded any work that could be defined as grey literature. We do not foresee this criterion to have impacted our study as the considered papers need to have undergone a rigorous peer-review process, which is an established requirement for quality publications. Lastly, we applied well defined inclusion and exclusion criteria, which we have rigorously followed during our manual selection phase.

Construct Validity. To be sure that the found primary studies would be able to competently answer the chosen research questions we manually carried out the selection process using the chosen inclusion and exclusion criteria, reported in subsection III-B. Such results were then further expanded by also conducting forward and backwards snowballing on those same selected studies.

Conclusion Validity. In order to reduce potential bias our classification framework is based on established modelling dimensions found in [2]. This way we can confidently guarantee that the data extraction process was aligned with our chosen research questions. Furthermore we reduced potential threats to conclusion by following well-known systematic literature review guidelines [16], [35], [46].

VII. Related Work

Works related to ours are secondary studies on self-adaptation.

Yang *et al.* [47] focused on requirements modeling and analysis for self-adaptive systems. They carried out a systematic literature review of 101 primary studies, from which they elicited 16 modelling methods and 10 requirement quality attributes. They observed that some of the modeling methods need further study, and most qualitative studies need better evaluation.

Krupitzer *et al.* [18] survey the engineering approaches found for self-adaptive systems. To this aim, they use a taxonomy for self-adaptation extended with the "context perspective", *i.e.,* the ability of systems to adapt their context. The survey identifies and classifies several approaches used to build self-adaptive systems.

Macias-Escriva *et al.* [23] analyze self-adaptability from the perspective of computer science and cybernetics, and examine the approaches found in the literature, to gain an overview of the state-of-the-art techniques used for self-adaptation. As one of the main conclusions, they identify feedback control and artificial intelligence as enabling fields to help further develop self-adaptive systems.

Both Krupitzer *et al.* and Macias-Escriva *et al.* do not report on the number of studies used in the data extraction for the surveys.

Mahdavi-Hezavehi *et al.* [24] conducted a systematic literature review of 54 primary studies, with the goal of understanding 'the state-of-the-art of architecture-based methods for handling multiple quality attributes (QAs) in self-adaptive systems'. They found that the most frequently addressed QAs are performance and cost, and the most common domains are robotics and web-based system.

Muccini *et al.* [30] focused on self-adaptation in the context of cyber-physical systems, and analyzed 42 primary studies. As part of their main results the authors found MAPE (Monitor-Analyze-Plan-Execute) as the most common mechanism used to perform adaptation in this context, and energy as that the most common application domain.

Lastly, Weyns *et al.* [43] examine the claims that are associated with self-adaptation. They analyzed 96 primary studies identified from the SEAMS conference series between 2006 and 2011, and the papers published in 2008 in [6]. They observe that (i) the main focus is on architecture and models, (ii) the most common application domain is service-based systems, and (iii) at the time of publishing only a few empirical studies were performed with no industrial evidence.

In spite of the relatively large number of secondary studies on self-adaptation and self-adaptive systems, none explored the state of the art of self-adaptation in the context of mobile applications. Our study certainly fills this gap, which with the increasing pervasiveness of mobile software in all application domains is turning into a necessity.

VIII. Conclusions and Future Work

This paper presents a systematic literature review on self-adaptation in the context of mobile applications as defined in section II. Starting from 607 possibly relevant studies, we found 44 primary studies which we analyzed via the presented classification framework, in order to answer our chosen research questions. By answering these questions, we give an in-depth look at the field of self-adaptation in the mobile application context, and therefore provide valuable information for researchers and developers who wish to work in the future within this area.

As future work, we will perform a longitudinal analysis across the various dimensions of our classification framework as it would help discover more complex (and hidden) patterns among the analyzed approaches. Furthermore, a more in-depth analysis of the contents of the primary studies could contribute in better understanding the current research gaps about self-adaptation in the context of mobile applications. Finally, we are actively working on an approach in which dynamic clustering techniques will be used for automatically personalizing an e-health mobile app so to better support users in following medical advice.

REFERENCES

[1] *Android connectivity*, 2018. Available at https://developer.android.com/training/basics/network-ops/connecting.

[2] J. Andersson, R. De Lemos, S. Malek, and D. Weyns. Modeling dimensions of self-adaptive software systems. In *Software engineering for self-adaptive systems*, pages 27–47. Springer, 2009.

[3] R. Block. *Live from Macworld 2007: Steve Jobs keynote*, 2007. Available at https://www.engadget.com/2007/01/09/live-from-macworld-2007-steve-jobs-keynote/.

[4] G. Caldiera, V. R. Basili, and H. D. Rombach. Goal question metric paradigm. *Encyclopedia of software engineering*, 1:528–532, 1994.

[5] J. C. Casquina, J. D. S. Eleuterio, and C. M. Rubira. Adaptive deployment infrastructure for android applications. In *Dependable Computing Conference (EDCC), 2016 12th European*, pages 218–228. IEEE, 2016.

[6] B. H. C. Cheng, R. d. Lemos, H. Giese, P. Inverardi, and J. Magee, editors. *Software Engineering for Self-Adaptive Systems*. Lecture Notes in Computer Science. 2009.

[7] D. Cooray, E. Kouroshfar, S. Malek, and R. Roshandel. Proactive self-adaptation for improving the reliability of mission-critical, embedded, and mobile software. *IEEE Transactions on Software Engineering*, 39(12):1714–1735, 2013.

[8] G. Cugola, C. Ghezzi, L. S. Pinto, and G. Tamburrelli. Adaptive service-oriented mobile applications: A declarative approach. In *International Conference on Service-Oriented Computing*, pages 607–614. Springer, 2012.

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

[10] H. Flores and S. Srirama. Adaptive code offloading for mobile cloud applications: Exploiting fuzzy sets and evidence-based learning. In *Proceeding of the fourth ACM workshop on Mobile cloud computing and services*, pages 9–16. ACM, 2013.

[11] V. Glavinic, S. Ljubic, and M. Kukec. Transformable menu component for mobile device applications: Working with both adaptive and adaptable user interfaces. *International Journal of Interactive Mobile Technologies (iJIM)*, 2(3):22–27, 2008.

[12] T. Greenhalgh and R. Peacock. Effectiveness and efficiency of search methods in systematic reviews of complex evidence: audit of primary sources. *Bmj*, 331(7524):1064–1065, 2005.

[13] E. Grua, I. Malavolta, and P. Lago. *Replication package of the study*, 2019. Available at http://s2group.cs.vu.nl/seams-2019-replication-package/.

[14] O. internationale de normalisation. *Systems and Software Engineering: Systems and Software Quality Requirements and Evaluation (SQuaRE): System and Software Quality Models*. ISO/IEC, 2011.

[15] H. Khalid, E. Shihab, M. Nagappan, and A. E. Hassan. What do mobile app users complain about? *IEEE Software*, 32(3):70–77, 2015.

[16] B. Kitchenham and P. Brereton. A systematic review of systematic review process research in software engineering. *Information and software technology*, 55(12):2049–2075, 2013.

[17] B. A. Kitchenham and S. Charters. *Guidelines for performing systematic literature reviews in software engineering*. Technical Report EBSE-2007-01, Keele Uni- versity and University of Durham, 2007.

[18] C. Krupitzer, F. M. Roth, S. VanSyckel, G. Schiele, and C. Becker. A survey on engineering approaches for self-adaptive systems. *Pervasive and Mobile Computing*, 17:184–206, 2015.

[19] J.-C. Laprie. From dependability to resilience. In *38th IEEE/IFIP Int. Conf. On Dependable Systems and Networks*, pages G8–G9. Citeseer, 2008.

[20] A. Lella and A. Lipsman. The 2016 U.S. Mobile App Report, 2016. comsCore white paper.

[21] F. S. Lopez and N. Condori-Fernández. Design of an adaptive persuasive mobile application for stimulating the medication adherence. In *International Conference on Intelligent Technologies for Interactive Entertainment*, pages 99–105. Springer, 2016.

[22] Y. Ma, X. Liu, Y. Liu, Y. Liu, and G. Huang. A tale of two fashions: An empirical study on the performance of native apps and web apps on android. *IEEE Transactions on Mobile Computing*, 17(5):990–1003, 2018.

[23] F. D. Macías-Escrivá, R. Haber, R. Del Toro, and V. Hernandez. Self-adaptive systems: A survey of current approaches, research challenges and applications. *Expert Systems with Applications*, 40(18):7267–7279, 2013.

[24] S. Mahdavi-Hezavehi, V. H. Durelli, D. Weyns, and P. Avgeriou. A systematic literature review on methods that handle multiple quality attributes in architecture-based self-adaptive systems. *Information and Software Technology*, 90:1–26, 2017.

[25] I. Malavolta. Beyond native apps: web technologies to the rescue!(keynote). In *Proceedings of the 1st International Workshop on Mobile Development*, pages 1–2. ACM, 2016.

[26] I. Malavolta, S. Ruberto, V. Terragni, and T. Soru. End users' perception of hybrid mobile apps in the google play store. In *Mobile Services (MS), 2015 IEEE International Conference on*, pages 25–32. Institute of Electrical and Electronics Engineers (IEEE), June 2015.

[27] R. Mizouni, M. A. Matar, Z. Al Mahmoud, S. Alzahmi, and A. Salah. A framework for context-aware self-adaptive mobile applications spl. *Expert Systems with applications*, 41(16):7549–7564, 2014.

[28] F. A. Moghaddam, M. Simaremare, P. Lago, and P. Grosso. A self-adaptive framework for enhancing energy efficiency in mobile applications. In *2017 Sustainable Internet and ICT for Sustainability (SustainIT)*, pages 1–3. IEEE, 2017.

[29] M. Moghimi, J. Venkatesh, P. Zappi, and T. Rosing. Context-aware mobile power management using fuzzy inference as a service. In D. Uhler, K. Mehta, and J. L. Wong, editors, *Mobile Computing, Applications, and Services*, pages 314–327, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[30] H. Muccini, M. Sharaf, and D. Weyns. Self-adaptation for cyber-physical systems: a systematic literature review. In *Proceedings of the 11th international symposium on software engineering for adaptive and self-managing systems*, pages 75–81. ACM, 2016.

[31] H. Muccini, R. Spalazzese, M. T. Moghaddam, and M. Sharaf. Self-adaptive iot architectures: an emergency handling case study. In *Proceedings of the 12th European Conference on Software Architecture: Companion Proceedings*, page 19. ACM, 2018.

[32] N. Z. Naqvi, J. Devlieghere, D. Preuveneers, and Y. Berbers. Mascot: self-adaptive opportunistic offloading for cloud-enabled smart mobile applications with probabilistic graphical models at runtime. In *System Sciences (HICSS), 2016 49th Hawaii International Conference on*, pages 5701–5710. IEEE, 2016.

[33] F. Palomba, M. L. Vásquez, G. Bavota, R. Oliveto, M. Di Penta, D. Poshyvanyk, and A. De Lucia. Crowdsourcing user reviews to support the evolution of mobile apps. *Journal of Systems and Software*, 137:143–162, 2018.

[34] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson. Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, EASE'08, pages 68–77, Swinton, UK, UK, 2008. British Computer Society.

[35] K. Petersen, S. Vakkalanka, and L. Kuzniarz. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18, 2015.

[36] G. P. Picco, C. Julien, A. L. Murphy, M. Musolesi, and G.-C. Roman. Software engineering for mobility: reflecting on the past, peering into the future. In *Proceedings of the on Future of Software Engineering*, pages 13–28. ACM, 2014.

[37] S. Raheel. Improving the user experience using an intelligent adaptive user interface in mobile applications. In *Multidisciplinary Conference on Engineering Technology (IMCET), IEEE International*, pages 64–68. IEEE, 2016.

[38] M. Sama, D. S. Rosenblum, Z. Wang, and S. Elbaum. Multi-layer faults in the architectures of mobile, context-aware adaptive applications. *Journal of Systems and Software*, 83(6):906–914, 2010.

[39] Statista. Number of available applications in the google play store from december 2009 to june 2018, 2018.

[40] J. L. Wesson, A. Singh, and B. Van Tonder. Can adaptive interfaces improve the usability of mobile applications? In *Human-Computer Interaction*, pages 187–198. Springer, 2010.

[41] D. Weyns. Software engineering of self-adaptive systems: an organised tour and future challenges. *Chapter in Handbook of Software Engineering*, 2017.

[42] D. Weyns, M. U. Iftikhar, D. Hughes, and N. Matthys. Applying architecture-based adaptation to automate the management of internet-of-things. In *European Conference on Software Architecture*, pages 49–67. Springer, 2018.

[43] D. Weyns, M. U. Iftikhar, S. Malek, and J. Andersson. Claims and supporting evidence for self-adaptive systems: A literature study. In *Proceedings of the 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 89–98. IEEE Press, 2012.

[44] C. Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, page 38. ACM, 2014.

[45] C. Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, EASE '14, pages 38:1–38:10, New York, NY, USA, 2014. ACM.

[46] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[47] Z. Yang, Z. Li, Z. Jin, and Y. Chen. A systematic literature review of requirements modeling and analysis for self-adaptive systems. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 55–71. Springer, 2014.