PROTOCOL FOR A MIXED-METHOD EMPIRICAL STUDY ON

Guidelines for Architecting Android Apps

AUTHORS BLINDED FOR REVIEW

PROTOCOL FOR A MIXED-METHOD EMPIRICAL STUDY ON

Guidelines for Architecting Android Apps

ABSTRACT

This document describes the review protocol of a mixed-method empirical study on guidelines for architecting Android apps.

KEYWORDS

Software Architecture, Android, Mixed-Method Empirical Study.

Contents

List of Figures

List of Tables

1 Background and rationale

The mobile apps ecosystem is reaching incredible numbers, both in terms of users and published apps, and it is showing no signs of slowing down market-growth anytime soon. For example, as end users, we are spending more than 2 hours *a day* on mobile apps [1]. Android is accounting for more than 85.9% of global smartphone sales worldwide [2], leading thousands of developers to choose Android as their first go-to development platform [3]. In the last quarter of 2018 more than 2.6 million Android apps were available in the Google Play, the official Android app store [4].

For surviving in such a highly competitive market, it is fundamental for app developers to deliver apps yielding high quality in terms of e.g., performance, energy consumption, user experience. Developers are investing great efforts to deliver apps of high quality and with short release times. In this context, a well-architected Android app is beneficial for developers in terms of maintainability, evolvability, bug fixing (e.g. resource leaks), testability, performance, etc. The most recent releases of the Android platform are putting more and more emphasis on the architecture of the apps, with a special focus on architecturally-relevant components[1], such as those belonging to Android Jetpack, the recently introduced collection of Android software components[2]. However, *how to properly architect Android apps is still highly debated and subject to conflicting opinions*, usually influenced by technological hypes rather than objective evidence.

The goal of the study described in this protocol is twofold: (i) to characterize the state of the practice on architecting Android apps and (ii) to provide a set of evidence-based guidelines for supporting developers while architecting Android apps. Given the relatively low maturity of the subject and its tight connection with industry, we apply a *mixed-method empirical research design* that combines (i) semi-structured interviews with Android practitioners in the field, and (ii) a systematic analysis of both the grey (e.g., websites, on-line blogs) and white literature (i.e., academic studies) on the architecture of Android apps.

The envisioned outcomes of the study described in this protocol are:

- interviews of practitioners that provide qualitative information about architecting Android apps;
- a systematic analysis of the grey and white literature about architecting Android apps;
- a set of evidence-based guidelines for architecting Android apps;
- a replication package of the study containing its results, raw data, and the protocol described in this document.

The envisioned audience of the study described in this protocol includes both Android developers and researchers. Specifically, this study benefits (i) developers by providing evidence-based guidelines for taking action towards improving the architecture of their Android apps and (ii) researchers by objectively characterizing the state of the art and practice about architecting Android apps.

1.1 Existing studies on the topic and the need of a mixed-method study

Despite the wide diffusion of Android apps and their increasing complexity [5], at the time of writing we found a surprising low number of **research studies about the architecture of mobile apps**. By mining and reverse engineering the architecture of more than 1400 Android apps, Bagheri et al. studied the role of software architecture in the design and development of mobile software, extracted the architectural principles that have been applied by app developers, and identified architectural anti-patterns of the Android programming model [6]. They found that Android

[1] https://developer.android.com/topic/libraries/architecture
[2] https://developer.android.com/jetpack

apps are complex, composed of tens of components, and organized according to multiple architectural styles. These findings motivated us to investigate how developers architect Android apps, eventually leading to the results in this study. Even though our study and that of Bagheri et al. share the same target audience, i.e., Android developers and researchers, the research goals are profoundly different: Bagheri et al. focus on known architectural principles and how they are reflected in the Android programming model, whereas we aim at characterizing the state of the art and practice on architecting Android apps. Moreover, the methodologies applied in the two studies are completely different – Bagheri et al. mined the apps from app stores and statically analyzed their bytecode, whereas we contact Android practitioners complemented with a systematic analysis of the grey and white literature.

An exploratory study targeting common architectural characteristics of 12 real Android applications is reported in [7]. The study is based on the partial extraction of the architecture of the apps using the JDepend tool, followed by the manual analysis of the source code of the targeted apps. The main results of such manual analysis revealed that MVC is a recurrent pattern in Android (although with some violations). In our work we apply a totally different research methodology, where we target professional developers working on industrial projects, rather than developers working on open-source apps. Also, the work proposed in [7] is exploratory in nature and aims at observing the characteristics of the architecture of Android apps, whereas we aim at providing actionable guidelines for helping developers during their everyday activities.

A new MVC-based architectural pattern called Android Passive MVC, is proposed in [8], with the aim of producing Android apps with better maintainability, extensibility, performance, and less complexity. The proposed pattern has been applied to an example of social networking app in collaboration with a development company. Differently from [8], we do not aim at providing a new architectural pattern, rather we accept the existence of many pre-existing ones in Android apps (also confirmed in [6, 7]) and aim at supporting developers while architecting Android apps, without forcing them to learn and apply new (potentially unsupported) architectural patterns.

The authors of [9] performed a preliminary study on how to develop Android apps according to the Software Product Line (SPL) approach. Their case study shows that an adaptation of the principles for SPLs can be adopted for developing Android apps. We differ as we do not imply a change in developers' habits by means of a new development paradigm like SPL, but aim at supporting them in taking better-informed decisions about the architecture of their apps.

A study about the challenges faced by mobile app developers (not only Android) has been proposed by Joorabchi et al. [10]. The challenges have been extracted in a qualitative manner from a combination of 12 interviews with practitioners and an on-line questionnaire with 188 participants. Differently from us, they focus on mobile apps in general (incl. web and hybrid apps) and are orthogonal to architectural concerns, i.e., they do not cover the challenges directly related to architecture, but focus on challenges related to e.g., testing. Interestingly, in our study we found some confirmation of their insights, e.g., the importance of testability, partially managed by following the MVP or MVVM styles.

From a methodological perspective, **multivocal studies** (i.e., systematic studies targeting both grey and white literature) are being published only recently in the field of software engineering [11], e.g., investigating code smells in testing artifacts [12], the startups ecosystem [13], and microservices [14]. Researchers are also complementing multivocal studies with other research methodologies (e.g., semi-structured interviews in our case), thus leading to mixed-method studies. For example, Maro et al. combined a tertiary literature review, a case study with a company, and a multivocal literature review to identify challenges and solutions about software traceability in the automotive domain [15]. In the literature there is no multivocal study on mobile apps. So, even though also our research combines a multivocal study with other research methodologies (e.g., interviews), the subjects and therefore the outcomes of our study are novel by focussing on architecting practices for Android apps.

2 Research Process Overview

This research will be carried out by following the process shown in Figure 1; it can be divided into five main phases, which are well-established when it comes to systematic literature studies [16, 17]: planning, conducting, and documenting.

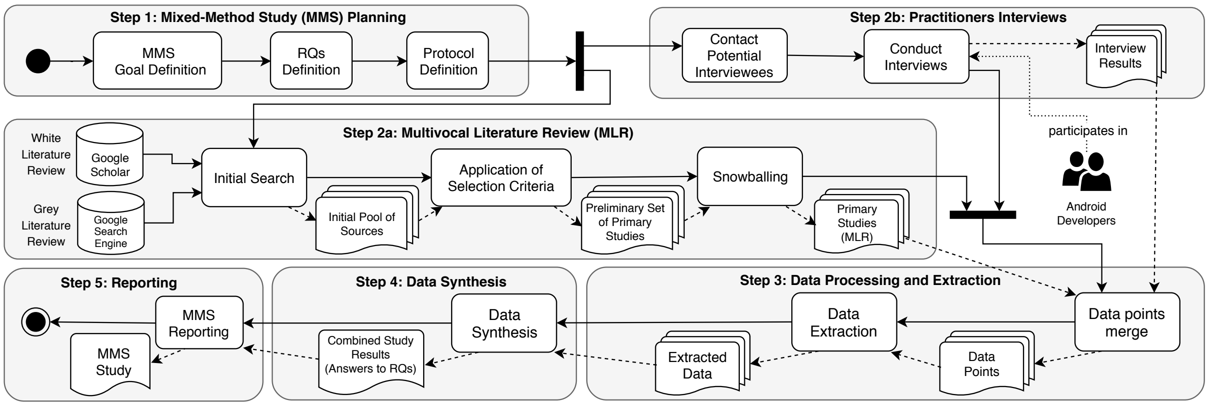

Figure 1: Overview of the whole review process

Each phase has a number of output artefacts, e.g., the planning phase produces the review protocol described in this document. In the following we will go through each phase of the process, highlighting main activities and produced artefacts.

2.1 Step 1: Planning

This phase aims at (i) establishing the need for performing a review on guidelines for architecting Android apps (see Section 1), (ii) identifying the main research questions (see Section 3), and (iii) defining the protocol to be followed by the involved researchers. The output of the planning phase is a detailed protocol (i.e., this document).

2.2 Step 2a: Multivocal Literature Review (MLR)

In this phase we will perform the MLR study by following all the steps previously defined. More specifically, we will carry out the following activities:

- *Search*: we will perform a combination of automated search and backward and forward snowballing for identifying the set of potentially relevant research articles on architectural practices for architecting Android apps. During this process, both results from a white-literature (WL) and grey literature (GL) sources will be considered.

- *Selection*: The identified candidate entries will be filtered according to a selection criteria in order to obtain the preliminary list of primary studies to be considered in later activities of the review.

- *Snowballing*: In oder to complement the preliminary list of primary studies, a forward and backward snowballing process [18] will be adopted. This procedure consists of inspecting the references from/to the preliminary set of primary studies and analyze them by applying the selection criteria in order to integrate potentially relevant researches in the final set of preliminary studies.

An in-depth description of the steps reported above is provided in Section 4.

2.3 Step 2b: Practitioners Interview

This steps entails contacting experienced Android developers (i.e. working with the Android platform for at least 5 years) and conducting the interviews according to the protocol. More details on this process are provided in Section 5

2.4 Step 3: Data Processing and extraction

This step consists of uniforming the data originated from Steps 2a and 2b and subsequently conducting a data extraction process on the processed data. The specifics of this procedures are reported in Section 6.

2.5 Step 4: Data Synthesis

This step regards the synthesis and analysis of the extracted data in order to answer our research questions. Specifically, in order to answer RQ_1 and RQ_3 (see Section 3) a purely quantitative analysis will be conducted. Instead, in order to answer RQ_2, a keywording process will be applied in order to condense the extracted architectural principles into guidelines. More detail on this process is reported in Section 7.

2.6 Step 5: Reporting

The main activities performed in this phase are: (i) a thorough elaboration of the data extracted in the previous phase with the main aim of setting the obtained results in their context, (ii) the discussion of possible threats to validity, specially to the ones identified during the definition of the review protocol (in this activity new threats to validity may emerge too), and (iii) the writing of a final report describing the performed mixed-method empirical study.

2.7 Team

Three researchers will carry out this study, each of them with a specific role within the research team:

- *Principal researcher*: "*Name and other sensible information blinded for review*." They will be part of all the activities, i.e., planning the study, conducting it, and reporting;

- *Research methodologist*: "*Name and other sensible information blinded for review*." They is mainly involved in (i) the planning phase of the study, and (ii) supporting the principal researcher during the whole study, e.g., by reviewing the data extraction form, selected primary studies, extracted data, produced reports, etc., (iii) replicating the MLR selection procedure to measure inter-researcher agreement level through Cohen's kappa, (iv) executing part of the data extraction and synthesis processes as well as (v) supporting the reporting process.

- *Advisor*: "*Name and other sensible information blinded for review*." They takes final decisions on conflicts and methodological options to 'avoid endless discussions' [19], and supports the other researchers during data and findings synthesis activities.

3 Research questions

This study aims at characterizing the current state of the art for understanding the state of the art of practices related to Android architecture. The results of this study are targeted to both (i) researchers willing to further contribute to this research area, and (ii) Android developers willing to understand existing practices on Android architecture. More formally, we formulate the goal of this study by using the Goal-Question-Metric perspectives (i.e., purpose, issue, object, viewpoint [20]). Table 1 shows the result of the above mentioned formulation.

Purpose	Identify, classify, and evaluate
Issue	chearacteristics
Object	of existing Android architecture practices
Viewpoint	from a researcher's and practitioner's point of view.

Table 1: Goal of this research

This abstract goal can be refined into the following research questions (for each research question we also provide the rationale for it being part of this study):

RQ1: *Which are the **general characteristics of Android architecture**?* This can be refined into two sub-questions:

- RQ1.1: *Which **architectural styles/patterns** are used for architecting Android apps?*
  Rationale: This research question aims at understanding which architectural styles/-patterns[3] are utilized for architecting Android apps. This provides us with a better understanding of current Android architecture practices, and sets the context for the next research questions.
- RQ2.2: *Which **libraries** are referenced while discussing Android architecture?*
  Rationale: With this question we aim at identifying the programming libraries regarded as most influential while architecting Android apps. This provides further data on the technologies used to support Android architecting processes.

RQ2: *Which are the **architectural guidelines** for developing Android apps?*

Rationale: This research question constitutes the core of the study. By answering it, we aim at synthesizing a set of architectural guidelines for architecting Android apps. This provides practitioners with actionable guidance when architecting their Android apps, and supports researchers in future investigations on Android architecture.

RQ3: *Which **quality requirements** are considered when developing and reasoning about the architecture of Android apps?* Rationale: Today developing apps of high quality is fundamental for surviving in the Android market. This research question aims to understand which quality requirements (QRs [21], e.g., performance, usability, maintainability) are taken into account when dealing with the architecture of Android apps. This provides a good understanding of which QRs are potentially impacted the most by architectural decisions.

Identified research questions will drive the whole study, with a special influence on (i) search and selection of data points (i.e. primary studies and interviews), (ii) data extraction, and (iii) data analysis.

4 Step 2a: Multivocal Literature Review

The main goal of our search and selection process is to retrieve a comprehensive set of research studies that are relevant and representative enough for the topic being considered. More specifically, it is fundamental to achieve a good trade-off between the coverage of existing research on the topic considered, and to have a manageable number of studies to be analysed. In order to achieve the above mentioned trade-off, our search strategy consists of two complementary methods: an

[3]For the sake of space, from this point onwards, "architectural style" and "architectural pattern" will be jointly referred to as "architectural pattern".

automated search process considering both WL and GL, and a manual snowballing process on the identified researches. Our search and selection process has been designed as a multi-stage process in order to have full control on the number and characteristics of the studies being either selected or excluded during the various stages. In the following we detail each step of our search and selection process.

The search strategy is divided into two subsequent and complementary steps. The first step is carried out by automatically inspecting all the publications returned from a query execution on both Google Scholar and the Google Search engine. The primary identified through this first step will then be subsequently utilized as input for a backward and forward snowballing[4] process [18].

In order to ensure the correctness of the adopted automated search approach, the backward snowballing activity will be based exclusively on the papers selected through a well defined set of inclusion and exclusion criteria. Furthermore, the backward snowballing results will be also contemplated by adopting a forward snowballing process, that will ensure the soundness and relevance of the set of selected primary studies.

The search and selection processes are conceived as multi-stage processes in order to control through a rigorous and pre-defined methodology the number and characteristics of the studies being either selected or excluded during the various stages. In the following sections each of the steps composing the search and selection processes are presented and detailed.

1. Initial search. In this initial stage we will perform an automated search by executing a search query on two search engines, one to identify potentially relevant studies in the white body of literature and one to identify potentially relevant studies in the gray body of literature and one. In order to identify the potentially relevant white literature (WL) studies, a research query will executed on Google Scholar. We opt for such digital library as (i) its adoption constitutes a sound choice to determine the initial set of literature for snowballing processes [18], and (ii) the results of the query can be processed automatically via tool-support. Additionally, we will execute the search query on two additional digital libraries, namely Scopus and IEEE Explore, in order to assess if these latter sources result to be more inclusive. In affirmative case, results from such digital libraries will be included in the initial pool of potentially relevant studies.

Below we report the search string we will use.

(intitle:*architect**) **AND** (intitle:*Android* **OR** intitle:*"mobile app"*)

The query is purposely designed to be generic, in order to be as encompassing as possible while selecting a significant set of potentially relevant studies. Regarding the initial search of grey literature (GL), the query will be executed on the regular Google Search Engine by omitting Google Scholar specific syntax (i.e. the "intitle" keywords). The search engine is selected in accordance to the recommendations for including GL in software engineering multivocal reviews [22]. Due to the high volume of returned results, we will limit the search to the top 100 results as stopping rule.

The potentially relevant publications will be thoughtfully examined by adopting several exclusion rounds in an adaptive reading depth fashion [23]. In the first round, the title of the publications will be examined. This first step will enable us to discard all those publications that clearly do not fall in the domain of Android architecture. In the second exclusion round, the introductions and the conclusions (if present) of the remaining publications will be inspected. Finally, the selected publications will be further inspected by considering their full text in order to ensure that only the ones relevant for answering the research questions.

The search query, both for WL and GL, will be executed at the beginning of September 2018. In order to reduce potential threats to validity and for the sake of replicability, all the cookies of the browser from which the query will be executed will be cleaned. Additionally, the search execution location will be set to "United States" from the Google Search Engine settings.

[4] Inspection of the studies referenced by a paper (*backward snowballing*) and of the studies referencing it (*forward snowballing*)

2. Application of selection criteria. Once the pubblications will be selected through the initial search phase, the resulting studies will be filtered according to a set of well-defined selection criteria. The adopted criteria are detailed in Section 4.0.1. An adaptive reading depth [23] will be utilised, in order to carry out the exclusion process in a time-efficient and objective manner.

3. Backward snowballing. In order to mitigate a potential bias with respect to the construct validity of the study, the automated search previously presented is complemented with an additional snowballing process [24]. The snowballing activity will be adopted in order to further expand the number of considered publications by taking into account also publications that were not returned by the automatic search. In particular, this process will be carried out by considering the publications selected in the initial search, and subsequently selecting relevant papers among those referenced by the initially selected ones. This method is commonly referred to as a *backward snowballing* activity [25]. As for the selection of the papers through the initial phase, an adaptive reading depth [23] is adopted to select the papers. The final decision about the inclusion of the publications will be based on the adherence of the full text of the studies to the predefined selection criteria presented in Section 4.0.1. This process will be carried out both for WL and GL.

4. Forward snowballing. In addition to the backward snowballing, we also analyzed the publications citing the studies selected through the initial search will also be analyzed. This process is usually referred to as a *forward snowballing* activity [25]. We include this further literature search method in order to further expand and refine the selection of studies gathered through the initial search and the backward snowballing activity. Regarding the forward snowballing process, the *Google Scholar*[5] bibliographic database will be adopted to retrieve the studies citing the ones selected through the initial search phase. As for the other search activities, an adaptive reading depth [23] will be adopted while evaluating the researches against the inclusion and exclusion criteria. Elements discovered through this search activity that will not correspond to research papers, such as textbooks or technical reports, will not be included in the set of primary studies. This process will be carried out exclusively for WL, as, due to the very large number of backward links present for the GL items, a forward snowballing process for GL results to be infeasible.

4.0.1 Selection criteria

Following the guidelines for systematic literature review for software engineering [16], in order to reduce the likelihood of biases, we have to define a rigorous set of inclusion and exclusion criteria during the initial protocol definition phase of the literature review. Additionally, in order to ensure the quality of the primary studies selected from the GL, a subset of 5 quality-evaluation criteria presented by Garousi et al. [22] will be adopted. A 3-point Likert scale (yes=1, partly=0.5, and no=0) will be used in order to assign quality scores. A GL pubblication will be considered of sufficient quality if it scores at least 2 out of 5 total points. We adopted to utilize only a subset of the criteria presented in [22] as we plan to consider a high number of potentially relevant GL publications. Hence a more extensive quality assessment process would result to be too time consuming to be considered. Additionally, by inspecting the criteria presented by Garousi et al., we can notice that most quality criteria are tailored to the topic considered in their example, and hence result to be ill-suited to be included in our protocol due to the encompassing nature of our study.

In the following we detail the set of inclusion, exclusion and quality criteria that will guide the selection of the primary research studies for the proposed MLR. A publication will be included in the set of primary studies if it satisfies *all* the inclusion criterion stated below. A study will be discarded if it satisfies *at least one* of the exclusion criteria reported below. Additionally, grey literature publications, in order to be included, will also have to satisfy a quality score of at least 2 out of five points.

The selection inclusion and exclusion criteria are divided into three categories, namely: *general*

[5] `https://scholar.google.it/`

(i.e. applying for both WL and GL), *WL-specific*, and *GL-specific*. The decision of adopting three categories of criteria originates from the different nature of the sources of primary studies considered (i.e. Google Scholar and the Google Search Engine). By defining three different sets it is possible to design selection criteria specifically tailored for the perils specific to the selection of GL and WL respectively, and hence improve the overall quality of the selection process.

General Inclusion criteria

GEN-I1) Studies focusing on the architectural level of software systems.

Rationale: This inclusion criterion is utilized to select exclusively studies discussing software on architectural level. It should define architecture, discuss the representation or discuss architectural patterns.

GEN-I2) Studies focusing on the Android operating system.

Rationale: This inclusion criterion is utilized to select exclusively studies discussing Android, since Android is not the only mobile operating system. Research might be focusing on comparing them, but we are interested in only Android.

General Exclusion criteria

GEN-E1) Studies that have not been published in English language.

Rationale: The inspection of such studies would result would result to be too time consuming.

GEN-E2) Duplicate or extensions of already included studies.

Rationale: The inclusion of such studies would hinder the conclusion validity of our research and hence should be considered as a threat to be mitigated.

GEN-E3) Studies which are not available.

Rationale: Under this category fall studies which are indexed by the digital library adopted, but are not accessible, and hence not analyzable.

GEN-E4) Studies not focusing on native Android applications, e.g., Unity-based videogames, and web-based apps.

Rationale: We want to exclude from our research architectural principles which, while adopted in a mobile context, are not specific to the Android framework, and hence are out of scope for our study.

WL-specific Exclusion criteria

WL-E1) Secondary or tertiary studies.

Rationale: This exclusion criterion is adopted in order to exclude studies which do not report the desired level of detail on Android architecture.

WL-E2) Studies in the form of editorials, tutorials, books, etc.

Rationale: Such studies are not deemed to provide the required level of detail and information, and their inclusion might hinder the quality of the extracted data.

WL-E3) Studies which have not been peer reviewed.

Rationale: This exclusion criterion is adopted in order to ensure the quality of the included WL, as peer-reviewing processes are nowadays a *de facto* standard of high-quality scientific literature.

GL-specific Exclusion criteria

GL-E1) Studies reporting exclusively the basic principles about the Android platform and its architecture

Rationale: This exclusion criteria is adopted in order to avoid the inclusion of items which do not add any significant contribution in order to answer our research questions.

GL-E2) Studies reporting exclusively abstract best practices

Rationale: This exclusion criteria is adopted in order to exclude the studies which do not report any concrete practices specific to Android architecture, and hence would result in mere noise during the data extraction phase.

GL-E3) Studies reporting only trivial Android implementations

Rationale: This exclusion criteria is adopted in order to exclude trivial programming tutorials on the basic aspects of developing Android apps.

GL-E4) Studies reporting an implementation without a discussion of its benefits and/or drawbacks

Rationale: This exclusion criteria is adopted in order to exclude sources whihc report exclusively implementations of Android architectural blueprints / practices without any type of discussion, as the data extraction would be potentially be a cumbersome and biased process.

GL-E5) White literature

Rationale: This exclusion criteria is adopted in order to exclude indexed WL, as such type of literature will be considered by a different process.

GL-E6) Videos, webinars, etc.

Rationale: The data extraction from videos, webinars, etc., would result too time-consuming in order to be considered for this study.

GL-specific quality criteria To further ensure the quality of the included grey literature, we will further assess the quality of GL by utilizing a subset of quality evaluation criteria taken from [22]. We adopted to utilize only a subset of the criteria presented in [22] as we plan to consider a high number of potentially relevant GL publications. Hence a more extensive quality assessment process would result to be too time consuming to be considered. Additionally, by inspecting the criteria presented by Garousi et al., we can notice that most quality criteria are tailored to the topic considered in their example, and hence result to be ill-suited to be included in our protocol due to the encompassing nature of our study.

GLQ-E1) Is the publishing organization or the author reputable?

Rationale: This quality criteria is adopted to assess to which extent the publishing organization of the study is reputable. As an example, official android documentation provided by Google will rank higher in reputability than a personal programming blog.

GLQ-E2) Has the author published other studies in the field?

Rationale: This quality criteria is adopted to assess the level of expertise in Android architecture of the author.

GLQ-E3) Does the study add value to the research?

Rationale: This quality criteria is adopted to assess to which extent the publication adds value to our research.

GLQ-E4) Is the presentation of the study of high quality?

Rationale: In order to fully understand a publication, the presentation quality should be at least sufficient, which unfortunately is not always the case for GL.

GLQ-E5) Is the study supported by evidence, e.g. examples/data?

Rationale: Studies supported by evidence, such as empirically gathered results or concrete implementation examples, will be considered of higher quality w.r.t. GL not supported by any evidence.

In order to reduce possible bias, two researchers will perform independently the selection of a random subsample of the potentially relevant literature by applying the above mentioned criteria. Such process will be adopted to measure the inter-researcher agreement via the measurement of Cohen's kappa.

5 Step 2b: Practitioners interview

In order to avoid potential biases which may emerge during the conduction of the interviews, we design *a-priori* an interview guide, which generated from the iteration of internally- and externally-conducted pilots. The interview guide, consisting of a set of definitions and questions, is reported in the reminder of this section. After the finalization of the design of the interview guide, we will contact by adopting convenience sampling, Android developers within our network. In order to mitigate potential threats to validity, only developers with at least 5 years of Android experience, belonging to different companies and developing apps belonging to heterogeneous business domains will be contacted. In order to introduce the context of this research and guide the interview, a set of slides was used to present the essential concepts and questions. Prior to the interview, interviewees were introduced to some brief definitions, in order to ensure the common agreement on some terms, described below.

Definitions

- *Android Architecture*: While a fine-grained definition has to be provided by the interviewee, the broad definition is Android specific components and their relations (e.g. the 4 Android building blocks and their dependencies)
- *Architectural practice*: Architectural development practices, such as: Architectural patterns and anti-patterns, General architectural principles followed during architecting activities, Architectural styles (e.g. MVP)
- *Quality requirements*: Defined also as non functional-requirements. QRs are broadly defined as criteria that can be used to judge the operation of a system, rather than specific behaviors. This category encompasses the QRs defined in the standard ISO 25010 [21].

Interview questions

IQ-1) Demographics:

- Company name: Name of the company the interviewee is working for
- Category of app(s): Category of app(s) the developer is currently focusing on
- Experience: Number of years as an Android developer
- Current role: Current role of the interviewee
- Development setting: Libraries adopted and framework utilized, e.g. Native Android setting, React Native, Corona SDK, Xamarin, Appcelerator Titanium, NativeScript, Ionic, ...

RQ mapping: RQ_1, $RQ_{1.2}$

IQ-2) Architecture definition and architectural patterns adopted

- Question: What aspects of your Android do you consider as architectural, and which architectural patterns do you adopt?
- Envisioned output: personal interviewee definition of Android Architecture and architectural patterns adopted (e.g. MVC)
- Goal: the goal of this question is to understand what practitioners intend by Android Architecture. Examples include: (i) relation between android building blocks, (ii) navigation between screens, and (iii) architectural layers. Additionally, we aim at isolating which architectural patterns are adopted in practice by the interviewees.
- RQ Mapping: RQ_1, $RQ_{1.1}$

IQ-3) Good Architectural practices

- Question: What are good architectural practices in Android?
- Envisioned output: Set of architectural principles which are deemed as good development practices by the interviewee. Note that this question does not focus on development processes (e.g. DevOps) but on architectural design principles.
- Goal: Extract architectural practices which are deemed as good to be followed by practitioners.
- RQ Mapping: RQ_2

IQ-4) Bad Architectural practices

- Question: What are bad architectural practices in Android?
- Envisioned output: Set of architectural principles which are deemed as bad development practices by the interviewee. Note that this question does not focus on development processes (e.g. DevOps) but on architectural design principles.
- Goal: Extract architectural practices which are deemed as not good to be followed by practitioners.
- RQ Mapping: RQ_2

IQ-5) Architectural Quality Requirements

- Question: What are your typical concerns w.r.t. quality requirements when considering the architecture of your apps?
- Envisioned output: Set of quality requirements which the interviewee deems as important when architecting Android apps.
- Goal: Understand which QRs are considered as most important when practitioners deal with Android architecture.
- RQ Mapping: RQ_3

The envisioned timeframe of each interview is of approximately 30 minutes, but it can be extended if required.

6 Step 3: Data Processing and Extraction

The main goal of the activity reported in this section is to (i) uniform the data gathered from the heterogeneous data points (i.e. WL, GL, and interviews) and (ii) to extract the data from each data point.

6.1 Data processing

In order to ease the process of data extraction, it will first be required to merge the different data points into a single pool of data. In order to do so, the totality of the data points will be uniformed and stored into a single dataset. Each row of the dataset will represent a data point, while the columns will be used to store data point metadata and the data extracted per data point. For the sake of backwards traceability, each data point will be mapped to a unique identifier, constituted by an acronym of the data point type (WL=White Literature, GL=Gray Literature, I = Interview) followed by an incremental numeric value (e.g. the second extracted WL data point will be mapped to "WL1").

6.2 Data extraction

In our study, the classification framework will be composed of three distinct parts, each of which addresses one of the research questions of our study (see Section 3):

1. *General characteristics of Android apps architecture*, addressing RQ_1, see Section 6.2.1,
2. *Architectural guidelines*, addressing RQ_2, see Section 6.2.2,
3. *Quality requirements*, addressing RQ_3, see Section 6.2.3.

6.2.1 General characteristics of Android apps architecture (RQ_1)

In the following we list all the parameters that will be extracted in order to answer the first research question of our study.

1. *Architectural patterns* ($RQ_{1.1}$): The first data extracted from the data points will be which architectural pattern or style (jointly referred to as "architectural pattern" for the sake of space and consistency) are used in the data point. In case that no pattern is mentioned in a specific data point, the corresponding entry for the data point will be left blank. Note that a pattern will be mapped to a data point exclusively if the data point is discussed in depth in the data point. Data points which simply mention a pattern without specifically focusing on it will not be mapped to that pattern.
2. *Programming libraries* ($RQ_{1.2}$): This field considers the programming libraries which are regarded as architecturally relevant in the data points. This includes both libraries which are used for architectural practices and libraries which are affected by them, e.g. a testing library which highly benefits from or influences architectural practices in Android.

6.2.2 Architectural guidelines (RQ2)

1. *Architectural practices* (RQ_2): In order to answer the second research question, we will need to extract from the data points architectural Android practices. The data to be extracted for this attribute consists of Android-specific architectural practices which can guide the high-level design of apps. As the definition of "Android architecture" may vary according to the data point considered, we will rely on the specific definition provided in the data points, in order to avoid potential threats to construct validity. A data point could contain zero or more architectural practices.

6.2.3 Architecturally significant quality requirements (RQ_3)

This latter attribute has been selected in order to assess which quality requirements [21] are considered while discussing Android architecture.

1. *Quality requirement*: This attribute will be used to identify which quality attributes are discussed when considering Android architecture. More specifically, we will extract the quality attributes referenced in the paper which are regarded to either guide or are influenced by architectural decisions in Android apps. To goal of the analysis of such attribute is a clear overview of which quality attributes are more considered while architecting Android apps, and hence which are more architecturally significant.

7 Step 4: Data synthesis

The data synthesis activity involves collating and summarising the data extracted from the primary studies [26, § 6.5] with the main goal of understanding, analysing, and classifying current state of the art of Android architectural principles.

In this phase we will have a fully populated spreadsheet with all the information coming from the data extraction form of each data point. According to this, our data synthesis will be divided into two main phases: a vertical analysis and a keywording process [27]. When performing *vertical analysis*, we will analyze the extracted data to find trends and collect information about each parameter of each attribute of our data extraction protocol. This strategy will be adopted in order to answer in a quantitative way to both RQ_1 and RQ_3, and in a qualitative way RQ_2. In order to answer RQ_2 we will also adopt a keywording process, in order to first cluster into different groups the extracted practices according to their theme and semantic meaning, and subsequently merge them into guidelines once a sufficient granularity of grouping is reached.

During the vertical analysis we will perform a combination of content analysis (mainly for categorizing and coding the studies under broad thematic categories) and narrative synthesis (mainly for explaining in details and interpreting the findings coming from the content analysis).

Depending on the parameters selected (see Section 6), in this research we will apply quantitative synthesis methods to answer RQ_1 and RQ_3 respectively. When considering quantitative data, depending on the specific data to be analyzed, we will apply descriptive statistics for better understanding the data. When considering qualitative data to answer RQ_2, we will apply the *line of argument* synthesis [17], that is: firstly we will analyse each primary study individually in order to document it and tabulate its main features with respect to each specific parameter of the classification framework, then we will analyze the set of studies as a whole, in order to reason on potential patterns and trends. When both quantitative and qualitative analyses are completed, we will integrate their results in order to explain quantitative results by using qualitative results [26, § 6.5].

In order to carry out a rigorous data extraction process for RQ_2, as well as to ease the control and the subsequent analysis of the extracted data, we will adopt a keywording process.

In order to derive architectural guidelines from practices, we will create a framework we will use to categorize the practices by semantic similarity. The structure of the framework will be composed of the various themes emerging from the practices. For each primary study, the principal researcher and the research methodologist will collect in a spreadsheet a record with the extracted information for subsequent analysis: the spreadsheet columns will be the parameters, while each spreadsheet row will represent the data of each primary study.

As suggested in [17], the principal researcher and the research methodologist will pilot the data extraction form independently. In order to validate our data extraction strategy, we will perform a sensitivity analysis to check whether the results are consistent independently from the researcher performing the analysis.

More specifically, the principal researcher and the research methodologist will get a random sample of 20 primary studies and will analyze them independently by filling the data extraction form for each primary study. Then, they will assess their level of agreement and each disagreement will be discussed and resolved with the intervention of the research methodologist, if needed.

When going through a primary study in detail for extracting information, researchers can agree that the currently analysed study may be semantically out of the scope of this research, and so it can be excluded.

We will follow a systematic process called *keywording* [27] for categorizing the practices into different sub-categories (namely the first level will be the considered theme, e.g. “MVP”, while the following levels will be defined *ad-hoc* guided by the topics covered by the practices). Basically, our keywording process aims at grouping semantically similar practices, till it will be able to merge them into a single guideline. In case that an architectural practice will not be mapped to any other practice, it will constitute a guideline *per se*.

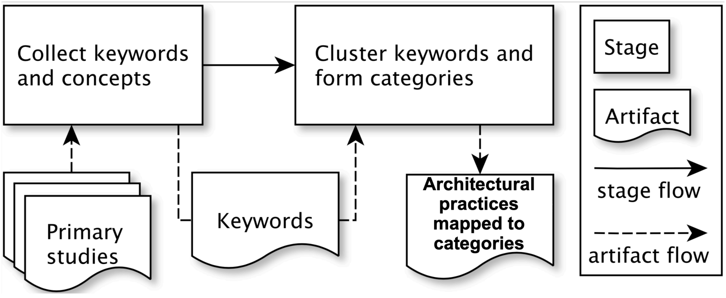

Figure 2: Overview of the keywording process

Figure 2 shows our keywording process in more details. The keywording is done in two steps, which will be repeated in a recursive fashion if required, till a complete grouping of semantically similar architectural practices. The two steps consist of:

1. *Collect keywords and concepts*: the principal researcher and research methodologist will collect keywords and concepts by reading the content of the practices. When all practices will be analysed, all keywords and concepts will be combined together to clearly identify the context, nature, and contribution of the pratice. Bearing in mind that the authors of the practices may use different terms for the same concepts and viceversa (e.g., “clean architecture” vs “clean system principles”), we will collate different keywords and terms to ensure consistency and compatibility. The output of this stage will be the set of keywords extracted from the practices.

2. *Cluster keywords and form categories*: once the keywords and concepts will be finalized, the principal researcher and the research methodologist will perform a clustering operation on them in order to have a set of representative clusters of keywords. The output of this stage will be the a comprehensive mapping of architectural practices to the identified keyword categories (each of them having a specific type and possible values), representing a specific aspect of Android architecture principles.

This process will be (if needed) be recursed on a sub-set of keywords, in case that the content of the practices result still too dissimilar in order to be merged into a single architectural guideline. The keywording process will end when all architectural practices will be support a determinate guideline.

8 Step 5: Reporting

The final step of the mixed-method study consist of reporting. During this final process we will thoroughly document (i) an introduction to the context of this study, (ii) the research method adopted, including the totality of the processes reported in this protocol, (iii) the results of the data extraction and subsequent analysis, (iv) a discussion of the obtained results, (v)the related work, and where this study positions itself w.r.t. the existing literature, (vi) the potential threats to validity which we identified, and (vii) the conclusions we can draw from our research and the future research steps which it will lead to.

References

[1] Ben Martin, Global Digital Future in Focus - 2018 International Edition, comsCore white paper (2018).

[2] Global mobile os market share in sales to end users from 1st quarter 2009 to 1st quarter 2018 (2018).
URL https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

[3] E. D. Corporation, Global Developer Population and Demographic Study 2017 Vol. 1 (2016).
URL https://evansdata.com/press/viewRelease.php?pressID=244

[4] Number of available applications in the Google Play Store from December 2009 to June 2018 (2018).
URL https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/

[5] A. I. Wasserman, Software engineering issues for mobile application development, in: Proceedings of the FSE/SDP workshop on Future of software engineering research, ACM, 2010, pp. 397–400.

[6] H. Bagheri, J. Garcia, A. Sadeghi, S. Malek, N. Medvidovic, Software architectural principles in contemporary mobile software: from conception to practice, Journal of Systems and Software 119 (2016) 31–44.

[7] E. Campos, U. Kulesza, R. Coelho, R. Bonifácio, L. Mariano, Unveiling the architecture and design of android applications, in: Proceedings of the 17th International Conference on Enterprise Information Systems-Volume 2, 2015, pp. 201–211.

[8] K. Sokolova, M. Lemercier, L. Garcia, L. C. Saint Luc, Towards high quality mobile applications: Android passive mvc architecture, International Journal On Advances in Software 7 (2) (2014) 123–138.

[9] T. Dürschmid, M. Trapp, J. Döllner, Towards architectural styles for android app software product lines, in: Proceedings of the 4th International Conference on Mobile Software Engineering and Systems, IEEE Press, 2017, pp. 58–62.

[10] M. E. Joorabchi, A. Mesbah, P. Kruchten, Real challenges in mobile app development, in: Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on, IEEE, 2013, pp. 15–24.

[11] V. Garousi, M. Felderer, M. V. Mäntylä, The need for multivocal literature reviews in software engineering: complementing systematic literature reviews with grey literature, in: Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering, ACM, 2016, p. 26.

[12] V. Garousi, B. Küçük, Smells in software test code: A survey of knowledge in industry and academia, Journal of Systems and Software 138 (2018) 52–81.

[13] N. Tripathi, P. Seppänen, G. Boominathan, M. Oivo, K. Liukkunen, Insights into startup ecosystems through exploration of multi-vocal literature, Information and Software Technology 105 (2019) 56–77.

[14] J. Soldani, D. A. Tamburri, W.-J. Van Den Heuvel, The pains and gains of microservices: A systematic grey literature review, Journal of Systems and Software 146 (2018) 215–232.

[15] S. Maro, J.-P. Steghöfer, M. Staron, Software traceability in the automotive domain: Challenges and solutions, Journal of Systems and Software 141 (2018) 85–110.

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

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

[18] 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, ACM, 2014, p. 38.

[19] H. Zhang, M. A. Babar, Systematic reviews in software engineering: An empirical investigation, Information and Software Technology 55 (7) (2013) 1341–1354.

[20] V. R. Basili, G. Caldiera, H. D. Rombach, The Goal Question Metric Approach, in: Encyclopedia of Software Engineering, Vol. 2, Wiley, 1994, pp. 528–532.

[21] ISO/IEC, 25010 System and software quality models, 2010.

[22] V. Garousi, M. Felderer, M. V. Mäntylä, Guidelines for including grey literature and conducting multivocal literature reviews in software engineering, Information and Software Technology.

[23] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, in: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering, EASE'08, British Computer Society, Swinton, UK, UK, 2008, pp. 68–77.
URL `http://dl.acm.org/citation.cfm?id=2227115.2227123`

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

[25] 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, ACM, New York, NY, USA, 2014, pp. 38:1–38:10.

[26] B. A. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering, Tech. Rep. EBSE-2007-01, Keele University and University of Durham (2007).

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