Development of Mobile Applications with Functions in the Cloud through the Model Driven Approach: A Systematic Mapping Study

Emanuel Sanchiz, Magalí González, Nathalie Aquino, Luca Cernuzzi
Universidad Católica "Nuestra Señora de la Asunción", DEI
Asunción, Paraguay
{*emanuel.sanchiz, mgonzalez, nathalie.aquino, lcernuzz*}*@uc.edu.py*

Abstract

Currently, a growing interest is being caused by mobile applications which have functions implemented in the cloud (MobileApps-FC). Improvements related to the portability of these applications among different platforms and different service providers are a critical need. Model Driven Development (MDD) constitutes one of the alternatives to address portability issues. This work presents a systematic mapping study that analyzes different proposals that apply MDD to the development of MobileApps-FC and that, at the same time, consider the improvement of the portability of such applications. Even though we have identified just a few studies related to our subject of interest, the validation experiences that are presented in them, encourage the adoption of MDD to address portability issues. However, further validation experiences that consider more complex cases in industrial environments will be required to justify the benefits of MDD in a substantial manner.

Keywords: systematic mapping study, mobile applications with functions in the cloud, model driven development, portability.

1 Introduction and Motivation

In recent years, the number of users of smartphones and tablets has been increasing quickly, overcoming the number of users of computers and notebooks[1]. Consequently, the interest for the development of mobile applications has also increased[2], particularly for native and hybrid applications. Moreover, it is important to consider the current growing market of the cloud, specially the public cloud[3], and its role as enhancer of the restricted resources of mobile devices [1, 2]. Taking into account both perspectives, mobile devices and the cloud, it is worth noting the growing convergence between them [3]. Therefore, the interest of this study is focused on mobile applications which have at least one module or implementation (e.g., procedure, service, database) running in the cloud. We refer to these applications as mobile applications with functions in the cloud (MobileApps-FC).

MobileApps-FC are composed of a mobile environment and a cloud environment. Both of them consider different operating systems, programming languages, libraries, services, frameworks, etc. Therefore, they constitute heterogeneous environments where working can be difficult, since usually different languages and tools must be employed.

On one side, the mobile environment can be based on several different platforms, being iOS and Android the most popular ones. These platforms, again, consider different operating systems, programming languages, tools, etc. In most cases, mobile applications must be developed for more than one mobile platform. And due to differences among these platforms, specific application versions must be built for each of them. Therefore, more effort and time must be employed, resulting in higher costs in the development process. Since a same implementation can hardly be used for different mobile platforms, there is a portability problem in the mobile environment [3, 4, 5].

[1]Morgan Stanley, link: https://goo.gl/Ifznvz
[2]Mobile is all about apps, https://goo.gl/tCnZfN
[3]Growing of public clouds, https://goo.gl/WOVYJW

On the other side, the cloud environment is constituted by several different service providers. In a similar way than previously exposed, each of these providers uses its own specific platform (operating systems, programming languages, libraries, etc.). This causes a portability problem, which in the cloud context is known as vendor lock-in. Therefore, tasks such as building an application that is able to run in clouds from different service providers or migrating an application from the cloud of one service provider to another one constitute interesting challenges, since a same implementation can hardly be used in different cloud service providers [6, 7].

Therefore, when considering applications that include, at the same time, the mobile and the cloud environments, the portability problem is highlighted.

Several possible solutions have been proposed to address the portability problem in the mobile context, from mobile web applications to cross-platform compilers [8].

In this work, we focus on native and hybrid mobile applications instead of web applications. Native and hybrid mobile applications present more attractive user interfaces as well as better performance, efficiency and access to hardware. These features are due to their good capabilities in exploiting the resources of a mobile device. Furthermore, the popularity of native and hybrid applications is increasing among users as opposed to the popularity of web applications[4].

It is also worth noting that most of the improvement efforts related to the portability problem in the mobile context have focused on the implementation stage of the development process (for instance, using cross-platform compilers). Nevertheless, it would be beneficial to address the portability problem earlier, at the design stage. These possible improvements at the design stage could contribute to the portability of the resulting implementation.

In this sense, Model Driven Development (MDD) emerges as a possible solution. In fact, one of the main motivations of MDD is related to the improvement of portability. MDD focuses efforts on designing domain models that capture the knowledge of the data and functionality that an application must provide independently of the platform in which it will be run. In this way, one same domain model is valid for any implementation platform. Model Driven Architecture (MDA) goes one step further and defines different abstraction layers for the design and development of applications: i) Computer Independent Model (CIM), which represents an application independently of computer-based concepts; ii) Platform Independent Model (PIM), which represents an application independently of an implementation platform; iii) Platform Specific Model (PSM), which represents an application for a specific platform; and iv) code, which corresponds to the source code of the application. Furthermore, following MDA the development process can start at higher abstraction layers (CIM or PIM) and then subsequent (semi-)automatic transformations can be performed until the source code is reached. In this way, PIM and transformation rules allow the solution logic of an application to be reused for different target platforms [9, 10]. This reuse helps to achieve savings of cost, time and effort in the design and implementation of applications that have to run in different platforms.

Based on a previous work [11], we have identified three aspects that could have a positive impact in the portability problem that is found in the MDD/MDA development of MobileApps-FC. These aspects are: i) incorporation of an Architecture Specific Model (ASM) as a new modeling layer; ii) clear separation of the presentation layer with regard to the navigation and behavior layers; and iii) definition of the navigational structure according to a function-oriented approach. Next section presents our arguments about how and why these aspects could have a positive impact in the portability problem. However, we acknowledge that more studies and research are necessaries to verify this positive impact.

In this work, we present a Systematic Mapping Study (SMS) about the use of MDD for the development of MobileApps-FC. We performed a SMS since this study is characterized for being more general than a Systematic Literature Review (SLR). Giving a global vision about a subject of interest, with empirical evidences or not, is the main goal of a SMS [12]. Moreover, we focused our study to identify and analyze proposals that address portability challenges. Therefore, each selected proposal has been analyzed regarding to the previously presented aspects that could improve portability in the context of the development of MobileApps-FC based on MDD. Additionally, we have analyzed the main contributions, limitations and validation experiences of the proposals.

This work extends another one in which a previous version of this SMS was presented [13]. This work presents a broader explanation of the motivations of this study, an increased list of analyzed papers and a more extensive analysis of results.

Next sections of this paper are organized as follows. Section 2 presents some aspects that could have a positive impact on the portability of MDD applications . Section 3 presents related work. Then, Section 4, Section 5, and Section 6 present the planning, execution and results of the SMS, respectively. In Section 7, we analyze the results. In Section 8, we specify the limitations of our study. Finally, Section 9 presents conclusion.

[4]Popularity of mobile apps, https://goo.gl/fDRZPV

2 Possible Positive Impacts for Portability

As previously commented, based on a previous work [11] we have identified certain aspects that could have a positive impact in the portability problem that is found in the MDD/MDA development of MobileApps-FC. This section presents these aspects arguing about their possible contributions on portability.

- Incorporation of an Architecture Specific Model (ASM) as a new modeling layer. While the portability to different platforms is well resolved by the MDA approach, the constant technological evolution brings also new architectures (e.g., Rich Internet Applications (RIA), mobile architectures). Usually, when MDA development methods are extended to cope with these new architectures, it happens that extensions are included at the PIM level, resulting in an enriched PIM that includes characteristics and constraints that are related to a certain specific architecture [14]. The enriched PIM loses its independence from the architecture and becomes more complex to understand and manage. The consequence is a loss of portability and reusability of models. To resolve this problem, some authors have proposed the introduction of the ASM, for instance, as an intermediate layer between PIM and PSM (e.g., [15, 11]). In this way, one PIM model can result in different ASM models, since one same application can be implemented in different architectures (e.g., a banking application with a web version using RIA and a mobile version based on the mobile architecture). In a similar way, one ASM model can result in different PSM models, since one same architecture can be implemented in different platforms (e.g., the RIA architecture can be implemented in different platforms such as Backbase, Dojo, GWT, jQuery, among others). Therefore, the ASM prevents adding architectural details at the PIM level, improving its portability.

- Clear separation of the presentation layer with regard to the navigation and behavior layers. The presentation layer is the one that presents more difficulties related to portability [4]. There are differences about screen size and resolution; text style (fonts, size, colors); user interface elements (e.g., buttons, search bars, menu); position and visibility; availability of physical buttons (e.g., a physical back button is available in Android and Windows phone, but not in iOS); types of control user interface elements; types of navigation representation; etc. For example, several of the mentioned differences can be seen in the Android and iOS versions of Twitter[5]. Hence, a clear separation of the presentation layer details regarding details of other layers would prevent additional complications to the presentation. This prevention could help the portability of the presentation to be improved, as well as for the other layers.

- Different MDD proposals deal with the modeling and construction of the navigational perspective according to a data-oriented approach (sometimes also called object-oriented) [16, 17, 18, 19]. This means that the navigation is based on data (e.g., classes) and its structure (e.g., relationships among classes). In practice, the content and structure of the menu of an application are the data and a derivation of its structure as it is stored, respectively [20, 21]. Alternatively, according to a function-oriented approach, navigation is based on functions. In this case, the content of the menu are functions (functional requirements) and the structure of the menu depends mainly on user requirements [22, 21, 23]. For instance, an application could not suffer modifications on its functional requirements but its data model could change (e.g., splitting or joining different classes because of a normalization or de-normalization). As a consequence, in the case of the data-oriented approach, such changes could affect the navigation. For example, a function requirement that *registers* users in an application can be implemented by means of a *user* class. This class has different attributes, including, for instance, home address and business address. If we normalize *user* regarding address, we obtain two classes, *user* and *address*. Considering a data-oriented navigation, before the normalization, the registration function involves a single screen. After the normalization, we have to navigate to a new screen to fulfill the addresses. Instead, considering a function oriented approach, independently of the data organization we still have just the same function *register*, and therefore navigation does not require to be modified. Furthermore, regarding platforms in mobile environments, the navigation design is different for each of them[6]. Each platform considers particular navigation patterns and design restrictions [24]. Therefore, it is necessary to adapt the navigation according to the constraints of each platform. Building a navigation structure that makes sense to the user (i.e., it is fast and easy to get to the content) is a common suggestion coming from design guidelines of mobile platforms. They have their own standard navigation elements and patterns. For example, the *tab bar* for iOS, the *side drawer* for Android, the *swipe views* in Windows phone. These kind of standards create habits in users. Therefore, to get an intuitive navigation, those standards have to be considered in the design process of each platform.

[5]Twitter iOS vs Twitter Android, link: https://goo.gl/Ro3qbA
[6]Android: https://goo.gl/kcCuxc; iOS: https://goo.gl/Jox14J; Windows phone: https://goo.gl/Ql3iOH

Table 1: Possible contributions of aspects with regard to portability

Aspect	Possible Contribution
ASM	Portability of the PIM with regard to different architectures
Separation of presentation from navigation and behavior	Portability of the presentation layer
Function oriented navigation	Portability of the modeling of navigation

Considering the registration example, in the data-oriented approach changes in the data model have to be applied to navigation. Since navigation elements and patterns differ according to platforms, it would be necessary to review the design made in each platform taking into account the mentioned changes. Therefore, such analysis and reviews for each platform imply more effort and could make even worse the effects of the portability problem. Nevertheless, considering a function-oriented navigation, the mentioned changes do not affect the navigation, whence there is no additional work to be performed. In this sense, the function-oriented navigation would help to attenuate the effects of the difficulties of portability. Therefore, there is an indirect positive relationship between the navigational function-oriented approach and the portability of applications.

Table 1 summarizes the possible contributions of the previous aspects with regard to portability. Despite the intuition on the positive impacts that these aspects could have, more studies and research are necessary to get a clear assess of them. In that way, this work is intended to offer a step forward.

3 Related Work

As a starting point of this work, we first conducted a search for SMS and SLR related to the development of MobileApps-FC under the MDD approach. Despite not having identified the existence of such work, we found studies that are partially related to our interest. Next, we briefly describe those studies.

Heitkötter et al. [8] present an analysis that focuses on the development of portable mobile applications. They compare cross-platform solutions (mobile web applications: PhoneGap[7] and Titanium Mobile[8]) against mobile native applications, according to two categories of criteria. On one side, criteria related to infrastructure: license and costs, supported platforms, access to platform-specific features, long-term feasibility, look and feel, application speed and distribution. On the other side, criteria related to development: development environment, GUI design, ease of development, maintainability, scalability, opportunities for further development, speed and cost of development. From their perspective, they conclude that the cross-platform approaches present interesting properties like the use of standard and popular programming languages (development side) and the access to a considerable set of platform-specific features (specially from PhoneGap and Titanium Mobile in the infrastructure side). Therefore, they could be considered as a substitute of the native development approach. The solutions analyzed in this work are not model-driven. Furthermore, they are focused on the implementation stage. These are differences with regard to our main interest, which relates to model-driven approaches that deal with portability at the design stage. Nevertheless, they propose alternative perspectives (as the web[9] and hybrid applications[10]) to reduce the impact of the portability problem in the mobile environment.

Ribeiro and da Silva [25] present a set of technologies that focuses on the development of cross-platform mobile applications. The technologies are Rhodes[11], PhoneGap, Dragon RAD, Titanium, mobl[12] and mdsl[13]. The study presents a comparative analysis of the technologies cited above, about the approaches, languages, tools, type of applications generated and device access. Among other conclusions, the study highlights the low consideration of the model driven approach for the development of mobile applications, just mobl and mdsl take into account MDD, and suggest a greater consideration of it, since MDD may represent a good

[7]Link: https://goo.gl/e2gG
[8]Link: https://goo.gl/794vzn
[9]Web mobile applications rely the portability in browsers like Safari or Chrome.
[10]Hybrid applications are web applications (or web pages) in native browsers, such as UIWebView in iOS and WebView in Android (not Safari or Chrome). Hybrid applications are developed using HTML, CSS and Javascript, and then wrapped in a native application using platforms like Cordova.
[11]Rhodes, link: https://goo.gl/Ivrq2f
[12]mobl, link: https://goo.gl/7xh46X
[13]Canappi mdsl, link: https://goo.gl/FWezFE

solution not only to develop cross-platform apps, but also to ease the development and to captivate a larger number of developers for the specific domain (in this case, the mobile domain).

Di Martino et al. [26] describe a set of projects which consider MDD to give support to portability and interoperability in the cloud. The set of projects includes MODAClouds[14], ARTIST[15], REMICS[16] and PaaSage[17]. Furthermore, this work also presents an interesting summary of cloud patterns. Some of these patterns are independent of the provider and consequently, platform independent. The platform independent patterns were extracted from CloudPatterns.org[18], which is a community dedicated to the documentation of a catalog of patterns that includes a set of design solutions for the modularization of the relevant technological solutions for the configuration of clouds [26]. This information is relevant to our area of research since it considers platform independent design aspects that could be used to enrich and to improve the portability of cloud mobile applications developed under the MDD process. In contrast to our work, Di Martino et al. refer to a level of modeling that is not precisely related to the design of an application, but rather refers to configurations and functions of the cloud environment.

Umuhoza and Brambilla [27] performed a survey on MDD approaches for the development of mobile applications. The selected studies were classified according to: i) the covered phases of the development process (requirements analysis, design, implementation, testing); ii) the covered aspects of mobile applications, like application data, user interaction, user interface, and business logic; iii) applied model driven techniques, including type of modeling language, model transformations, code generation or model interpretation; iv) the type of mobile application developed, like native, hybrid, or mobile web; v) the supported mobile platforms, including Android, iOS, or Windows Phone. As results, the survey shows a preference of code generation over model interpretation and of native applications over cross-platform ones. Regarding the relationship with our work, this study was focused exclusively on the development of mobile applications (leaving out implementations in the cloud) and it does not consider the analysis of portability issues.

Despite the contributions of the works that have been presented in this section, it is worth to mention that they have not followed the guidelines of a SLR or SMS. Therefore, since we have not identified a SLR or SMS that focuses on the development of MobileApps-FC under the MDD approach, we were motivated to perform the SMS that is described in the next sections.

4 Planning of the Systematic Mapping Study

We conducted this SMS following the guidelines proposed by Genero et al. [12] and Kitchenham and Charters [28].

The study consisted of the following steps: i) identification of the need for a systematic mapping; ii) formulation of research questions; iii) definition of the search protocol; iv) derivation of the search string and selection criteria; v) searching for primary studies; vi) identification of the data needed to answer the research questions; vii) summary and synthesis of study results; viii) report-writing.

In the Introduction and Section 3 we described the need for this SMS. In this section, we present information related to steps ii), iii) and iv). Steps v) and vi) are presented in Section 5 and Section 6, and the step vii) in Section 7.

4.1 Research Questions

As previously mentioned, this study's goal is to compile and analyze works that follow an MDD approach and that, at the same time, consider the improvement of the portability of MobileApps-FC *(see research question RQ1)*. The aim is to analyze if these works provide, or not, a layer related to the ASM, separation of presentation with regard to behavior and navigation, and function oriented navigation, since we consider these aspects could provide benefits related to the portability problem *(see RQ2, RQ3 and RQ4)*. Furthermore, we consider the identification of other contributions or limitations, specially with regard to portability, to learn about complementary approaches that could improve this aspect *(see RQ5)*. In addition, we analyze the evaluation presented by the selected articles, in order to have an assessment of the practical benefits of the MDD approach regarding the portability problem, and to get a perspective of its possible evolution *(see RQ6)*.

Given these goals and interests, we have defined the next research questions:

- **RQ1**: Which are the model driven proposals for the development of MobileApps-FC that address the portability problem?

[14]MODAClouds project web-site, link: https://goo.gl/lLBnPF
[15]ARTIST project web-site, link: https://goo.gl/6ecBGO
[16]REMICS project web-site, link: https://goo.gl/U7fDxp
[17]PaaSage project web-site, link: https://goo.gl/PPMYgz
[18]Link to the site: https://goo.gl/OsXNFZ

In the identified proposals:

- **RQ2**: Are there evidences of an independent layer for ASM?
- **RQ3**: Are there evidences of a clear separation of the presentation layer regarding the navigation and behavior layers?
- **RQ4**: Are there evidences about the adoption of the function oriented navigation?
- **RQ5**: Are there other contributions or limitations related to portability?
- **RQ6**: Are there validations? If so, are there positive results regarding the portability problem?

4.2 Search Protocol

The search protocol that we followed consists of the next steps:

1. Derive terms from research questions to build an initial search string.
    (a) Use similar and related terms along with acronyms to obtain the final search string.
2. Query information sources with the defined search string. Focus the search on titles, abstracts and keywords. Adapt the search string according to formats an restrictions of searching tools. Information sources to be considered are digital libraries and indexing services, including journals and conferences.
3. Apply a selection procedure on the query results. This procedure consists of applying some selection criteria in three stages:
    (a) First stage: Apply selection criteria taking into account the title of the works identified with the query of information sources (this is, works identified in step 2).
    (b) Second stage: Apply selection criteria taking into account abstracts and keywords of the works selected in step 3(a).
    (c) Third stage: Apply selection criteria taking into account the full text of the works selected in step 3(b). Discard duplicate articles and those which refer to the same work or proposal.

We have not considered any restriction about publication years at the moment of searching. The search was performed in October 2016. The selection procedure was carried out by one of the researchers. In cases of doubts, the other three researchers were consulted in order to achieve agreement.

4.3 Search String

The first research question (RQ1) has been used to derive terms. Then, we used these terms to build the initial search string. This was the case because from that question (RQ1) it is possible to identify the relevant proposals for this study. Then, the initial search string was enriched with acronyms, and similar and related terms. The resulting search string is:

((“model driven” OR “model-driven” OR MDD OR MDE OR MDA OR “model based” OR “model-based”) AND (mobile OR smartphone OR tablet OR Android OR “windows phone” OR iOS) AND (cloud OR “remote server” OR backend))

The search string was adapted according requirements and restrictions of each searching tool. These adaptations are shown in Table 2.

4.4 Selection Criteria

The remaining research questions (RQ2, RQ3, RQ4, RQ5, RQ6) have been used to guide the definition of inclusion and exclusion criteria.

4.4.1 Inclusion Criteria

This SMS includes studies and proposals that:

1. Consider some of the following properties related to portability: inclusion of an additional abstraction level for modeling, like the ASM; clear separation of the presentation layer regarding the navigation and behavior layers; function oriented navigation; or other mechanisms related to manage or improve portability.
2. Present some of the following elements: meta-models, profiles, transformation rules, frameworks, tools.

Table 2: Search string adapted to each considered information source

Information source	Search string
IEEE	((“model driven” OR “model-driven” OR MDD OR MDE OR MDA OR “model based” OR “model-based”) AND (mobile OR smartphone OR tablet OR Android OR “windows phone” OR iOS) AND (cloud OR “remote server” OR backend))
ScienceDirect	(((“model driven” OR “model-driven” OR MDD OR MDE OR MDA OR “model based” OR “model-based”) AND (mobile OR smartphone OR tablet OR Android OR “windows phone” OR iOS) AND (cloud OR “remote server” OR backend))) AND LIMIT-TO(topics, ”model,cloud”)
ACM	((“model driven” OR “model-driven” OR MDD OR MDE OR MDA OR “model based” OR “model-based”) AND (mobile OR smartphone OR tablet OR Android OR “windows phone” OR iOS) AND (cloud OR “remote server” OR backend))
Scopus	((“model driven” OR “model-driven” OR MDD OR MDE OR MDA OR “model based” OR “model-based”) AND (mobile OR smartphone OR tablet OR Android OR “windows phone” OR iOS) AND (cloud OR “remote server” OR backend))
Google Scholar	allintitle: mobile cloud ”model driven” OR MDD OR MDE OR MDA OR ”model based” OR smartphone OR tablet OR remote OR server OR backend OR Android OR ”windows phone” OR iOS
DBLP	(model.driven|MDD|MDE|MDA) (mobile|smartphone|ios|android)

4.4.2 Exclusion criteria

This SMS excludes studies and proposals that:

1. Are not related to software engineering.
2. Do not follow a model driven approach.
3. Are not related to the development of mobile applications.
4. Focus on mobile web applications instead of native or hybrid applications.
5. Do not consider the modeling and generation of functions in the cloud.
6. Do not consider the portability problem.

4.5 Information Sources

In principle, we have chosen information sources in accordance with a list proposed by Kitchenham and Charters [28], including digital libraries and indexing services. Since some libraries of the mentioned list, like Citeseer library and Inspec, did not give results that match our research area, we focused our search on the information sources listed below:

- Digital libraries: IEEE, ACM and Science Direct.
- Indexing services: Scopus, Google Scholar and DBLP.

Nevertheless, the indexing services have identified studies belonging to different digital libraries, even those that we did not consider directly. It is worth mentioning that this work has included Scopus as an additional source of information. Scopus was not used in the previous version of this SMS [13].

5 Execution of the Systematic Mapping Study

Table 3 presents the number of articles which have been obtained when the search protocol was executed. Based on the protocol previously presented, column *Search* of Table 3 presents the number of articles that have been found after applying the search string in each digital source (step 2 of the protocol). Similarly, columns *Criteria on title*, *Criteria on abstract and keywords*, and *Criteria on full text* show the number of articles that have been selected after the execution of the steps in which inclusion and exclusion criteria were applied (steps 3(a), 3(b) and 3(c) of the protocol, respectively). As we can see, finally, six articles were selected to be analyzed in this SMS. Table 4 shows the proposals presented in the six selected articles with their corresponding references.

It is worth mentioning that, at the stage of applying criteria on titles, we included all studies whose titles refers at least to the MDD approach and the mobile environment, since we obtained articles from other research areas, which did not refer to the development of mobile applications. At this stage, we considered all selection criteria, except for exclusion criteria number 5 and 6 since they relate to more particular issues (cloud and portability), which are not necessarily appearing in titles. At the stage of applying selection criteria on abstracts and keywords, we focused on selecting articles that include portability as their main problem, and excluding those which did not fill that criteria. At this stage, we considered all exclusion criteria except for number 5, which refers to cloud. Finally, at the stage of performing the analysis on the full text of the articles, all selection criteria were considered. Some more details about the application of selection criteria are presented in Section 8.

Table 3: Number of articles found in different steps of the search protocol

Digital sources		**Search**	**Criteria on title**	**Criteria on abstract and keywords**	**Criteria on full text**
Digital libraries	IEEE	82	1	1	6
	Science Direct	180	2	2	
	ACM	20	2	2	
Indexing services	Scopus	291	6	4	
	Google Scholar	169	4	3	
	DBLP	58	39	19	
Total of results		800	54	31	6

6 Results

Table 4 presents a summary of the results of our SMS, considering each research question. Results associated to each research question are described next.

- **RQ1**: Which are the model driven proposals for the development of MobileApps-FC that address the portability problem?

  As previously mentioned, in this SMS six studies were identified that fulfill our selection criteria (see Tables 3 and 4). These studies constitute the answer to our first research question. In other words, the model driven proposals for the development of MobileApps-FC that address the portability problem are WebRatio [4], MD^2 [29], SIMON [30] ,MobiCloud [31], and the proposals of Steiner et al. [32], and Ruokonen et al. [33]. All these proposals were analyzed to answer the remaining research questions.

  Regarding WebRatio and MD^2, it is worth mentioning that their corresponding papers do not explicitly mention functions running in the cloud. Nevertheless, they are both able to generate applications with functionalities that can be run in the cloud. In the case of WebRatio, the documentation of their tools mention that it is possible to generate code that will be run in the cloud[19]. In the case of MD^2, it generates a Java EE 6 application as back-end and this kind of application can be easily deployed in

[19]WebRatio Mobile, https://goo.gl/rTDSrU

Table 4: Summary of results by research questions

	Works					
	WebRatio	MD^2	SIMON	MobiCloud	Steiner et al.	Ruokonen et al.
Ref.	[4]	[29]	[30]	[31]	[32]	[33]
Desc.	Framework with graphical modelling language based on IFML-mobile, UML, and BPMN	Textual modeling language based on the MVC schema	Framework with XML as a textual modeling language and middlewares for the adaptation to different plarforms	Textual modeling language based on the MVC schema	Textual modeling language based on the MVC schema	Graphical modeling for UI generation based on web services
RQ1	Yes	Yes	Yes	Yes	Yes	Yes
RQ2	Conceptually no, in practice yes	No	Partially	No	No	No
RQ3	No	Yes	No	Regarding navigation no, regarding behavior yes	There are not enough details about modeling	Regarding, navigation no, regarding behavior yes
RQ4	It is possible to get	It is possible to get	No	No	It is possible to get	No
RQ5	Pros: Open source generation and REST. Cons: Different tools and languages for modeling and generation	Pros: Server portability based on the JAVA Server and REST. Cons: Basic approach for modeling and generation in the cloud	Pros: Reuse of common functions and details of clouds. REST. Cons: Focused in a specific type of application.	Pros: REST. Cons: Basic modeling and generation.	Cons: Basic modeling and generation.	Cons: It considered an old mobile environment.
RQ6	Yes	Yes	Yes	Yes	Yes, but there is a lack of details	No

the cloud[20]. Therefore, both approaches were considered as MobileApps-FC and were included in this work.

Among the selected proposals, we could consider WebRatio as the most complete work in our study area. WebRatio includes modeling capabilities that allow the design of demanding applications for real use cases. Furthermore, it has earned a place in industry. MD^2, based on the Model-View-Controller (MVC) schema, could be considered as the second work regarding modeling and generation scope. MD^2 has been used in experiments in industry. Unlike the other selected works, SIMON, a framework focused on data oriented applications, does not generate code from the models. Instead, it includes runtime modules, which are middlewares that are adapted according to the specifications coming from models. MobiCloud and Steiner et al. are also based on the MVC schema, but they consider more basic use cases for modeling and generating applications. The work of Ruokonen et al. refers to the old Nokia platform. They consider business processes for modeling, and from these processes they generate different user interfaces according to the specific model of the target device. According the model, a device could have particular specifications of display size, positions of user interface elements, functions.

[20]Push Java EE 6 Application to the cloud, https://goo.gl/DpgNOK

Although we have identified more studies which refer to MDD applications for the mobile environment, we have only found these six studies that also include the cloud.

- **RQ2**: Are there evidences of an independent layer for ASM?

  Starting from WebRatio, it defines a way to extent a pre-existent PIM (the IFML[21] language) in order to obtain models oriented to the mobile environment. For the cloud side, other languages like UML and BPMN are used. In practice and from our perspective, WebRatio has an ASM, although they do not formally recognize this ASM as a new independent layer in their development process. Also in SIMON, it is possible to identify some aspects of architecture (specially details of the cloud, like operations on it), which are encapsulated in a runtime module. This module is separated from the PIM and also is still independent of a platform. It is considered as a middleware that abstracts details of the cloud architecture. In this sense, unlike WebRatio, SIMON distinguishes an additional level of abstraction besides the PIM and the PSM. Therefore, we could say that SIMON considers an ASM, although, in the case of MobileApps-FC, we should say that the considered ASM is partial, because it considers only the cloud side, and there is no middleware (independent from platform) for the mobile architecture.

  In Ruokonen et al., from our perspective, they have a PIM, composed by the business processes and the user interface model, and a PSM where they define the specific characteristics of the target devices. Therefore, they do not consider an additional level of modeling as the ASM.

  In the remaining works, they only talk about domain specific languages, specifically defined for modeling and generating MobileApps-FC. Hence, they consider a PIM, and from it, they generate platform specific implementations.

- **RQ3**: Are there evidences of a clear separation of the presentation layer regarding the navigation and behavior layers?

  We identified this separation in MD^2, where navigation and behavior are modeled in the controller module while presentation (e.g., graphic elements, styles) is isolated in the view module (schema MVC). In MobiCloud and in Ruokonen et al. there is no mention about navigation modeling, hence the separation of the details of these layers is not clear. Regarding the behavior layer we could assume, as a first judgment, that they consider a clear separation since MobiCloud models it inside the controller and Ruokonen et al. through business processes which are isolated from user interface modeling. Also, in Steiner et al. there is not a clear specification of the presentation layer design, neither of the behavior nor navigation ones, whereby the separation is not clear. Nevertheless, in WebRatio a mix of details of the presentation layer regarding the navigation and behavior modeling can be identified, since graphical elements, navigation, events, and service callings are specified in the same diagram. This mixture could be reducing the portability of the mentioned layers. In SIMON, there are three models: the user interface configuration model (where presentation is defined), the data model, and the security policy model. Nevertheless, there is no specification about the navigation modeling, and furthermore there are not enough details about the behavior modeling. According to the article, in principle, we could say that behavior is divided between the user interface configuration and security policy models. Hence, we cannot ensure that there is a clear separation between the layers.

- **RQ4**: Are there evidences about the adoption of the function oriented navigation?

  In WebRatio, MD^2 and Steiner et al. we identified particular models to design navigation. Whereby, it is possible to obtain a function oriented navigation, but it depends on the designers decision because the mentioned proposals do not impose a specific type of navigation. In WebRatio, transitions among different elements of the user interfaces can be defined by means of the IFML-mobile language, as well as transitions among different screens that conform the application structure. Similarly, in MD^2 navigation modeling is carried out in the controller module (schema MVC) and the workflow concept makes it possible to model different types of transitions. In Steiner et al. navigation can be modeled but it is limited to a subset of patterns that are available on both mobile platforms considered for generation, iOS and Android. In this case, we can get function oriented navigation but with limitations. In SIMON, MobiCloud and Ruokonen et al. there is no consideration of navigation modeling, whence there is no guarantee to get a function oriented design.

- **RQ5**: Are there other contributions or limitations related to portability?

  Since we have only considered proposals based on MDD for the development of MobileApps-FC, the six selected works aim to abstract the developer from specific implementation details of the mobile and

[21]IFML, link: http://goo.gl/mWdPpo

cloud environments, through the PIM, allowing the generation of the final applications for different platforms from the same model. This implies possible savings in effort, time and cost, considering the portability problem.

Besides other aspects, which we have verified through the previous three research questions, we did an analysis in order to identify other strategies which complement the use of MDD for the improvement of the portability of applications.

WebRatio considers the generation of mobile code based on Apache Cordova[22], also known as an hybrid approach or native wrapper, and the generation of standard Java code for the cloud, relying its portability on the Java server. In general, we could say that WebRatio chooses for a complement between MDD and the generation of open source code, which is considered as an approach that gives more flexibility for addressing the vendor lock-in problem[23]. All of this can be considered as a suitable combination to face the difficulties originated by the portability problem [4, 29]. Furthermore, WebRatio provides the automatic generation of a smart method for synchronization, which is portable thanks to the PIM and transformation rules. At the same time, this property could be relevant for saving resources of mobile devices at the moment of network communication. Nevertheless, for the design of MobileApps-FC, WebRatio uses different modeling languages, with different modeling tools. This could mean, in some sense, a limitation regarding the abstraction about the differences between the mobile and cloud environments, and consequently, derives in the need of learning to work with different tools (we could have a harder learning curve).

Regarding SIMON, the modularization of architectural aspects of the cloud and the reuse of them through different providers, can be mentioned as contributions related to portability. Another contribution is related to the level of abstraction, since there is a unique modeling language, XML. About limitations, SIMON focuses only on data oriented applications.

MD^2, MobiCloud and Steiner et al. use textual domain specific languages, based on the MVC design pattern. Then, from the same language, they generate implementations for both, the mobile and cloud environments. They manage only one language and one tool for modeling and generation. This could be considered as an advantage regarding the learning curve, comparing with WebRatio. These three works consider the generation of native code for the mobile side. Generally, this kind of code is the most preferred because it presents better possibilities for a suitable use of resources of mobile devices. This means that applications can be obtained with better performance and presentation. As well as WebRatio in the cloud side, MD^2 considers the generation of Java applications, relying the portability of the implementation on the Java server.

MobiCloud generates specific implementations for two proprietary cloud providers, Amazon Elastic Compute Cloud (Amazon EC2) and Google App Engine (GAE). Regarding disadvantages, the modeling and generation approach in MobiCloud is restricted to basic CRUD operations and to the generation of communication between the mobile and cloud environments. Likewise, Steiner et al. consider the generation for GAE, but in this case it was the only one taken into account.

From all proposals, we identified REST as the most adopted communication architecture. This uniformity in communication interfaces becomes a positive property for the improvement of the portability of the communication between the mobile and the cloud, and at the same time for the different platforms belonging to these two environments respectively [34, 35].

About Ruokonen et al., they consider the user interface generation for devices of different physical characteristics (e.g., according to model, screen size). They have identified common and repetitive patterns in the context of business process models. About disadvantages, they generate non-native code for user interfaces. Besides, according to their work context (Nokia Research Center, 2008), we could say that they focused in a specific type of environment, so they did not take into account current mobile platforms.

In general, as a limitation we can mention that most of the studies evidence early research stages. In some cases, the modeling and generation in the cloud side is minimal.

- **RQ6**: Are there validations? If so, are there positive results regarding the portability problem?

  Five of the six selected works included validations. All of them presented positive results.

  The most applied evaluation method consisted in comparing the number of lines of textual models against the number of lines of code generated from those models. This kind of evaluation was performed

[22]Apache Cordova, link: https://goo.gl/QocB2x
[23]More information in: https://goo.gl/mOCr8P

by proposals that rely on textual modeling languages: MobiCloud, MD^2 and Steiner et al. These studies relate the number of lines of code with the effort, time and cost that would take the manual implementation, following the traditional development approach. In all cases, differences in at least one magnitude order were found regarding the number of lines. Specifically, the number of lines of textual models was smaller than the number of lines of code of final implementations. From that, it can be derived that the effort, time and cost to obtain a MobileApps-FC for more than one platform, using an MDD approach, are lower than in the case of the traditional development.

Regarding SIMON, the considered validation method was the comparison of lines of code, again. Unlike the evaluations described above, this comparison was made between the traditional development and the use of SIMON. An application for gathering data was developed. The considered target platforms were Google App Engine (GAE) and Microsoft Azure, as cloud providers, and Android and Windows Phone, as mobile platforms. First, the application was developed for GAE and Android. In this case, SIMON already had the native mobile runtime engine and the cloud adapter (which are middlewares that encapsulate the specific details of a particular platform. For each mobile platform, it is necessary to build its corresponding mobile runtime engine. The same case for the cloud side, for each provider/platform, a cloud adapter must be developed). Nevertheless, when they developed the version for Azure and Windows Phone, they had to developed an specific native mobile runtime engine and cloud adapter. The results of the evaluation were the following:

– GAE and Android: For the traditional development the lines of code were 6224 and for SIMON were 280. A 95% of lines of code decreasing.
– Azure and Windows phone: For the traditional development the lines of code was 6324 and for SIMON were 4014. A 37% of lines of code decreasing.

Regarding WebRatio, they compared their MDD proposal against the traditional development. Specially, they considered the mobile side. Three teams, with two members each, developed an application according to the following description:

– Team 1: They worked in the server side, implementing REST services.
– Team 2: They worked in the client side, adopting a manual development approach (traditional approach).
– Team 3: They also worked in the client side, but adopting the MDD approach proposed by WebRatio.

Teams 2 and 3 developed the front-end following the Apache Cordova cross-platform approach. All teams were tested during 2 weeks. The following results were obtained:

– 9 person-days were employed for the server side development, using the MDD approach through the WebRatio Framework.
– 21 person-days were employed for the manual client side development: i) graphical style: 2 person-days; ii) structure of the application and user interaction: 12 person-days; iii) client-server interaction and push notifications: 7 person-days.
– 11 person-days were employed for the MDD development of the client side: i) graphical style: 1 person-day; ii) structure of the application and user interaction: 7 person-days; iii) client-server interaction and push notifications: 3 person-days.

From the data described, the following benefits were derived: 48% of effort saving, 21% of cost reduction, time reduction from 21 person-days to 11 person-days. In all the cases, the numbers favor the MDD approach proposed by Brambilla et al. [4] (WebRatio).

Ruokonen et al. do not present a validation experience regarding their implementation. Instead, they pose some issues related to design patterns, from the mobile environment, which can be automated through MDD.

Finally, from all these validations and considering the portability problem, we can say that their results are positives and consequently, encouraging.

7 Analysis of Results

Table 5 presents a summary of the results regarding the aspects that we considered for the improvement of portability: inclusion of an ASM, separation of presentation with regard to navigation and behavior, and function oriented navigation.

From the perspective of these aspects, it is possible to model a function oriented navigation in three proposals: WebRatio, MD^2 and Steiner et al. Hence, this aspect is the most feasible one to be adopted by most of the selected works. From the perspective of the works, WebRatio and MD^2 are closer to meet the studied aspects than the others. Both of them are able to adopt the function oriented navigation. MD^2 considers a clear isolation of the presentation layer and, in practice, WebRatio has an ASM. In general, the separation of details between the presentation and behavior layers, and the ASM are the least considered aspects.

From this situation, we can propose studies about either the enrichment of the selected proposals, specially with the two least considered aspects, according to their current status, or new proposals that include these aspects in a native way. First, we further analyze these two alternatives and suggest possibles ways to follow. Then, we propose other possible ways and complementary issues about future researches.

Table 5: Summary of Works vs Aspects

	Works					
Aspects	WebRatio	MD^2	SIMON	MobiCloud	Steiner et al.	Ruokonen et al.
ASM	Conceptually no, in practice yes	No	Partially	No	No	No
Clear separation of the presentation layer regarding navigation and behavior layers	No	Yes	No	No	No	No
Function Oriented Navigation	It is possible to get	It is possible to get	No	No	It is possible to get	No

7.1 Enrichment of the Selected Proposals

7.1.1 WebRatio

WebRatio presents a navigation model independent of the data modeling, so we deduce that it is possible to obtain a function oriented navigation. About the clear separation of the presentation layer regarding the navigation and behavior layers, we identified a mixture of details because in the same diagram they model user interface elements, transitions among these elements, events and calling of services. Therefore, it would be interesting an study that compare or analyze in a deeper way the actual modeling of WebRatio against a modeling where these details are specified in different diagrams, and see if this separation contributes to the portability of the design, or at least, present other benefits. About the ASM, WebRatio do not consider this aspect in a conceptual or formal way. Nevertheless, they did extensions from an existing PIM, so in practice and from our perspective, they have ASM. It would be interesting if some experiments are made to build model to model transformation rules for different architectures, for example for the RIA architecture and for the mobile architecture, in order to verify the portability of their PIM.

7.1.2 MD^2

MD^2 includes navigation modeling, so it is possible to obtain function oriented navigation. It also includes the separation of the details of the presentation layer regarding the navigation and behavior layers, through a clear encapsulation of these details in the view and controller modules respectively (considering the MVC schema). MD^2 does not consider an ASM, hence it would be interesting to perform experiments about the obtainment of the PIM, which could be reused to generate the MobileApps-FC architecture and other architectures (e.g., RIA), trough its textual model language.

7.1.3 SIMON

Compared with the other proposals, SIMON has a particular approach. It considers XML as a textual modeling language, to specify as a Blueprint Engine (BE) three design aspects: data model, security policy and user interface configuration. Besides the blueprint, SIMON includes two additional modules, the Data Runtime Engine (DRE) and the Security Runtime Engine (SRE), which according to the blueprint performs, respectively, data-oriented operations in specific clouds (e.g., interoperation data, inserts, updates, etc.) and security validations between the user and the application (e.g., authentication, validation of privileges, etc.). The BE, DRE, and SRE are the core components and they are independent of platform. Attached to DRE, there is an adapter for each cloud provider. The adapter is a middleware, between the DRE and the cloud provider, that encapsulates specific details of a cloud platform. On the mobile side, it is considered a Native Mobile Runtime Engine (NMRE). It is also a middleware, between the core components and the mobile platform, composed of prefabricated components that are responsible for presenting applications functions to the user by using necessary services provided by BE, DRE and SRE. As we saw, SIMON does not consider the three aspects studied here. SIMON does not present navigation modeling, so it would be interesting that it includes the mentioned modeling and suggests a function oriented approach. Also, some experiments could be perform to clearly separate the modeling of the presentation and the behavior layers, in order to see if this separation contributes to a greater reuse of the existing NMRE and cloud adapters, to build new implementations according to other specific platforms. Partially, SIMON considers an ASM that is encapsulated in the DRE and SRE modules. But, these modules are focused on the cloud side. In this sense, they could consider to build this kind of modules on the mobile side.

7.1.4 MobiCloud, Steiner et al. and Ruokonen et al.

From these studies, only Steiner et al. consider one of the aspects that we have studied. We have not identified that MobiCloud and Ruokonen et al. consider at least one of the mentioned aspects. Steiner et al. include a navigation modeling that gives the possibility to obtain a function oriented navigation. But, this modeling is limited to a subset of patterns (of navigation) that are present in two specific mobile platforms that they consider for the generation of code. These studies evidence an early stage about MDD for the development of MobileApps-FC. The cases of study considered by these three last works implies basics CRUD operations on a simplified data model, or some application dimensions are not considered for the modeling (like the navigation modeling), and the user interface is restricted to basic elements (like labels, buttons and entry boxes) [31, 32, 33].

7.1.5 Other Considerations

It should be noted that despite the mentioned limitations, all these works present initial evidences that are positive for the adoption of MDD for the development of MobileApps-FC. Therefore, they should be considered as reference points for the continuation of the research. More complex cases of study could be considered, in order to identify other patterns, repetitive processes, and functions which can be incorporated in the MDD approach. Even MD^2, would need to consider more complex cases of study to verify the applicability and feasibility of their proposal in industry cases.

Further and besides the three aspects studied here, these studies could be enriched by a unification of their research or at least they could benefit from learning from their respective research. Specially, those which consider a textual modeling language: MD^2, MobiCloud and Steiner et al. Also these works are based on the MVC schema. At first glance, they consider a quite similar process and approach.

Once there are more evidences, it would be possible to get more information about the effects of the three studied aspects regarding the portability and, also, about the applicability of MDD for the development of MobileApps-FC.

7.2 New Proposal Including the Desirable Aspects

A new proposal that includes in a native way the three main aspects considered here could be proposed. This proposal could be a methodology considering a complete modeling and generation process, complemented by its development environment, specifically its modeling and transformation tools, all of them based on standards like UML and MDA. Considering a general view, we could divide the development process in three phases:

- Problem modeling.
- Solution modeling.
- Implementation or source code generation.

About the problem modeling, we highlight here the PIM specification. This PIM could be based on five models, offering a strong separation of concerns that is in line with some of the three aspects studied in this work:

- A navigation model, including particular diagrams to model the transitions among the different parts of an application, enabling the possibility to obtain a function-oriented navigation, independent of the data model.
- A domain model for the data layer modeling, where it is possible to model the structure and representation of data.
- A presentation model, where all the user interface design and its details are encapsulated.
- A logic model for behavior design, isolated from the presentation layer details.
- A user model for defining roles and assign specific functions to them.

About the solution modeling, we highlight the inclusion of the ASM as an additional level of models, and as a previous stage of the PSM, or directly, as a previous stage of the specific implementation, the application generation. Therefore, from the PIM and through model to model transformation rules, this new proposal should generate, with manual adjustments if it is necessary, an ASM that incorporates the concepts, patterns and functions coming from the emerging architectures.

About the implementation phase, either from the ASM or PSM, it should be possible the generation of code and other artifacts (e.g., documentation) that constitute the generation of a working application.

This new proposal could also take profit of the proposals studied in this work. The new proposal could be enriched by:

- The use of standard modeling languages. For instance, WebRatio [4] uses UML graphic models while SIMON [30] uses XML for textual models.
- The support for modeling and implementation of settings related to security policies in the cloud, that can be derived from SIMON [30].
- The adoption of the MVC schema, that can be derived from MobiCloud [31], MD^2 [29] and Steiner et al. [32].
- The combination of MDD and the open source approach, that can be derived from WebRatio [4].
- The use of REST as a uniform and portable communication interface, that can be derived from WebRatio [4], MD2 [29], SIMON [30], MobiCloud [31].

7.3 Complementary Issues

First, as a complement of modeling and generation in the cloud, there are particular configurations and functions that are peculiar of the clouds. In Di Martino et al.[26], those configurations and functions are presented as patterns. In their analysis, they summarize patterns independently of providers and platforms. Some of these patterns are about sharing, scaling and elasticity patterns, data management and storage, network security, identity and access management. As we can see, they are not related to the design of applications specifically, but they are closely related and they contribute to alleviate the vendor lock-in problem [26]. This could be an interesting option to enrich the modeling of applications since, it should contribute to improve the portability of the models.

Second, considering the clear separation of the presentation layer regarding the navigation and behavior layers, the principle of Separation of Concerns can be identified, which facilitates the building, understanding and expressiveness of model languages. In this sense, as future work, it would be interesting to perform an study about the separation of concerns in the specification of mobile applications, since these languages that apply the mentioned principle could be used in the MDD process and, at the same time, it could enrich that process.

Third, it would be interesting to confront and to compare different proposals, that combine MDD with other methodologies or approaches, regarding the qualities of portability and efficiency. This last one, is also relevant, specially for the mobile side, because the restrictions of resources like the battery. For instance, these possible studies could compare the combination of MDD with open source/independent platform generation (e.g., Apache Cordova, hybrid applications) against MDD with a proprietary and/or native generation. The open approach is considered as a more flexible way to improve the portability, while the native or proprietary approach is considered better to get more efficiency. It would be interesting to know which combination presents the better trade-off between the mentioned qualities. In the same line, keeping the analysis regarding portability and efficiency, it would be interesting an study that compares the implementation of the REST interface communication, which is uniform and portable, against a native interface communication, which is better about efficiency.

Fourth, about the development environment. MD^2 presents a unified modeling language and tool to model and generate MobileApps-FC. It could be considered as an advantage comparing MD^2 against WebRatio, since WebRatio presents different languages and tools to model and generate the mentioned applications. It would be interesting to analyze or perform experiments to see if there are significant differences between these both alternatives or which of them present the better properties. For such a study, the learning curve and the tools and language power could be considered as parameters for analysis.

As summary, considering the portability problem of the MobileApps-FC from the analyzed works, we have found that there is a low consideration of the aspects of interest that could improve the portability. Nevertheless, we have identified initial positive evidences supporting MDD for the development of the mentioned applications, regarding the portability issue. These positive evidences encourage future research to get more information and even more evidences in order to enable more general conclusions about the benefits of MDD on the portability problem, at least for the kind of applications that we consider here. Future works could adopt our aspects of interest. This adoption could be done either proposing new approaches or enriching the selected ones. Some other complementary research lines were proposed considering some principles and qualities of the software engineering and which are somewhat related to our main subject.

8 Threats to Validity

This section describes some threats that must be improved in future replications of this study. We consulted experienced researchers to validate the research design and its understanding. Their feedback and the trials contributed to reduce threats.

8.1 Threat of Missing Literature

Since in [28] it is mentioned that in some cases a simple search string could be as effective as a complex one, we tried to use a search string that is not very complex. Our search string included frequently used terms related to our areas of interest, according to our experience and to what we saw in the literature. Nevertheless, in future studies, in order to try to get more results from the sources, our search string could be enriched with additional terms, synonyms, abbreviations (e.g., "platform independent model", "platform agnostic model", "mobile app", "mobile device", services, server). In this study, we could not use the same string for all search engines because some of the tools didn't give us any results. Therefore, we slightly modified our search string according to the search facilities of the different information sources. The string suffered changes specially in DBLP (as it is shown in Table 3). This search tool is not as flexible as the other ones. Despite of such limitations, we opted to keep DBLP because, in general, it gives interesting results of our area of study. Another threat of missing literature is related to the fact that we have defined many exclusion criteria that could have rid off studies about mobile applications that do not consider the cloud but could have been interesting to analyze.

8.2 Threat of Selection Bias

As we have explained previously, during the study selection procedure, we gradually applied the selection criteria according to the different stages. In the first stage, we considered all the selection criteria, except the exclusion criteria number 5 and 6 for being more particular issues. These criteria refer to cloud and portability. Since we cannot obtain enough information from title, we assumed that, at least, the articles in which we were interested, should mention the MDD approach and the mobile environment. In the stage of applying the selection criteria on abstracts and keywords, we focused on selecting the articles which include the portability as the main problem and exclude those which do not filled that criteria. In this stage, we considered all the selection criteria except the number 5, which refers to the cloud. We did things in that way because we were searching for works that focus on the portability problem. We assumed that in the abstracts and keywords should be a mention of the problem. Finally, in the stage of doing the analysis on the full text of the articles, all the selection criteria were considered. Additionally, we mention that just one researcher did the search and selection of articles. Hence, we did not do a parallel search and selection which could have helped the validation of such processes.

8.3 Threat of Inaccuracy of Data Extraction

About the data extraction and summary, these stages were not planned in advance. Nevertheless, they were performed in a structured way, gathering all the necessary information to answer the research questions. Finally, we have not made an evaluation about the quality of the selected works. In Genero et al. [12], they explained that the inclusion of this point is not essential, since, in a SMS both theoretical studies as well as empirical ones can be included, and therefore, it can be difficult to define a common evaluation mechanism for all the included works.

9 Conclusion

This study presents a SMS of MDD proposals for the development of MobileApps-FC. The process has resulted in the identification of a small number of proposals that deal with the subject of interest. This suggest that the area should progress further. At the same time, the analyzed proposals agree in pointing some possible positive aspects about the use of MDD regarding the portability issues in the context of the development of MobileApps-FC. These evidences are still early and they need to be consolidated through further studies. In this sense, the analysis of this SMS have identified possible research lines which include the proposal of new methodologies or improving existing ones to incorporate the desirable analyzed aspects. Moreover, the discipline needs to increment the empirical evidences in order to generalize the consideration about the usefulness of MDD regarding the portability issues for the development of MobileApps-FC.

Acknowledgment

This work has been funded by CONACYT through the PROCIENCIA program with resources from “Fondo para la Excelencia de la Educación e Investigación - FEEI” from FONACIDE. This work has been developed under the project “Mejorando el proceso de desarrollo de software: propuesta basada en MDD” (14-INV-056).

References

[1] V. March, Y. Gu, E. Leonardi, G. Goh, M. Kirchberg, and B. S. Lee, “Mcloud: Towards a new paradigm of rich mobile applications,” *Procedia CS*, vol. 5, pp. 618–624, 2011.

[2] D. Sahu, S. Sharma, V. Dubey, and A. Tripathi, “Cloud computing in mobile applications,” *International Journal of Scientific and Research Publications*, vol. 2, no. 8, pp. 1–9, 2012.

[3] S. Abolfazli, Z. Sanaei, A. Gani, F. Xia, and L. T. Yang, “Rich mobile applications: genesis, taxonomy, and open issues,” *Journal of Network and Computer Applications*, vol. 40, pp. 345–362, 2014.

[4] M. Brambilla, A. Mauri, and E. Umuhoza, “Extending the interaction flow modeling language (IFML) for model driven development of mobile applications front end,” in *Mobile Web Information Systems - 11th International Conference, MobiWIS 2014, Barcelona, Spain, August 27-29, 2014. Proceedings.* Springer International Publishing, 2014, pp. 176–191.

[5] Z. Sanaei, S. Abolfazli, A. Gani, and R. H. Khokhar, “Tripod of requirements in horizontal heterogeneous mobile cloud computing,” *CoRR*, 2012. [Online]. Available: http://arxiv.org/abs/1205.3247

[6] P. Gupta and S. Gupta, “Mobile cloud computing: The future of cloud,” *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 1, no. 3, pp. 134–145, 2012.

[7] E. A. N. da Silva, R. P. Fortes, and D. Lucrédio, “A model-driven approach for promoting cloud paas portability.” in *CASCON*, 2013, pp. 92–105.

[8] H. Heitkötter, S. Hanschke, and T. A. Majchrzak, “Evaluating cross-platform development approaches for mobile applications,” in *Web information systems and technologies.* Springer, 2013, pp. 120–138.

[9] C. Pons, R. Giandini, and G. Pérez, *Desarrollo de Software Dirigido por Modelos*, E. de la Universidad Nacional de La Plata, Ed. Editorial de la Universidad Nacional de La Plata (EDULP)/McGraw-Hill Educación, 2010.

[10] M. Brambilla, J. Cabot, and M. Wimmer, *Model-Driven Software Engineering in Practice*, M. &. Claypool, Ed. Morgan & Claypool, 2012.

[11] M. González, L. Cernuzzi, and O. Pastor, “A navigational role-centric model oriented web approach - moweba,” *Int. J. Web Eng. Technol.*, vol. 11, no. 1, pp. 29–67, 2016. [Online]. Available: http://dx.doi.org/10.1504/IJWET.2016.075963

[12] M. Genero, J. Cruz-Lemus, and M. Piattini, *Métodos de Investigación en Ingeniería del Software*, R.-M. Editorial, Ed. RA-MA, 2014.

[13] E. Sanchiz, M. González, N. Aquino, and L. Cernuzzi, "Mobile cloud applications development through the model driven approach: A systematic mapping study," in *Latin American Computing Conference, CLEI 2016.* IEEE, oct 2016, pp. 605–614.

[14] M. González, L. Cernuzzi, N. Aquino, and O. Pastor, "Developing web applications for different architectures: The moweba approach," in *Tenth IEEE International Conference on Research Challenges in Information Science, RCIS 2016, Grenoble, France, June 1-3, 2016*, 2016, pp. 1–11. [Online]. Available: http://dx.doi.org/10.1109/RCIS.2016.7549344

[15] T. Mikkonen, R. Pitkänen, and M. Pussinen, *On the Role of Architectural Style in Model Driven Development.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 74–87.

[16] N. Koch, A. Knapp, G. Zhang, and H. Baumeister, "Uml-based web engineering," in *Web Engineering: Modelling and Implementing Web Applications.* Springer, 2008, pp. 157–191.

[17] S. Ceri, P. Fraternali, and A. Bongio, "Web modeling language (webml): a modeling language for designing web sites," *Computer Networks*, vol. 33, no. 1, pp. 137–157, 2000.

[18] J. Fons, V. Pelechano, O. Pastor, P. Valderas, and V. Torres, "Applying the oows model-driven approach for developing web applications. the internet movie database case study," in *Web Engineering: Modelling and Implementing Web Applications.* Springer, 2008, pp. 65–108.

[19] C. Cachero, J. Gómez, and O. Pastor, "Ooh-method: un método de diseño de lugares web," in *Conference Proceedings, IDEAS 00. Cancún*, 2000, pp. 133–144.

[20] A. Adamko, "Modeling data-oriented web applications using uml," in *EUROCON 2005-The International Conference on" Computer as a Tool"*, vol. 1. IEEE, 2005, pp. 752–755.

[21] M. Winckler and V. Jean, "Vanderdonckt: Towards a user-centered design of web applications based on a task model," in *In Proceedings of 5th International Workshop on Web-Oriented Software Technologies (IWWOST.* Citeseer, 2005.

[22] P. A. Laplante, *Dictionary of computer science, engineering and technology.* CRC Press, 2000.

[23] S. Howard, J. Hammond, and G. Lindgaard, *Human-Computer Interaction: INTERACT 97.* Springer, 2013.

[24] T. Neil, *Mobile Design Pattern Gallery: UI Patterns for Smartphone Apps*, 2nd ed., I. O'Reilly Media, Ed. O'Reilly Media, Inc., 2014.

[25] A. Ribeiro and A. R. da Silva, "Survey on cross-platforms and languages for mobile apps," in *Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the.* IEEE, 2012, pp. 255–260.

[26] B. Di Martino, G. Cretella, and A. Esposito, "Methodologies for cloud portability and interoperability," in *Cloud Portability and Interoperability.* Springer, 2015, pp. 15–44.

[27] E. Umuhoza and M. Brambilla, "Model driven development approaches for mobile applications: A survey," in *International Conference on Mobile Web and Information Systems.* Springer, 2016, pp. 93–107.

[28] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Keele University and University of Durham, Tech. Rep., 2007, version 2.3.

[29] H. Heitkötter, T. A. Majchrzak, and H. Kuchen, "Cross-platform model-driven development of mobile applications with md 2," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing.* ACM, 2013, pp. 526–533.

[30] N. Chondamrongkul and N. Chondamrongkul, "Model-driven framework to support evolution of mobile applications in multi-cloud environments," *International Journal of Pervasive Computing and Communications*, vol. 12, no. 3, pp. 332–351, 2016.

[31] A. H. Ranabahu, E. M. Maximilien, A. P. Sheth, and K. Thirunarayan, "A domain specific language for enterprise grade cloud-mobile hybrid applications," in *Proceedings of the compilation of the co-located workshops on DSM'11, TMC'11, AGERE! 2011, AOOPES'11, NEAT'11, & VMIL'11.* ACM, 2011, pp. 77–84.

[32] D. Steiner, C. Turlea, C. Culea, and S. Selinger, "Model-driven development of cloud-connected mobile applications using dsls with xtext," in *EUROCAST (2)*, ser. Lecture Notes in Computer Science, vol. 8112. Springer, 2013, pp. 409–416.

[33] A. Ruokonen, L. Pajunen, and T. Systä, "On model-driven development of mobile business processes," in *SERA.* IEEE Computer Society, 2008, pp. 59–66.

[34] R. T. Fielding and R. N. Taylor, "Principled design of the modern web architecture," *ACM Transactions on Internet Technology (TOIT)*, vol. 2, no. 2, pp. 115–150, 2002.

[35] L. Richardson, M. Amundsen, and S. Ruby, *RESTful Web APIs.* "O'Reilly Media, Inc.", 2013.