New approach for smart composition of mobile applications

Afrah Djeddar
LAMIS Laboratory, University of Tebessa, Algeria
afrah-djeddar@hotmail.fr

Hakim Bendjenna
LAMIS Laboratory, University of Tebessa, Algeria
hbendjenna@yahoo.fr

Abdelkrim Amirat
LIM Laboratory, University of Souk Ahras, Algeria
abdelkrim.amirat@yahoo.com

Mourad Oussalah
Lina Laboratory, University of Nantes, France
mourad.oussalah@univ-nantes.fr

Abstract— In the mobile environment, building applications submits to several constraints (i.e. the limited resources offered by the mobile devices and their execution state variability) which prevents their porting to heterogeneous devices available on the market. To deal with this challenge, we present a generic approach to compose mobile applications, using existing heterogeneous software entities, which can be adaptive to several heterogeneous devices mobile. It enables the smart and the dynamic composition of software entities independently from theirs implementation platforms. It also takes into account the specific features of the mobile device in which the desired application will be installed during the composition. This approach aims to provide mobile applications with context-aware contents adapted to the user's requirements.

Keywords—Composition, Reusing software entities, Heterogeneity, Mobile devices constraints.

I. Introduction

Mobile applications usage and development is experiencing exponential growth. According to Gartner, by 2016 more than 300 billion applications will be downloaded annually [1]. Despite the large number of available mobile applications, user's need in the daily-life differ from each other. Thus, mobile devices are characterized by various setting (i.e. software and hardware characteristics) and mobile platforms are rapidly changing, including diverse capabilities as GPS, sensors, and input modes [2].

However, the mobile domain presents several challenges to software engineering [1]. The need to provide mobile applications according to the user's requirements and to support them in a range of mobile devices available in the market (e.g. cell phones, smartphones, tablets, etc.) is one of these challenges. Thereby, mobile applications need to be able to specialize their behavior according to the context information and thus must be Omni-channel and work on all platforms. These issues give some need to a composition mechanism of mobile applications with several versions in order to satisfy the user's requirements and to deal with the features heterogeneity of mobile devices.

In this paper, we propose new approach for smart composition of adaptive mobile applications using existing software entities according to the MDE (i.e. model driven engineering) approach. By this approach we try to fulfill a key software engineering challenge: *the reusability of existing entities* [3] *regardless theirs implementation platforms* and thus *the mobile devices heterogeneity* challenge. Therefore, we aim to address and to solve the heterogeneity problems by allowing heterogeneous composition of adaptive mobile applications. In order to achieve our objectives we propose to: 1) Elaborate a functional model by identifying all desired functionalities and thus the different invocation orders (i.e. identify composition relations by representing dependencies between the desired functionalities), 2) Refining the functional model by attaching for each functionality the different information of the best suited concrete software entities that will be used to implement it (i.e. that is adaptive with the mobile device features), 3) Generating the composite mobile application (CMA) architectural model by replacing each functionality by its selected concrete software entity (i.e. service or component, or application) while respecting mobile device constraints during the composition in order to ensure the correct functioning of the global application.

We implement the passage among these different models using models transformation mechanism where: a) the passage from the functional model to the refined functional model consists in a refining task and implemented with endogenous transformations [4], b) the passage from the refining functional model to the CMA architectural model consists in a projection task and implemented with exogenous transformations [5].

The remainder of this paper is organized as following: Section 2 discusses the related works in the composition domain while Section 3 explains the proposed approach for composing mobile applications and describes their transformation models. Finally, Section 4 concludes the paper.

II. RELATED WORK

In this section we describe several approaches that are focused on composition mechanism. Chakraborty and al. in [10] describe some issues related to services composition in mobile environment. They presented a distributed Service Composition Protocol for mobile environments that take into consideration mobility, dynamic changing service topology and device resources. In [11], authors presented an automatic approach for web service composition, while addressing the problem of process heterogeneities and data heterogeneities by using a planner and a data mediator. In [6], authors created a software infrastructure called AppSpotter that makes possible the dynamic and automated selection and composition of software components for building mobile apps. They proposed mechanism for selection of software components is based-on mobile device features. On the contrary, we were intended in [3] to reify the relevant notions of existing mechanisms of composition in a composite service meta-model. This meta-model defines all interlaced features and provides a global and explicit vision of the service composition. In [7], authors presented a model to design context-aware services that can be exploited as a flexible domain to automatically generate context-aware compositions by means a specific tool. Each of these approaches tries to solve some particular problems from different point of views: composition in terms of components, services, applications, environment of execution: mobile devices or other (e.g. laptops), adaptation capabilities in terms of composing context-aware or not adaptive apps as presented in Table1.

	Composition Object			Composition type		Adaption	
	Components	Services	Heterogeneous entities	(Mobile environment)	(classical environment)	Context-aware	Not-adaptive
chakraborty and al. (2005) [10]		X		X		X	
Zixin and al. (2007) [11]		X			X		X
Anthony and Oussalah (2010) [3]		X			X	X	
Ricardo and Vicente (2011) [6]	X			X		X	
Furno et al. (2014) [7]		X			X	X	

TABLE 1- COMPARISON OF SOME EXISTING COMPOSITION APPROACHES.

These multitude of approaches with theirs features often specialized does not have a global vision of mobile application s composition. Our modest work represents the first attempt to support the development of context-aware mobile applications by composing heterogeneous software entities.

III. Proposed Approach

Our generic approach allows the smart composition of mobile applications by reusing and composing any kind of software entities in order to take advantage of theirs services regardless theirs implementation platforms. It aims also to respect the limited resources offered by the mobile device in which the composed application will be deployed in order to build adaptive mobile applications.

It starts by identifying the needed functionalities ignoring how they will be implemented (i.e. without the limitation of the desired application usability). These choices are performed with respect to the mobile device features such as: availability of GPS device, the presence of an auto-focus camera…etc. The selected concrete entities will be composed using *mediators* in the case of heterogeneous coordination. The same application may be deployed and adapted with any device mobile by recomposing it in according to the new context information (cf Figure1).

Transformation engine1: Refining the CMA functional model

The proposed paradigm aims, as a first step, to define abstractly the desired mobile application. Thereby, it must firstly focus on the definition of the different functionalities which are required to compose the future mobile application and providing the dependencies between these desired functionalities.

The metamodel presented in the Figure 2 allows the former of the application to elaborate the functional model. It allows defining primitive or composite functionalities showing the different needed composition relations. This later specifies Workflow and Dataflow between identified functionalities. Workflow schedules the invocations of functionalities (i.e. establishes precedence links) and Dataflow expresses the data exchanges, inputs to output, between them (i.e. establishes use links). CMA functional model aims to represent the desired application independently of any technology or application domain.

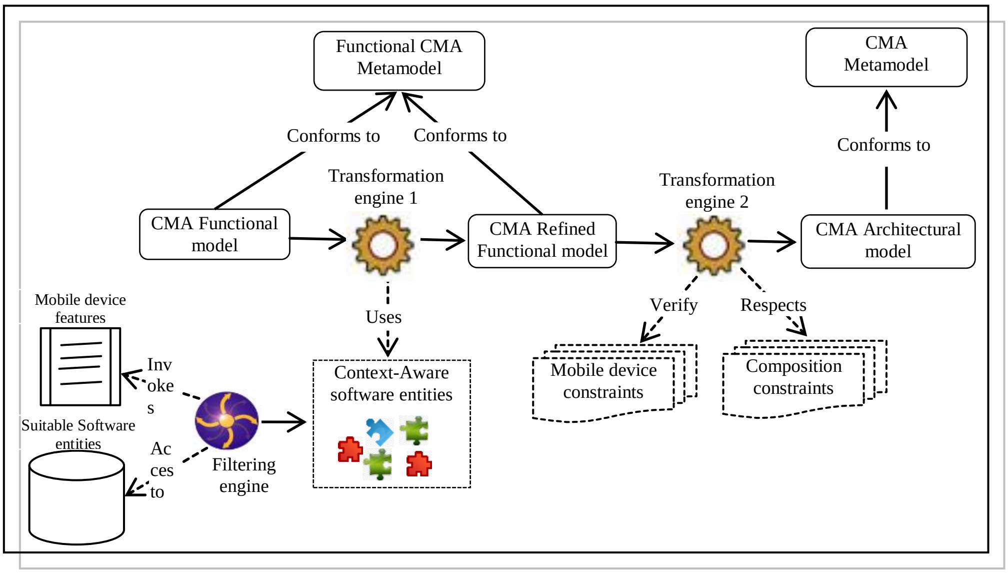

Figure 1- Generic approach overview.

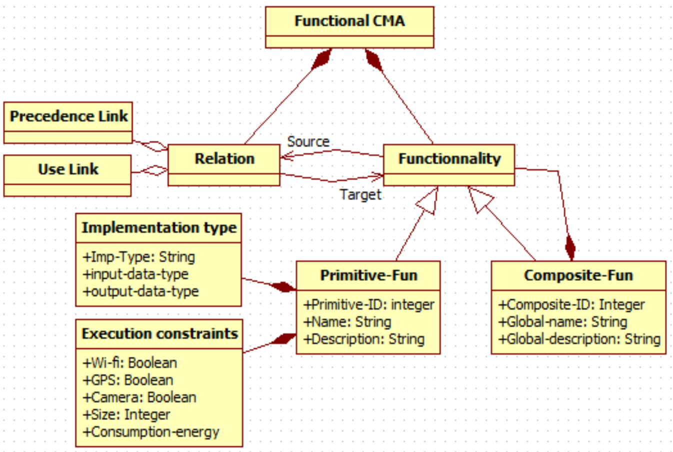

Figure 2- Functional CMA Metamodel.

Now, in order to elaborate the desired CMA towards a specific mobile device we need to attach any identified functionality with its appropriate concrete software entity that will be used to implement it. Each concrete software entity has: implementation type (such as: service, component, or application) and execution constraints (i.e. a list of features required to deploy and execute this concrete entity such as needed capacity storage, needed energy… etc.). We associate this ability with the Transformation engine1 as presented in the Figure1. This later need, as source information, a set of suitable concrete entities those are compatible with the device mobile features to can perform its task.

These selected context-aware concrete entities are the result of the Filtering task (see the Figure 1). After defining the functional model, it is necessary to connect each abstract entity with their corresponding existing concrete entities that can implement it (i.e. existing services which much user's needs). This task is hand performed in our approach; we directly provide the suitable software entities which implement the required functionalities. At this base, Filtering Engine invokes the different characteristics of the mobile device in order to select among all concrete entities of each abstract entity those that are best suited with respect to these characteristics (i.e. deployment and execution context of the mobile device in which the composed application will be deployed). Thereby, filtering engine based on execution constraints of each concrete entity and the different characteristics of the mobile device to perform the comparison in order to select those that are adaptive to these characteristics.

The result of the *Transformation engine1* is an adaptive CMA functional model where this CMA is designed to be executed in a specific mobile device (i.e. Refined CMA functional model).

Transformation engine2: Generating the CMA architectural model

After setting the desired mobile application in an abstract level and after connecting each desired functionality with its corresponding context-aware concrete entity, the proposed approach designed to build the final application architecture. We associate this role to the *Transformation engine 2* (see Figure 1). This later consists to do a projection operation where each functionality in the CMA refined functional model will be represented with its concrete software entity. Thus, it aims to perform the different composition relation among these concrete entities. *Transformation engine 2* must respect the composition constraints to can perform its task.

Composition constraints: Mediation task

Our research work tries to overcome the limit of the different existing composition approach (e.g. [7], [8]) that is: “the use one kind of software entity to compose an application”. It aims to provide an architectural description for mobile applications different from the existing ones (i.e. component based approach, service based approach [9]). This high level description (cf. Figure 4) represents the CMA architecture with: services, components, applications separately or with heterogeneous entities. Thus, it allows building mobile applications as endogenous or exogenous composition. However, heterogeneity issue in this composition mechanism can present two kinds of heterogeneity problems:

- Heterogeneous nature of entities: the composed entities cannot directly communicate because the data which are exchanged between these entities are not understandable (e.g. microphone provides an audio stream and the jukebox needs to string input to perform its task).
- Heterogeneous type of entities: represent the coordination of two different types of software entities (e.g. component connected with service).

The proposed approach aims to address these heterogeneity issues by proposing two kinds of mediators where:

- Endogenous mediator: that overcomes the heterogeneity between two entities of different nature; exchange data can require some transformation to be understandable. Endogenous mediator represents the Mediation services needed to ensure these data transformations (e.g. AudioToText mediation service to transform an audio stream into a string).
- Exogenous mediator: this kind of mediators is intended to eliminate the heterogeneity between two entities of different kinds. These entities cannot directly communicate owing to their different implementation type. Exogenous mediator aims to encapsulate related heterogeneous entities, it builds well-formed interface for each of them in order to take advantage of their services but just with manipulating necessary inputs and outputs independently from implementation languages of these constituents entities.

  At this purpose, this approach treats four type of composition as illustrated in the following table:

Composition Type	Heterogeneity problems	Proposed Mediators
Heterogeneous Exogenous Composition	Heterogeneous Nature of entities Heterogeneous Type of entities	Endogenous Mediators Exogenous mediators
Homogenous Exogenous Composition	Heterogeneous Type of entities	Exogenous mediators
Heterogeneous Endogenous Composition	Heterogeneous Nature of entities	Endogenous Mediators
Homogenous Endogenous Composition	None heterogeneity problems	None needed mediators

Table 2- Composition types.

Transformation engine 2 during the composition task choose to associate the composition relation with exogenous mediator (encapsulate the source and the target entities) and/or endogenous mediator (add a mediation services) in the case of heterogeneous and/or exogenous composition (see Figure 3). In contrary of homogeneous endogenous composition that allows to compose software entities with the same type and the same nature by a simple relationship and therefore without adaptation. Transformation engine2 aims to eliminate heterogeneity problems from the architectural perspective.

Mobile devices constraints: Adaptation task

The composition of context-aware concrete entities is not sufficient to ensure that the CMA it-self will be adaptive to the current context of the mobile device but it is also necessary to satisfy the both following constraints:

- The overall size of composite application does not exceed the remaining storage capacity in the mobile device.
- Handling the composed application does not consume an energy which exceeds the current energy level of the mobile device.

Transformation engine 2 has the role to verify, after each composition step, these two constraints (i.e. if the actual free storage capacity of the mobile device is sufficient for deploy the composed app and if the consumed energy to handle this CMA does not exceeds the current battery level) in order to ensure the correct deployment and the proper functioning of the composed application. Transformation engine 2 aims to construct adaptive mobile applications.

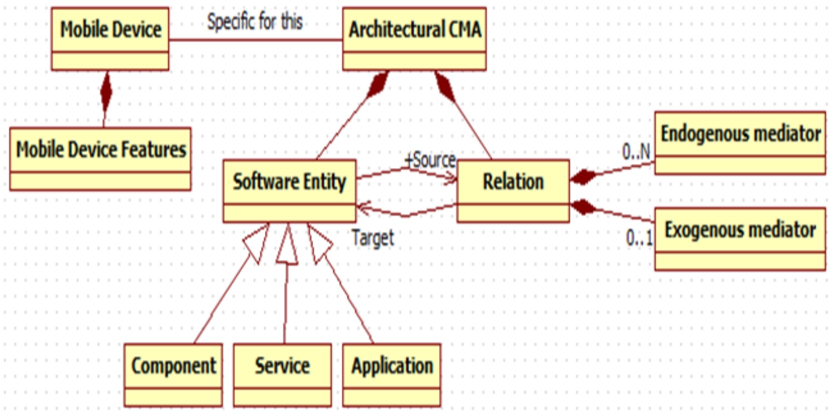

Figure 4- CMA Architectural Model

IV. Conclusion

In this paper we have presented a conceptual framework for our proposed idea to provide a new approach for smart composition of mobile device applications. It aimed to meet user's requirements. It also intended to compose mobile applications that are sensitive and adaptive to the context information of the mobile device in which they will be installed. In this work, we are addressed the heterogeneity challenge during the composition of different kind of software entities and thus the mobile device heterogeneity challenge when deploying these composed mobile applications.

As future works, we intend to implement these proposed transformation engines according to three methodologies: Bottom-Up, Top-Down, Mixt. And thus, to use *ModelToCode* transformation mechanism to generate the application code from the CMA architectural model presented in this work.

References

[1] Jeffrey S. Hammond and Julie A. Ask , The Future Of Mobile Application Developmentn, January 17, 2013.

[2] Cugola, G., et al., Selfmotion: A declarative approach for adaptive service-oriented mobile applications. Journal of Systems and Software, 2013.

[3] Hock-Koon, A. and M. Oussalah. Expliciting a composite service by a metamodeling approach. in Research Challenges in Information Science (RCIS), 2010 Fourth International Conference on. 2010. IEEE.

[4] Barbier, Eric Cariou—Nicolas Belloir—Franck. "Contrats de transformation pour la validation de raffinement de modèles." IDM 2009 Actes des 5emes journées sur l'Ingénierie Dirigée par les Modèles (2009): 1.

[5] CZARNECKI, Krzysztof et HELSEN, Simon. Feature-based survey of model transformation approaches. IBM Systems Journal, 2006, vol. 45, no 3, p. 621-645.

[6] Rosa, R.E.V. and V.F. Lucena Jr. Smart composition of reusable software components in mobile application product lines. in Proceedings of the 2nd International Workshop on Product Line Approaches in Software Engineering. 2011. ACM.

[7] Furno, A. and E. Zimeo, Context-aware Composition of Semantic Web Services. Mobile Networks and Applications, 2014. 19(2): p. 235-248.

[8] Hock-Koon, A. and M. Oussalah, Composite service metamodel and auto composition. Journal of Computational Methods in Science and Engineering, 2010. 10: p. 215-229.

[9] Abdelkrim Amirat, Hock-Koon Anthony, and Mourad Chabane Oussalah, Chapter 1: Object-Oriented, Component-Based, Agent-Oriented and Service Oriented Paradigms, In M. C. Oussalah, editor, Software Architectures 1, Wiley-ISTE, April 2014.

[10] Chakraborty, D., et al., Service composition for mobile environments. Mobile Networks and Applications, 2005. 10(4): p. 435-451.

[11] Furno, A. and E. Zimeo, Context-aware Composition of Semantic Web Services. Mobile Networks and Applications, 2014. 19(2): p. 235-248.