❒ 596

Development of Mobile Cloud Applications using UML

Dong Kwan Kim
Department of Computer Engineering, Mokpo National Maritime University, 91, Haeyangdaehak -ro, Mokpo-si,
Jeollanam-do, Korea

Article Info	ABSTRACT
	With the proliferation of cloud computing technologies, smartphone users are able to use a variety of cloud computing-based mobile services such as games, education, entertainment, and social networking. Despite the popularity of such a mobile cloud computing, the complicated multi-tier system configuration of the mobile application must be one of the major impediments to develop mobile cloud applications. This paper presents development processes and procedures for developing mobile cloud applications by effectively applying Unified Modeling Language (UML), a representative object-oriented modeling language. The paper is intended to enhance the development productivity of the mobile cloud application and to improve the effectiveness of communication between software developers. In addition, we used the Android mobile platform and Amazon Web Service for cloud computing in order to demonstrate the applicability of the proposed approach to systematically apply the UML profiles and diagrams for cloud-based mobile applications.

Corresponding Author:

Dong Kwan Kim,
Department of Computer Engineering,
Mokpo National Maritime University,
91, Haeyangdaehak -ro, Mokpo-si, Jeollanam-do, Korea.
Email: dongkwan@gmail.com

1. INTRODUCTION

The convergence of mobile and cloud computing has opened a new era of mobile cloud services in a variety of areas ranging from education, healthcare, business as well as games, entertainment, and social networks. Due to the increase of the availability of high-end and high-volume computing resources, cloud computing enables 24-hour continuous service delivery for mobile users. Thus, such mobile services can be provided whenever mobile clients want and wherever they are without time and space constraints. The sustainability of mobile applications has been increased by utilizing high computing power [1]. Despite the popularity of the mobile cloud computing, it is not trivial for software developers to construct mobile applications which run on cloud servers over the Internet. While the inherent characteristics of such computing infrastructures support the scalability and dependability of the mobile application, they can make it difficult to develop the cloud-based mobile application. The mobile cloud application often runs on the multi-tier enterprise architecture including client layers (mobile clients), application layers (application servers), and database layers (database servers). Such a complicated multi-tier system configuration of the mobile application must be one of the major impediments to develop mobile cloud applications.

This paper presents development processes and procedures for developing a mobile cloud application by effectively applying Unified Modeling Language (UML) [2], a representative object-oriented modeling language. Software development activities and procedures will be proposed to build mobile applications on the cloud service platform by applying UML diagrams and artifacts. UML can shorten the gap between software design and implementation and reduce the complexity of the development of the mobile application. UML provides the expressive notations for diagrams so that it represents the different

perspectives of a target system. UML is a general-purpose modeling tool that can be used to design various domain applications through extension mechanisms. The UML extension mechanism, called the UML profile, contains stereotypes, constraints, and tag values. This paper provides UML profiles for mobile cloud applications so that software developers use these profiles to effectively visualize cloud-based mobile applications. The proposed approach to design the mobile application provides systematic guidelines for applying UML profiles extensions, class diagrams, and deployment diagrams. To demonstrate the effectiveness of the proposed approach, case studies have been conducted by developing a mobile application which runs on the typical three-tier architecture including the Android mobile platform [3], Amazon Web Service [4] for cloud computing, and MySQL for data management.

The rest of this paper is organized as follows. Section 2 fully describes the proposed approach to modeling mobile cloud application using UML. Section 3 presents the results of the case study to demonstrate the effectiveness of the proposed methodology. Section 4 describes the strength and weakness of the proposed approach. Section 5 introduces the related work and finally Section 6 remarks the conclusions and future work directions.

2. PROPOSED APPROACH

The UML provides lightweight extension mechanism to allow adaptation or extension of the standard UML metamodel elements for different software platforms or application domains. We can define custom stereotypes, tagged values, and constraints to express more specific information on UML models. Table 1 shows the stereotypes for the Android mobile platform which are used in this paper. The main components of an Android application are represented as stereotypes—activity, service, content provider, and broadcast receiver. Along with these application components, the Android-specific elements are listed as stereotypes to express unique features of the Android mobile platform.

Table 1. Stereotypes for the Android Mobile Platform

Stereotypes	TargetElements	Description
Activity	UML Class	Units or building blocks of Android applications
Fragment	UML Class	Sort of a mini activity
Thread	UML Class	Independently executable code block
Service	UML Class	Components that execute their own tasks in the background
Content Provider	UML Class	Components that store and manage data of Android apps.
Broadcast Receiver	UML Class	Components that register and receive an event from the Android runtime
Explicit Intent	UML AssociationClass	-A message from one component to another -Specify a called component
Implicit Intent	UML AssociationClass	-A message from one component to another -Specify an action w/o a called component
Bundle	UML Class	Bundles are used for passing data between activities and they can be included in the Intent
AndroidManifest	UML Class	It can contain information on an Android application including components, permissions, and used android APIs
Resource	UML Class	All non-code assets associated with an Android application such as images, layouts, strings, etc.
Java Native Interface (JNI)	UML Class	JNI allows for calling native codes written in C or C++ within Android code
External Library	UML Subsystem	-External libraries that will be used in the Android application -The Gradle build system specifies external libraries in the dependencies block of the build.gradle file

Figure 1 presents the UML profile diagram for the Android mobile platform. The UML profile diagram describes custom stereotypes, tagged values, and constraints so that it can extend standard UML models. The proposed profile diagram specifies the stereotypes described in Table 1. The profile diagram may include metaclasses, stereotypes, extension relationships, and generalization relationships. Metaclass is a profile class which can be extended through stereotypes. In the profile diagram, the arrows with the filled triangle arrowhead mean an extension association relationship between the metaclass and the stereotype. All the stereotypes can extend the properties of the metaclass. The other relationship is a generalization which is depicted in a line with a hollow triangle as an arrowhead. In the profile diagram, <<stereotype>> Component can be generalized Activity, Service, Content Provider, and Broadcast Receiver. These generalization

relationship represents there can be four main components as the essential building blocks within an Android application—Activities, Services, Broadcast Receivers, and Content Providers. In addition, <<stereotype>> Intent is generalized as Explicit Intent and Implicit Intent. The Intent is glue for connecting activities in an Android application. Intents make it possible for activities or applications to be navigable from one to another. The Intent is a messenger including data which should be delivered between activities. While Explicit Intent specifies a called component within a calling component, Implicit Intent specifies an action without pointing out a called component.
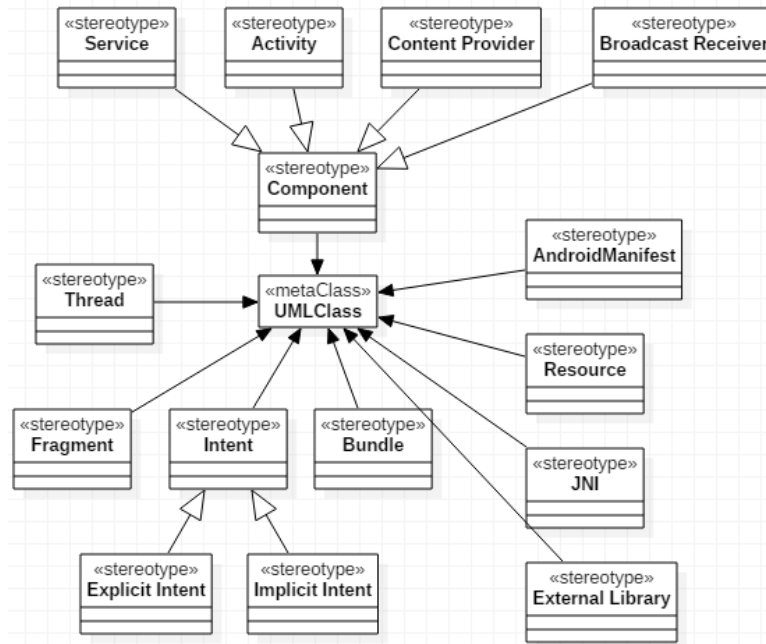

Figure 1. The UML Profile Diagram for the Android Mobile Platform

Figure 2 illustrates the overall process of the proposed approach to design mobile cloud applications using UML. Use case diagrams analyze users' textual requirements so that it can elaborate them by avoiding unclear and ambiguous parts of the requirements. Once the requirements are extracted, conceptual class diagrams can be modelled from the requirements. The conceptual class diagram aims to identify the main classes and their relationships at the analysis level. The attributes and operations of the class can be determined if needed. The conceptual class diagram represents the structural perspective of a mobile cloud application. Other UML diagrams including activity diagrams can be used to clarify the previous modeling information and to represent the dynamic view of the target application. Figure 2 shows the activity diagram, state diagram, and sequence diagram due to space limitations. In the design phase, the class diagram includes more detailed information on the target system. In addition, the UML profiles for mobile platforms should be added in the class diagram so that they can express more specific information of the classes. The class diagrams are passed as input into the package diagrams which describe the logical distribution of the mobile cloud application. The package diagram can be separated into three parts: mobile client layer, business layer and data layer. As the name suggests, the mobile client layer contains the mobile client packages which provide user interfaces for application users. The packages in the business layer perform business tasks by exchanging data with the packages in the mobile client layer. The packages related to data sources reside in the data layer. Each layer can communicate with other layers. The deployment diagram provides the physical distribution of the mobile cloud application. Therefore, operational computing systems such as mobile devices, application servers, runtime environments, and database management systems can be specified in the deployment diagram. It also contains the network protocols which allow components in a node to transmit information. A mobile application developer implements mobile code, application server code, and DB server code from the modeling diagrams. The mobile code can be run on the Android platform or the iOS platform. JSP, Servlets, or PHP can be used for the application server code and DB server code.
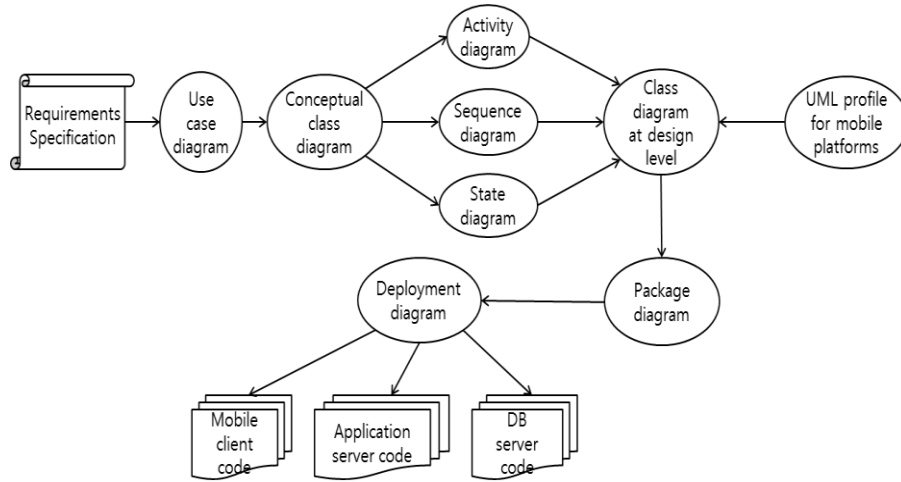
Figure 2. The development process for the mobile cloud applications

3.   CASE STUDY

We have applied the proposed approach into a bulletin board application which operates on a mobile cloud infrastructure. Such a case study may evaluate the effectiveness of the proposed approach that uses the UML artifacts to design mobile cloud applications. For the case study, the bulletin board system has been considered since it is a typical and common module for multi-layer applications. We have applied the proposed approach into a bulletin board application which operates on a mobile cloud infrastructure. Such a case study may evaluate the effectiveness of the proposed approach that uses the UML artifacts to design mobile cloud applications. For the case study, the bulletin board system has been considered since it is a typical and common module for multi-layer applications.

The requirements for the bulletin board application are straightforward and Figure 3 shows an excerpt of the UML use case diagram for the BBS Android application which runs on a cloud computing infrastructure. There are main eight use cases and one actor in the diagram for the BBS application. The BBS application supports to manage multiple bulletin boards and provides basic functions for each bulletin board. Actor Member can browse a list of the bulletin boards and a list of the items in a bulletin board. He or she can also look at the contents of the selected item. After being verified, he or she can upload, modify, delete, and search messages on the bulletin board. Use case Login has the <<include>> relationship with the bulletin board operations.
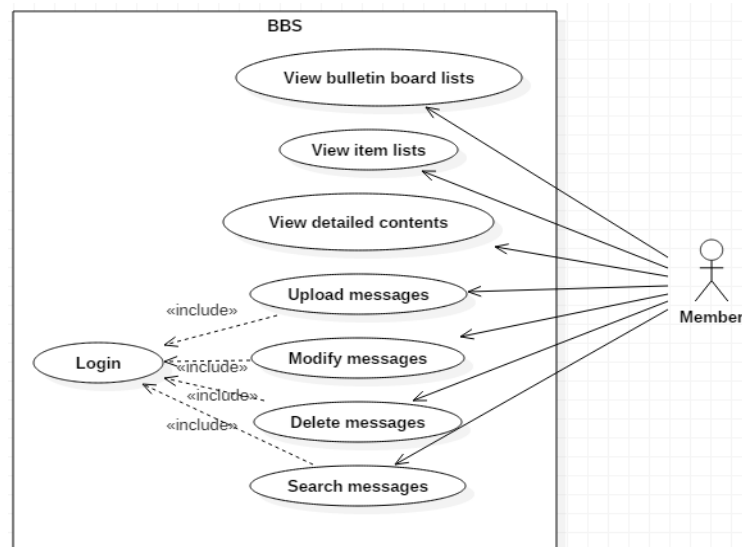

Figure 3. The Use Case Diagram for the BBS Android application

Once the software developer considers the use case diagram completed, it is passed into the next modeling task which will create the conceptual class diagram to represent central classes and relationships between them in the BBS application. The conceptual class diagram represents the structural perspective of the mobile cloud application. The conceptual class diagram may contain some attributes and operations of a class. Even if the conceptual class diagram cannot provide fully detailed information for the implementation, it is obvious that such a diagram may be useful to sketch the comprehensive aspects of the mobile cloud application in the initial modeling. After completing the conceptual class diagram, we can construct other UML diagrams and artifacts including activity diagrams, state diagrams, and sequence diagrams. They help describe the dynamic perspectives as well as the structural perspectives of the BBS application.

The next step is to construct class diagrams which model the BBS application at the design level. Compared to the conceptual class diagram, the class diagram contains the UML profiles for the Android mobile platform presented in Table 1. It is expected for the class diagram to capture more specific information on the BBS application using the UML profiles. Figure 4 shows the class diagram which expresses the stereotypes for the Android mobile platform. The class diagram is based on the MVC design pattern. All the classes in the diagram are categorized into three parts-model classes, view classes, and controller classes according to the characteristic of the class. The model class maintains data in the problem domain. The view class provides user interfaces to display all or a portion of the data based on changes in the model. The control class can handle events that affect the model's state or the view's presentation. Most view classes can be represented with stereotype <<Activity>> since the Android activity is a code block of the user interface. The Android activity class has an Android manifest class which contains information on the BBS application including components, permissions, and used android APIs. An activity class can navigate to another activity class by passing an explicit intent in the BBS application. The intent classes can have class Bundle to pass data between Android activities. The bundle class can hold all types of values by using key-value mappings. In the diagram, some classes are denoted by stereotype <<Thread>> which indicates the corresponding classes perform an independent task from the main UI thread.
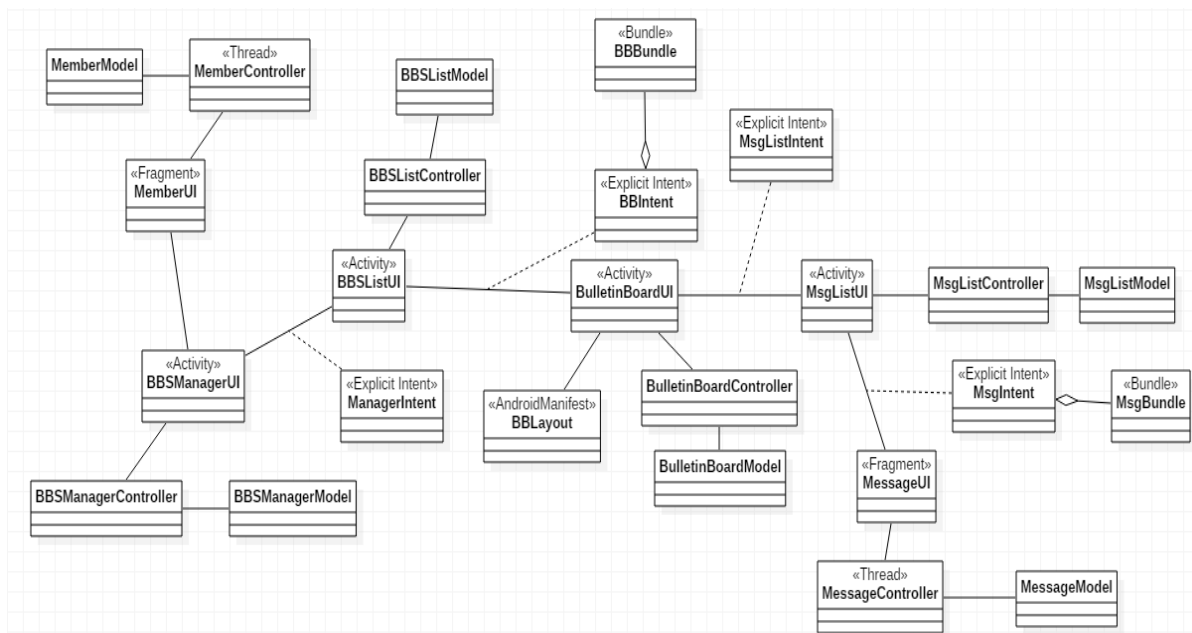

Figure 4. The Class Diagram with the Stereotypes for the BBS Android application

In this paper, package diagrams and deployment diagrams are used to visualize the features of multi-layered applications. Figure 5 shows the package diagram for the BBS Android application which describes the logical distribution of the mobile cloud application. The package diagram consists of the mobile client layer, business layer, and data layer. The mobile client layer contains package modules for user interfaces where users can enter events or requests and receive results. The packages in the mobile client layer hold the view classes in the class diagram. The business layer takes data from the mobile client layer and then performs business tasks with the data. The business task can involve in managing some data through the data

layer. The results of the completed tasks can be sent to the mobile client layer or be stored into the data sources in the data layer.

The final step of the proposed approach is to build a deployment diagram which visualizes the physical distribution of the BBS application. Figure 6 shows the deployment diagram for the BBS Android application. In the deployment diagram, we can illustrate runtime environments, mobile platforms, application servers, containers, database, and network protocols. We can understand the physical configuration of the operating system for the BBS application. The deployment diagram can be constructed from the package diagram. In Figure 6, the Android mobile platform is used for the client layer and the BBS Android package file resides on the client layer. The Apache Tomcat server is used for the application server and runs JSP pages for the BBS application. The AndroidMobileDevice node can exchange data with the ApplicationServer node via the HTTP protocol. The MySQL database places on the DBServer node. The JSP pages can access data via the JDBC API.
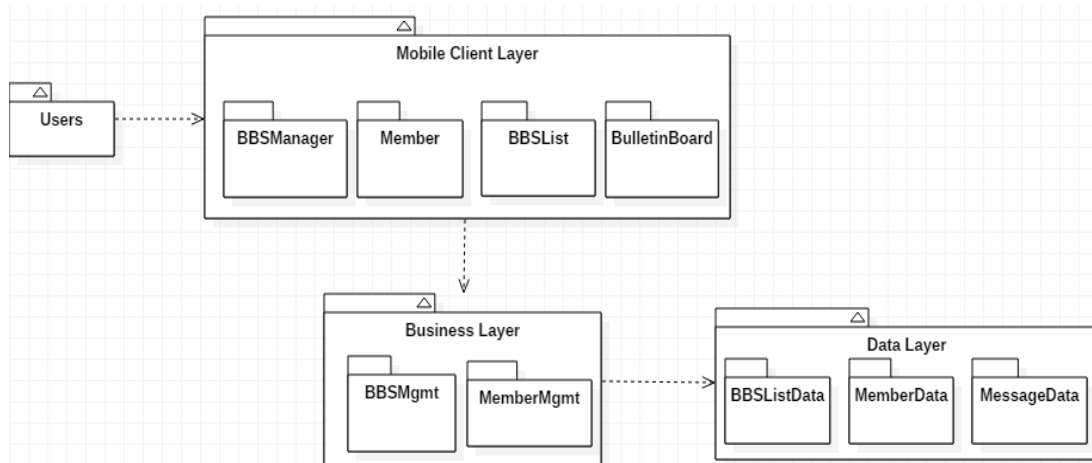

Figure 5. The Package Diagram for the BBS Android application

A mobile application developer can implement the nodes with different programming languages. The mobile client node can be run on the Android platform or the iOS platform. The modules in the application server node can be written in the server-side programming languages such as JSP, Servlets, and PHP. For the DBServer node, the software developer can use a typical relational DBMSs or NoSQL databases.
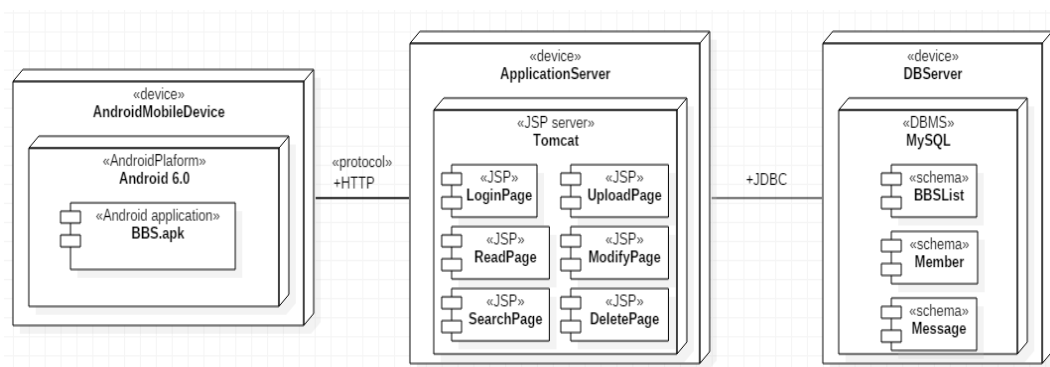

Figure 6. The Deployment Diagram for the BBS Android application

The UML profile for mobile cloud application presented in this paper has been evaluated with a typical bulletin board mobile application which runs on the Amazon Web Service for cloud computing. Figure 7 shows the system architecture of the BBS application which consists of the mobile client, the logic

server, and the database server. The bulletin board mobile application includes typical functions—to display the list of bulletin boards, to display the list of posted items, to display the detailed contents of the selected item, to write a new message, to delete a message, and to modify a message. Users can post images as well as simple text. The user requests through the mobile client are delivered to the logic server which will store, retrieve, delete, and modify the postings. The BBS application uses the MySQL DBMS to store and manage the uploaded data.

The BBS application uses the Retrofit library [5] which is a type-safe HTTP client program for Android applications. It provides the RESTful HTTP APIs including GET, POST, PUT, and DELETE. The server-side functions of the mobile application have been implemented in JavaServer Pages (JSP) which enables software programmers to build web pages dynamically using Java technologies. For the data exchange between clients and servers, the BBS application uses the JavaScript Object Notation (JSON) data format [6]. JSON defines a data format which consists of attribute-value pairs. The syntax of the JSON standard is simple so that developers can easily specify a JSON-based data format and validate the correctness of the defined format. The mobile application uses the JSON format to exchange data over the Internet.
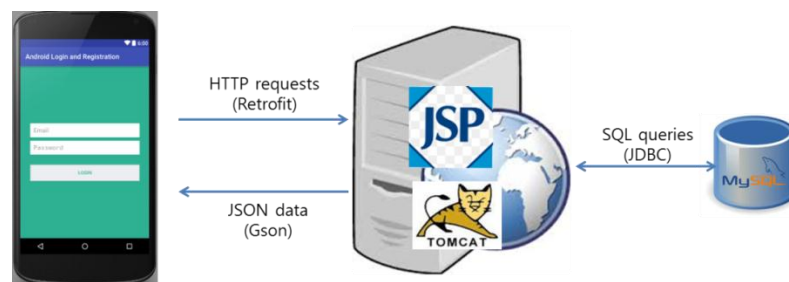

Figure 7. The System Architecture of the Mobile Cloud Application for the Case Study

4.   DISCUSSION

The main contribution of this paper is to provide a UML modeling guideline for developing mobile cloud applications. Since the multi-layer application often runs on heterogeneous computing environments, it is tricky for the software developer to build the applications without the aid of practical and stepwise methods. Many mobile apps have been developed and run on the cloud computing infrastructure on which the server program resides. Due to such a different platform, the software developer should use different programming languages, software development kits, and runtime systems. The mixed multi-tier application could lead to some difficulties of modeling of the application in the analysis and design phases. Even if the proposed approach to design the mobile application does not provide the fine-grained steps and practices, the development processes in this paper allow the software developer to describe and capture the central portions of the mobile cloud application. Another limitation of the proposed approach is that it does not support any software development tools and code generation. However, since the only standard UML notations are used, most of the UML tools could be used to support the proposed stereotypes and diagrams.

5.   RELATED WORK

Since the introduction of object-oriented software development techniques, UML has provided systematic guidelines and regulations to improve the performance of software development. In addition, the research results have been published that apply UML to various application domains ranging from desktop applications [7] and Web applications [8] to embedded applications [9]. In particular, due to the widespread use of mobile devices, UML has been applied to improve the productivity of mobile application development. The UML meta-models and profiles have been extended to provide best development practices and guidelines for Android mobile applications [10], [11]. Mobile platform-specific source code can be automatically generated from UML models to improve development productivity. UML is also used to specify structural and dynamic models in Model Driven Development (MDD) [12], a software development methodology based on Model Driven Architecture (MDA). Along with the UML modeling concepts and notations, MDD supports efficient and effective software development via model conversion and automatic code generation using Platform Independent Model (PIM) and Platform Specific Model (PSM). Due to the productivity, MDD is actively applied to build mobile applications [13-17]. Fabiano Freitas and his

colleagues introduced an MDD-based approach to develop Android applications, called *JustModeling* [18] which provides UML-based modeling diagrams and code generation. The MDD approach is also applied to construct embedded sensor-based applications in a mobile platform [19] or Android application prototypes [20]. For all mobile platforms, a meta-model provides a high-level abstraction that is independent from a specific mobile platform. The abstract model will be transformed into a mobile platform-specific model according to the mapping rules. Finally, the source code of a mobile application will be generated from the specific model by using a code generation template.

6. CONCLUSIONS AND FUTURE WORK

This paper has proposed software development activities and procedures to build mobile applications on the cloud service platform by applying UML diagrams and artifacts. The mobile cloud application often runs on the multi-tier enterprise architecture including client layers (mobile clients), application layers (application servers), and database layers (database servers). The proposed approach to design such applications provides systematic guidelines for applying UML profiles extensions, class diagrams, and deployment diagrams. The experimental results suggest that the proposed guidelines are able to improve the productivity, scalability, and maintainability of software design models. For the case studies, we used the Android mobile platform, Amazon Web Service for cloud computing, and MySQL for data management.

As a future work, the proposed approach will be tested and refined on the multi-platforms such as the iOS mobile platform and Windows Mobile other than the Android platform. We believe that the extension to other mobile cloud applications may be straightforward because mobile platforms show common characteristics in many ways. In addition, to improve the formal specification of the proposed rules, the Object Constraint Language (OCL) will be applied for more precise representation.

REFERENCES

[1] K. I. Koshy, et al., *"Can Cloud Computing Lead to Increased Sustainability of Mobile Devices?."* IEEE International Symposium on Sustainable Systems and Technology (ISSST), May 2012.

[2] Object Management Group, "Unified Modeling Language TM (OMG UML) Version. 2.5." March 2015.

[3] Android Mobile Platforms, https://developer.android.com.

[4] Amazon Web Services, https://aws.amazon.com.

[5] Retrofit, http://square.github.io/retrofit/.

[6] Introducing JSON, http://www.json.org/

[7] Y. Junru, et al., "Architecture Design of the Cladding Part Modeling System," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 11, no. 9, Sep. 2013, pp.5237-5244.

[8] S. Singh, et al., "Advance Billing and Metering Architecture for Infrastructure as a Service*," International Journal of Cloud Computing and Services Science (IJ-CLOSER)*, vol.2, no.2, April 2013, pp. 123-133.

[9] A. Kaur, et al., "Application of UML in Real-Time Embedded Systems*," International Journal of Software Engineering & Applications (IJSEA)*, vol.3, no.2, March 2012.

[10] K. Minhyuk, et al*., "Extending UML meta-model for android application,"* International Conference on Computer and Information Science (ICIS), 2012 IEEE/ACIS, pp. 669-674.

[11] A. Bergmayr, et al*., "UML-based Cloud Application Modeling with Libraries, Profiles, and Templates,"* 2nd International Workshop on Model-Driven Engineering on and for the Cloud (CloudMDE), Sep.2014.

[12] M. Yaghoubi, et al., "Different Proposed Models to Mapping MDA to RUP," *International Journal of Electrical and Computer Engineering (IJECE),* vol. 3, no. 3, June 2013, pp. 301-306.

[13] S. Vaupel, et al., "Model driven development of mobile applications for Android and iOS supporting role-based app variability*," Software & Systems Modeling*, 2016, pp. 1-29.

[14] M. Usman, et al., *"A Model-driven Approach to Generate Mobile Applications for Multiple Platforms,"* 21st Asia-Pacific Software Engineering Conference (APSEC), 2014.

[15] A. Parada, et al., "Automating mobile application development: UML-based code generation for Android and Windows Phone," *Journal of Theoretical and Applied Informatics (RITA),* vol. 22, no. 2, Jan. 2015, pp 31-50.

[16] A. Parada, et al*., "A model driven approach for Android applications development,"* 2012 Brazilian Symposium on Computing System Engineering (SBESC), Nov. 2012.

[17] A.Ribeiro, *"Development of Mobile Applications using a Model-Driven Software Development Approach,"* Master's thesis, Instituto Superior Técnico, Nov. 2014.

[18] F. Freitas, et al., *"JustModeling: an MDE Approach to Develop Android Business Applications,"* VI Brazilian Symposium on Computing Systems Engineering (SBESC), Nov. 2016.

[19] M. Lachgar, et al., "Rapid Mobile Development: Build Rich, Sensor-Based Applications using a MDA approach," *International Journal of Computer Science and Network Security*, vol.17, no.4, April 2017.

[20] [C. Thanaseth, et al., *"Model Driven Development of Android Application Prototypes from Windows Navigation Diagrams,"* International Conference on Software Networking (ICSN). IEEE, 2016, pp. 1-4.

BIOGRAPHY OF AUTHOR

Dong Kwan Kim is an Assistant Professor in the Department of Computer Engineering at Mokpo National Maritime University. His research interests include dynamic software updating, run-time systems, and mobile programming.