2009 Sixth International Conference on Information Technology: New Generations

Applying Domain-Specific Modeling to Mobile Health Monitoring Applications

Florence T. Balagtas-Fernandez and Heinrich Hussmann
Department of Computer Science, University of Munich, Germany
{florence.balagtas, heinrich.hussmann}@ifi.lmu.de
http://www.medien.ifi.lmu.de

Abstract

This paper introduces a conceptual framework and process that aims to simplify the development of applications for mobile devices. We specialize on the domain of health monitoring by providing a whole system that would allow people to model their own mobile health monitoring applications in a simplified manner without dealing with low-level coding themselves.

1. Introduction

Nowadays, there has been an increase in consciousness on how people take care of their health. People have devised so many ways in order to stay fit and motivated by trying to apply different technologies in helping them maintain good health. Examples of such technologies are monitoring devices such as the MOPET wearable system [4], MPTrain Personal Trainer [6] and Nike+iPod Sports Kit [3] that helps people be motivated in their exercise routines. Other types of devices such as the Alive Technologies Heart Monitor and Alive Pulse Oximeter [1] are mainly used by people who are already suffering from specific ailments and would want to avoid serious complications by closely monitoring specific aspects of their health. However, the problem with such devices and their supporting applications is that they already have predefined displays and functionalities. In order to create a customized application with the information collected from these health monitoring sensors, frameworks have been created to allow easy access of information through API calls, and allow customized applications to be created in specific devices (e.g. MobHealth Framework [2]). Having these types of frameworks allows people to easily access low level data that is sent by the sensors. However, another problem here is that, to achieve customization by the users themselves, users should know some programming. They should know how to use the APIs provided by these frameworks in order to create their own applications. It would be better if there was a way for users to easily create their own applications without low level coding.

The following are some example scenarios that illustrates how end-user creation of mobile health monitoring applications could be beneficial. Suppose a person wants to have an application on his mobile device coupled with some sensors that would help keep track of how much time he spends on physical activities. However, he wants something that is customizable and other additional functionalities not provided by the vendor. He could just use a modeling tool that would allow him to easily create this specialized application based on his wishes. Another useful application would be in a clinic that provides patients with such health monitoring devices to keep track of specific aspects of their patients' health and prevent complications. However, each patient is a unique case and an application tailor-made for each patient would be very helpful. In this case, one of the doctor's assistants would be assigned to create these customized health monitoring applications based on the patient's needs and as instructed by the doctor. This would be like having some pharmacist concoct a special medicine prescribed by the doctor for a certain patient. Another application is in the medical research field. Some researches involve having test subjects wear some non-invasive devices to monitor the subject's activities or temperature for instance, and have this collected data analyzed by the researchers (e.g. http://thewep.org). It would be helpful for these researchers to have some tool that they could use to create customized applications depending on the experiments they have devised, and not constantly rely on a programmer to create the application for them.

Having a modeling tool that would allow the easy creation of applications for mobile health monitoring would be very useful for non-expert users who would like their own custom made applications on their devices. It would be synonymous to the current WYSIWYG web page editors that allows even non-experts in the field of programming, to create their own web pages.

This paper proposes a conceptual framework and process that would allow the simplification of developing ap-

978-0-7695-3596-8/09 $25.00 © 2009 IEEE
DOI 10.1109/ITNG.2009.14

plications for mobile health monitoring by applying model-driven development (MDD), specifically the domain specific modeling approach (DSM)[5]. The process starts by having a graphical model of the application which contains specific constructs of the target domain which is mobile health monitoring. This graphical model is then processed by an intermediate framework which will decide how to interpret the model and transform this model to some specific code based on a the target platform. Further details of our approach will be discussed in the next section.

2. The Mobia Modeling Framework: Modeling Mobile Health Applications

The Mobile Applications Modeling Framework (Mobia Modeling Framework) aims to simplify the development of mobile applications by applying the DSM approach. For this particular project, we introduce a mobile health monitoring module in the mobia framework which allows the easy creation of applications for mobile health monitoring.

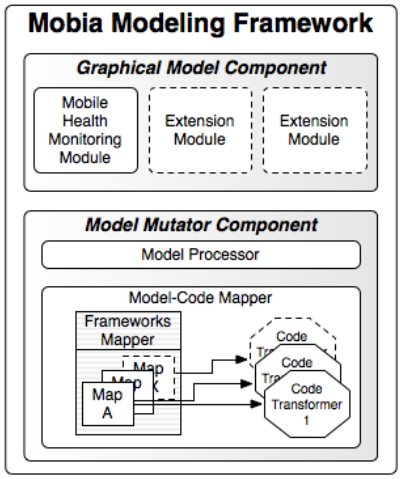

Figure 1. Overview of the Mobia Modeling Framework

Figure 1 shows an overview of the components of our conceptual framework which basically consists of two major components: the *Graphical Model Component* and the *Model Mutator Component*. The Graphical Model component is responsible for the visualizations of the modeling constructs. As seen in figure 1, we have designed it to be extensible and to support other types of modules for solving other problem domains in the area of mobile development. However, for this particular stage of our research, we focus mainly on mobile applications involved in health monitoring. The Model Mutator Component is responsible for processing the model and code transformation.

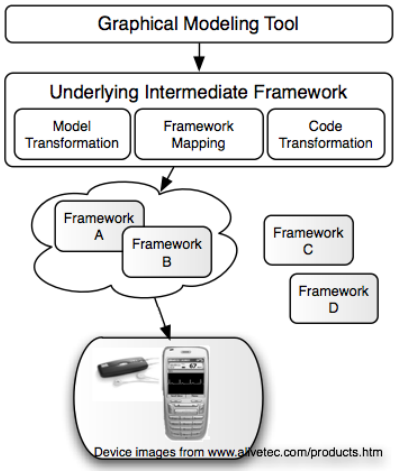

Figure 2. The Process

Figure 2 shows an overview of the process of how an application in the Mobia framework is created. First, a graphical model is created using the Mobia Modeler Suite. This graphical model is then passed on to the model mutator for processing. Referring to figure 1, the *model processor* handles the interpretation and checking of the model. After it is done processing the model, it then passes the information to the *model-code mapper*. Inside the model-code mapper, the information is first processed inside the *frameworks mapper module* in which it contains a list of possible combinations of frameworks that is used by a certain model to create an application. For instance, if the application uses a certain sensor device which provides some framework M to access the data transmitted by this device, and then the framework responsible for showing visualizations on the mobile device is from some framework N, then the specific *mapper* containing such list of frameworks will invoke the *code transformers* for frameworks M and N. The *code transformers* will then output the necessary code for the final application.

References

[1] Alive technologies. http://www.alivetec.com/products.htm.

[2] Mobhealth framework. http://sourceforge.net/projects/mobhealth/.

[3] Nike + ipod sports kit. http://www.apple.com/ipod/nike/.

[4] F. Buttussi and L. Chittaro. Mopet: A context-aware and user-adaptive wearable system for fitness training. *Artificial Intelligence In Medicine*, pages 153–163, 2008.

[5] S. Kelly and J.-P. Tolvanen. *Domain Specific Modeling, Enabling Full Code Generation*. IEEE Computer Society Publications and John Wiley and Sons Inc, 2008.

[6] N. Oliver and F. Flores-Mangas. Mptrain: a mobile, music and physiology-based personal trainer. In *MobileHCI '06: Proceedings of the 8th conference on Human-computer interaction with mobile devices and services*, pages 21–28, New York, NY, USA, 2006. ACM.