Utilization aspect oriented approach and local data in the development of adaptive user interface for Android

Jiří Šebek[1]

Dept. of Comp. Science and Engineering, Czech Technical University, Karlovo nám. 13, 121 35 Praha 2, Czech republic

sebekji1@fel.cvut.cz

Abstract.
This paper deals this issues of the Context-Aware Computing and explores the possibilities of utilizing the obtained context for creating the adaptive user interface. The proposed framework is used to develop adaptive user interface, uses a context of hardware devices for mobile OS Android and follows the style of Aspect-oriented programming. Framework principles are based on modular architecture. One of the main module is the aspect module providing the programmer easy and fast access to the context of mobile device sensors in order to help the individual aspects. Additionally, it offers access to the more advanced, specific aspects, which are used for automatic generation of adaptive interfaces in the runtime.

Keywords

adaptive user interface, context based approach, aspect-oriented approach, aspect-driven design, reduced maintenance and development efforts

1. Introduction

Currently, the number of people, who does not own smartphone, is becoming less and less. Most people wear almost daily phone in their pocket (if they go to work, when they are on vacation, if they are shopping or at home - wherever they are in the vicinity of the phone, and this fact can be used in the interests of developers). The development process focused on user provides a degree of compliance with its requirements.

Adaptive kind of thinking helps developers find a solution that can dramatically improve user interaction with the environment.

Using data received from sensors phone developer acquires the ability to analyze user activity, predict its future behavior and use this information to create an adaptive interface that is adjusted to the individual needs of the user.

The main aim of this paper is to create an Android framework that will allow a developer to create Adaptive Android applications with minimal effort in a short time with the aspect-oriented approach.

2. Background

Context is the basis of human communication. In life usually just saying a few words is enough for telling the meaning of the message. In order to quickly understand the human subconscious message reads context environment is saved and used to "decrypt" incoming messages from other people. So it is context of history, context of their own experiences of a person, context that can be gained by sensory organs. However, devices are not people, but they also have their "sensory organs" which ensures their ability to perceive and remember the context. It is necessary to use all possibilities of the device because it will increase the level of communication with the user. A great benefit is the ability of a device to adapt to user needs. Such "smart" device is convenient and enjoyable to use.

Adaptive interface is one possible tool for adaptivity. In order to elaborate adaptive interface a deep understanding of the context is required, which can be obtained for example by sensors of mobile device. The main advantage of contextual approach is the ability to discern important information for a given user based on the received contextual characteristics. The interface is adjusted in such a way that the user do not perform redundant work and do not waste time on things that device can perform independently.

There interdisciplinary science "theory of human-computer interaction" (Eng. Human Computer Interaction) or abbreviated - HCI. HCI is a science that examines the methods and ways of interaction between users and computers. This science is a summary of diverse areas such as: computer science, programming, engineering, psychology, design, and more.

Ubiquitous computing (Ubicomp) [10] (or pervasive computing) is a model that describes the penetration of computers in all sorts of human activity spheres. Ubicomp it is possible to imagine a post-PC model HCI. Ubicomp is the era of smart devices entering the human environment. Devices must be able to independently gather information about the environment which user prefers and predict

intentions. It must have the ability to identify needed context and properly interpret it. In this way it is possible to obtain intelligent systems that can adapt to situations that were not programmed, perceive changes in the environment and interact with users.

Context is a very well-known term that does not have accurately specified definition. Day and Abowd [12] defined context as:

Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the user-application interaction, including the user and applications themselves.

There is a special approach to the development of different types of information systems, which takes into account all the characteristics of the environment and is designed to compensate for the shortcomings of computer technology, which cannot accumulate knowledge about the context. This approach is called Context-Aware Computing (CAC).

Context-dependent systems belong to the category of Ubiquitous computing. The main source of information for context-dependent systems are:

- Location, location of the user,
- social environment (crowd, family, friends, colleagues), who is in proximity to the user
- technological environment (PCs, phones, tablets, servers and etc., Internet access)
- physical environment (weather, pressure, light intensity, noise, noise level).

Modern mobile devices have a wide range of communication skills (wireless computer network, mobile network) and sensors. Depending on the functional possibilities provided by the smartphone can be used in these sensors: accelerometer, barometer, ambient light sensor, magnetometer, a thermometer, a sensor measuring the noise level sensor, measuring the quality of mobile communication and others. Universal devices (mobile phones) provide a lot of options collecting context necessary for optimizing human-computer interaction (HCI) system.

2.1 Aspect-Oriented Approach

Aspect-oriented programming (AOP) approach is a paradigm whose main goal is increasing modularity. Usually the code of applications can be separated into logical sections. These logical sections are called aspects. AOP supports separating development of any aspects option, resulting in a better code. In OOP designs, we use classes to describe only instances and their attributes. That is the representation for data only. Therefore, the best way to do things in aspect way approach is to add these additional pieces of information to the same representation (class). This is called Rich Entity Aspect/Audit Design (READ) [2,3]. Above any instance, attribute or method you can place annotation containing additional information. By this procedure, we can add all required aspects. The key concept for AOP is Join Point Model (JPM). This model allows to define the dynamic structure for cross-cutting concerns independently of the object-oriented hierarchy. JPM defines several important terms:

- Join point is a place in the program where it is possible to apply cross-cutting concerns. It can be calling methods, initialization of classes.
- Advice is expanding code that we use to extend the existing model. Advice can use the Join Point.
- Pointcut is called the place in which it is necessary to apply cross-cutting concerns. This is a selection of Join points, that Advice will be applied.
- Aspect combines Advice and Pointcut, it determines logic inserted into and a place where it applies.
- Weaving is a process integration of aspects into application specific places so that combine aspects and hierarchical structure of the application.

2.2 Operation systems for mobile devices

Within all operation systems (OS) for a mobile device, the Android is the most expanded as shown in Figure 1. Because of that, applications targeted for Android are very desirable. As you can see from Figure 1, the ratio of applications targeted for the Android OS for mobile device is steadily growing up. This is the reason why developers cannot omit the Android in their analysis.

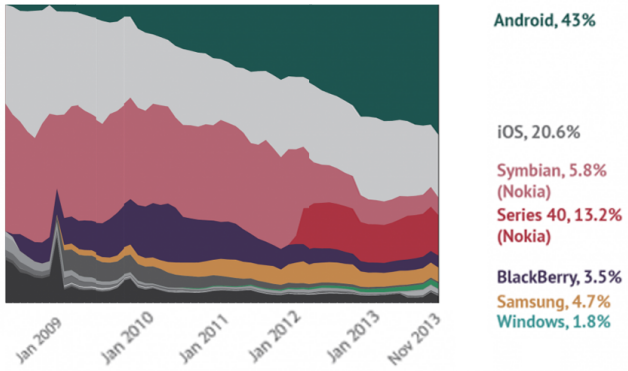

Fig. 1. Distribution of all OS for mobiles (adopted from [11])

2.3 Android sensors

Android supports sensors of the following three categories:

- Motion sensors (motion sensors)

These sensors measure the acceleration forces and rotational forces about three axes. This category includes accelerometers, gravity sensors, gyroscopes and rotary vector sensors.

- Sensors environment (Environmental sensors)

These sensors measure different environmental parameters such as temperature, pressure, light and moisture. This category includes barometers, thermometers and photometers.

● Position Sensors (Position sensors)

These sensors measure the physical position of the device. This category includes orientation sensor and magnetometer.

3. Related Work

Approach in the articles [2,3] is not the only way how to generate User Interfaces (UI) from the model. The topic of UI generated from domain objects is mentioned in [6,7]. The framework is called Meta-widget and it is based on Model driven development (MDD). The user just creates objects and puts them to Meta-widget's framework. The UI is generated according to the model. Meta-widget supports a lot of technologies from Android, Google Web Toolkit (GWT), HTML 5 (POH5), JavaScript to JSF and JSP. Meta-widget works in three basic steps. First, Meta-widget comes with a UI component native to your existing front-end. Second, Meta-widget inspects, either statically or in the run-time, your existing back-end architecture. Third, Meta-widget creates native UI subcomponents matched to the back-end. In articles [2,3], the other aspects were added based on annotations. Meta-widget adds this information based on existing back-end of any applications.

Model driven development (MDD) is based on the idea that the model should be primary centralized place for all information. This model is then compiled or transformed by another way into the deployed application code. The benefits are reduction of information in application and concentration of the structure of information into one place. The disadvantages can be adaptation and evolution management [8]. This approach does not go well with OOP, because we need to maintain the interconnection between the models with the back-end of the application. There exist another tools, how to describe an additional information. These tools are called in the MDD the Domain-Specific Languages (DSL). Sometimes, they are informally called mini-languages, because they describe the additional information inside the other language. There are a wide variety of DSL.

Generative programming (GP) is a specific type of a programming that generates the source code from domain-specific code. The goal is to improve productivity of developer, make the way between application code and domain model, support reuse, adaptation, and simplify management of components [13].

Currently there is only one framework for Android [9] which is aimed at creating adaptive style interface AOP. Android framework [9] does not use the context of the sensors, so the development of this branch provides a lot of options for searching and finding interesting solutions.

4. Design of Aspect-Oriented Framework

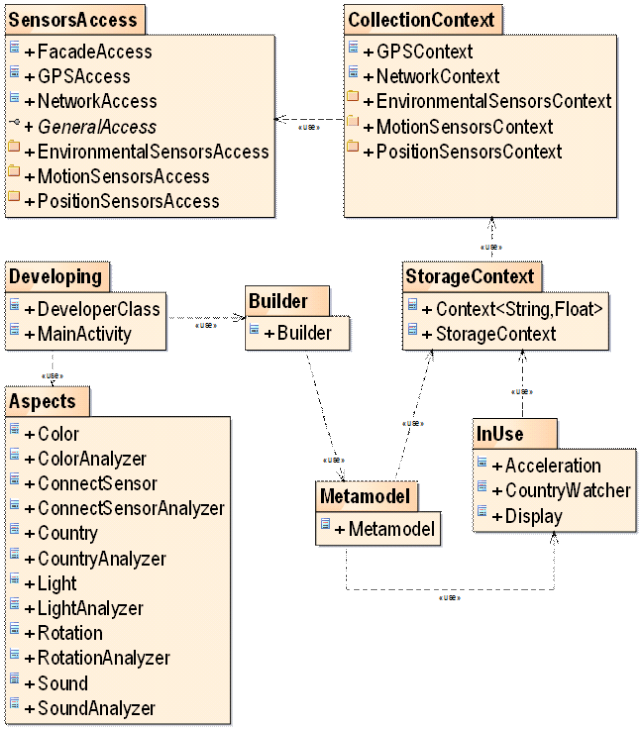

Fig. 2. Package diagram

Figure 2 illustrates modularity framework and describes the entire system. Package "Package Access Sensors" contains the classes that are designed to be connected to different types of sensors. The design pattern Facade were used to simplified interface of framework. Facade allows access to any sensor through the use of a single class FacadeAccess.

Package "Package Collection Context" is responsible for obtaining the context of a single sensor (e.g. TYPE_GYROSCOPE). SensorEvent class represents an event sensor and holds the following information: sensor type, time intervals, data accuracy and sensor data. SensorEvent includes public field of type float called SensorEvent.values. The current values for each sensor are stored there.

Package "Package Storage Context" contains all application context during runtime. StorageContext class is based on the data structure HashMap. Context <String, Float> contains a single context of the sensors. Storage Context has as key sensor name. Each individual key will contain all the context that belongs to it. Context is stored as corresponding to a specific key.

Package "Package In Use" belongs to the logical layer includes classes that implements specific functionality such as calculating the speed of movement, finding the state using GPS. Other extensions of functionality are possible to make that developers need.

Package "Package Aspects" implements aspects that support aspect oriented programming style.

Package "Developing Package" is designed for developers. Similarly, developers can use the package "In Use" if you will need to design a logical function which does not have this framework. The difference is that the core of application should be placed here and logical modul should be placed in "In Use".

Package "Package Builder" is used to start the framework. The class Builder reads all data and annotations, which were used by the programmer, it will initialize necessary data, retrieves and stores the context that is used to adapt the interface.

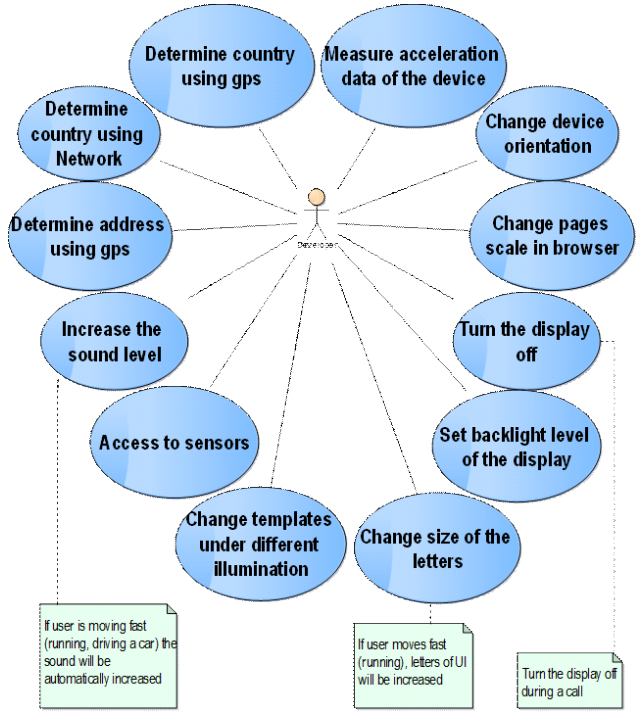

Fig. 3. Use case diagram

Chart from figure 3 reflects the relationship between the developer and functions supported by framework. Diagram shows developers what features of the framework are available for him.

5. Framework vs Conventional approach

The advantage of the framework is that provides a simple principle of access to the sensors, the developer does not need to deal with how to get the sensor. Just know from which sensor needs to obtain information and annotation used @Sensor (SENSOR_TYPE). And it will have access to the data of selected sensor.

Another advantage is that the framework provides several ready solutions for the adaptation of the user interface. For example, the use of GPS sensor, it is possible to define the state where the user is at this moment and automatically fill in the text field "Country: _____" in any window that is part of the interface.

The main advantage is the reduction in the line of code. The code is readable and easy extensible because of aspect oriented approach and the scalability of the code. Future developers do not follow strict rules and use the very methods that provides the framework. Developer needs to introduce a new aspect if he wants to expand the functionality of the end application. The ease access to context-based annotations may help to propose any particular logic functionality, for example calculate the acceleration of the aircraft or count the heartbeat and then generate a UI using its own calculations.

6. Conclusion and Future Work

This paper contains background of aspect-oriented approach, and describes the design of the new framework for an Android application development. Our framework solves interesting use cases that aims for adaptive development. The application that reacts on enviroment based on context information are good for end user.

In future work we want to test our framework on test application and use it in user tests. In the future work we will focus on extending the framework with collaborative use cases.

Acknowledgements

Research described in the paper was supervised by doc. Ing. Karel Richta, CSc., FEE CTU in Prague and supported by the Czech Student Grant Agency under grant No. SGS15/210/OHK3/3T/13.

References

[1] Czarnecki, K. and Eisenecker, U. W.: Generative programming: methods, tools, and appli-cations. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (2000)

[2] Černý, T., Donahoo, M. J., and Song, E.: Towards effective adaptive user interfaces de-sign. In Proceedings of the 2013 Research in Adaptive and Convergent Systems (RACS '13). ACM, New York, NY, USA, 373-380. DOI=10.1145/2513228.2513278, http://doi.acm.org/10.1145/2513228.2513278 (2013)

[3] Černý, T., Čemus, K., Donahoo, M. J., and Song, E.: Aspect-driven, Data-reflective and Context-aware User Interfaces Design. In: Applied Computing Review, Vol. 13, Issue 4, ACM, New York, NY, USA, 53-65. ISSN 559-6915, http://www.sigapp.org/acr/Issues/V13.4/ACR-13-4-2013.pdf (2013)

[4] Černý, T. and Song, E.: UML-based enhanced rich form generation. In: Proceedings of the 2011 ACM Symposium on Research in Applied Computation (RACS '11). ACM, New York, NY, USA, 192-199. DOI=10.1145/2103380.2103420, http://doi.acm.org/10.1145/2103380.2103420 (2011)

[5] Forman, I. R. and Forman, N.: Java Reflection in Action (In Action series). Manning Pub-lications Co., Greenwich, CT, USA (2004)

[6] Kennard, R. and Leaney, J.: Towards general purpose architecture for UI generation. Journal of Systems and Software, 83(10) http: / / metawidget . sourceforge . net / media / downloads / Towards a General Purpose Architecture for UI Generation.pdf (2010) 1896-1906

[7] Kennard, R. and Robert, S.: Application of software mining to automatic user interface generation. In SoMeT'08. http: / / metawidget . sourceforge . net / media / downloads / Application of Software Mining to Automatic User Interface Generation.pdf (2008) 244 - 254

[8] Morin, B., Barais, O., Jezequel, J.-M., Fleurey, F. and Solberg. A.: Models@run.time to support dynamic adaptation. Computer, 42(10) (Oct. 2009) 44-51

[9] Šebek, J.: Aspect-oriented user interface design for Android applications, diploma thesis. Department of Computer Science, CTU FEE, Prague (2014)

[10] Open Systems: DBMS [online]. Moscow: Publishing house "Open systems", 2012, 2012(3) [cit. 2016-03-20]. ISSN 1028-7493. Dostupné z: www.osmag.ru

[11] Šebek, J. - Richta, K.: Aspect-oriented User Interface Design for Android Applications. In DATESO 2015. Prague: MATFYSPRESS, 2015, p. 121-130. ISSN 1613-0073.

[12] G. Abowd, A. Dey, P. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in Handheld and Ubiquitous Computing, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1999, vol. 1707, pp. 304–307.

[13] Introduction To Android Mobile Operating System. Android Development, Tutorials [online]. August 1, 2011 [cit. 2014-04-29]. http://www.blogsaays.com/tutorial-part1-introduction-android-mobile-operating-system (2011)

About Authors...

Jiří ŠEBEK was born in Prague in 1989. He received his bachelor's and master's degree from the CTU in Prague, FEE in 2012 and 2014 respectively. Since autumn 2014, he has been studying the branch of doctoral studies called Information Science and Computer Engineering at the Department of Computer Science on CTU in Prague, FEE.