

Duna-kavics

A Dunaújvárosi Főiskola online folyóirata 2015. III. évfolyam III. szám

Műszaki-, Informatikai és Társadalomtudományok

ROLAND CSEH-JÓZSEF

KATONA

Design a Bill Monitor application based on Android Operation System

BARÁTH ARTUR

A közegészségügyi informatikai rendszerek biztonságát veszélyeztető humán forrású veszélyek

Kovács Lajos
Élhető világ

Magyar András
Pentele – Párt – Politika



DunaKavics

A Dunaújvárosi Főiskola online folyóirata 2015. III. évfolyam III. szám

Műszaki-, Informatikai és Társadalomtudományok

MEGJELENIK ÉVENTE 12 ALKALOMMAL

SZERKESZTŐBIZOTTSÁG

András István, Király Zoltán, Kukorelli Katalin, Palotás Béla,
Rajcsányi-Molnár Mónika,

SZERKESZTŐSÉG

Ladányi Gábor (Műszaki)
Nagy Bálint (Informatika és matematika)
Szakács István (Gazdaság és társadalom)
Klucsik Gábor (technikai szerkesztő)

Felelős szerkesztő Németh István

Tördelés Duma Attila

Szerkesztőség és a kiadó címe 2400 Dunaújváros, Táncsics M. u. 1/a.

Kiadja DUF Press, a Dunaújvárosi Főiskola kiadója

Felelős kiadó András István, rektor

A lap megjelenését támogatta TÁMOP-4.2.3-12/1/KONV-2012-0051

„Tudományos eredmények elismerése és disszeminációja
a Dunaújvárosi Főiskolán”.

<http://dunakavics.duf.hu>

ISSN 2064-5007

Tartalom

ROLAND CSEH-JÓZSEF KATONA

Design a Bill Monitor application based on Android Operation System

5

BARÁTH ARTUR

*A közegészségügyi informatikai rendszerek biztonságát
veszélyeztető humán forrású veszélyek*

17

KOVÁCS LAJOS

Élhető világ

29

MAGYAR ANDRÁS

Pentele – Párt - Politika

37

Galéria

(Bakos Miklós fotói)

43

Design a Bill Monitor application based on Android Operation System

Abstract: The article presents an application design which run on the Android Operating System and requirement analysis and specification has been provided. The designed application will lead to the users a full range of bill about fees of the current period or balance. The design is one of the most important life cycle of the software development, which can be accomplished by several technological tools, such as the UML (Unified Modelling Language) which provided numerous models. In this article, different design patterns and results of the designed application will be demonstrated by mainly diagrams which are provided by UML.

Keywords: Android, software development, design, UML.

Összefoglalás: A cikk egy olyan Android operációs rendszerre szánt alkalmazás tervezését mutatja be, amelynek követelmény-feltárása és specifikációja már megtörtént. A tervezendő alkalmazás egy teljeskörű „szám-lát” vezet a felhasználók számára az aktuális időszak forgalmi díjáról vagy éppen telefon egyenlegéről. A tervezés a szoftverfejlesztési életciklus egyik legjelentősegteljesebb állomása, amelyet több technológiai eszközzel is megvalósíthatunk, mint például az UML (Unified Modelling Language – Egységes Modellező Nyelv) által szolgáltatott modellek. A cikkben jórészt ezen modellező nyelv által biztosított diagramok segítségével ismertetésre kerülnek a megvalósítandó alkalmazás különböző terv mintáik és eredményeik.

Kulcsszavak: Android, szoftverfejlesztés, tervezés, UML.

**PentaTrade Kft*

E-mail: cseh.roland@pentatrade.hu

***Dunaújvárosi Főiskola,
Informatikai Intézet*

E-mail: katonaj@mail.duf.hu

[1] Attila Kővári, (2009): Hybrid Current Control Algorithm for Voltage Source Inverters, Proceedings First IEEE Eastern European Conference on the Engineering of Computer-Based Systems, Novi Sad, Pp. 65–70.

[2] Attila Kővári, (2010): Real-Time Modeling of an Electro-hydraulic Servo System, Computational Intelligence in Engineering, Springer, Pp. 301–311.

[3] Whitten, J. L.–Lonnie D. Bentley, Kevin C. Dittman. (2003): *Systems Analysis and Design Methods*.

[4] McConnell, S. (2004): *"7: Lifecycle Planning"*. Rapid Development. Redmond, Washington: Microsoft Press.

[5] Blanchard, B. S.,–Fabrycky, W. J. (2006): *Systems engineering and analysis*. New Jersey: Prentice. Hall.

Introduction

The modelling is one of the most important stage of engineering development [1, 2] and software development also. The software development life cycle (SDLC), which may be in all period of this cycle, but it is more typical during the design of an application. Emphasis must be placed by software developers during a careful design, as a good plan is able to help them disclose errors, so developers can improve these bugs earlier. Because of the earliest errors correction, considerable cost savings can be achieved. One of the most widely used modelling language (UML – Unified Modeling Language) during design method, which provides several diagrams to use. Important to know that there are not good or wrong diagrams, their selection depends on the type of the designer, moreover the software type can also determine their uses. [3, 4, 5]

Figure 1: Software Development Life Cycles (SDLC)



Source: <http://www.atksolutions.com/articles/images/>

The article presents an Android OS-based software design, which will be lead a phone bill, based on specification of the application. Having finished the design, the application implementation, testing and migration can be started. In planning

stage, the most important aspect was the efficiency and flawless operation. It was tried to plan so screen design that ensure the easiest possible overview in the in the implemented application.

The result of the plan, diagrams and models will be the core of the implemented application. [6, 7]

Specification of software requirements [6]

The most important features that has to be solved:

- Following contract or prepaid phone card bills;
- Following the billing period correctly (each subscription has a date when the period is restarted);
- Handling free minutes and free text messages;
- Managing free phone numbers and text messages numbers;
- Within the network and outside the network minutes / text messages fee, free minutes and text messages tracking;
- Handling periods (business, outside business and free hours);
- Calculating with tax of the phone;
- Service for monthly limit and balance limit;
- Invoice report function (in case of subscription);
- Handling the balance charging (in case of prepaid);
- Managing other monthly expenses;
- Nice and simple HOLO theme user interface based on the Android Design Guidelines;
- Various HOLO themes;
- Consistent appearance in case of Android 2.3 or even 4.x with ActionBarSherlock management;
- Advanced options understandably;
- Widgets and/or notification on the status which is updated automatically after finishing a phone call or sending a text message;
- Roaming notice;
- Logging the program events correctly.

[6] R. Cseh, –J. Katona, (2015): *Requirement analysis and specification of a Bill Monitor application based on Android Operating System*. Dunakavics, Vol III., Issue II., Pp. 35–44.

[7] Ralph, P.–Wand, Y. (2009): A Proposal for a Formal Definition of the Design Concept. In: Lyytinen, K.–Loucopoulos, P.–Mylopoulos, J.–Robinson, W., (Eds.): *Design Requirements Engineering: A Ten-Year Perspective*. Springer-Verlag, Pp. 103–136.

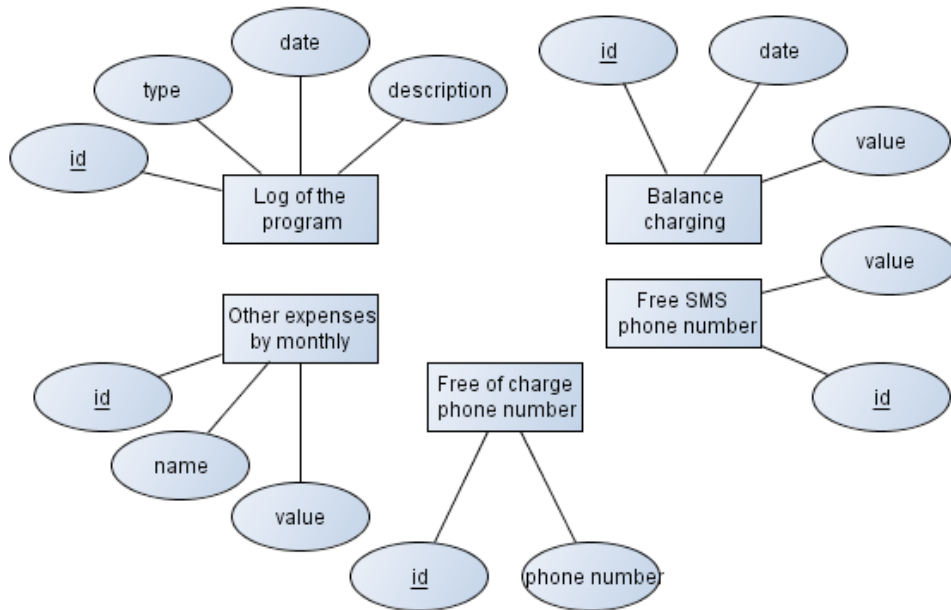
Design the database

It is recommended to use database because of the appropriate data storage and the features which are described in the specification. The following functions require to create a database:

- Free of charge phone numbers and free text messages number management;
- Other expenses by monthly;
- Balance charging;
- Event log.

Each feature will be separated into different data table and relationship between them is not necessary (Figure 2.).

Figure 2. Entity-relationship model (E/R model) in a big way.



The Android Operating System provides that any application are able to create a file-based database, which can be accessed only by itself. It is advisable to implement the above designed database with this in mind.

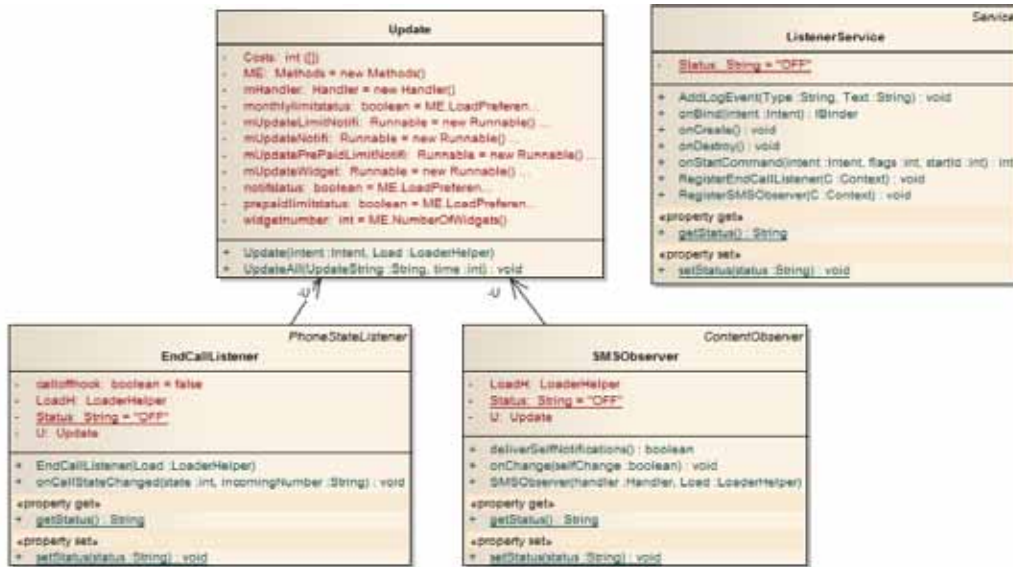
UML diagrams

The design of the application used several UML diagrams, such as static, dynamic, use case and which describe the system architecture and components. In this article only a few will be presented because of the page limits.

CLASS DIAGRAM

The class diagram (*Figure 3.*) illustrate the Listener package contents above. The application services and any other classes which are necessary for the proper functioning of the service can be found in this package. As appears in the diagram, there are two factual observer classes, the EndCallListener and the SMSObserver. These classes are served separately for calls and SMS monitoring. They contain 1–1 static variable which are named Status. They are intended to serve that indicate to the Service that there are still available and they need to be started or not. Update class contains all the function which are responsible to refresh the data and display the new data via a widget or notification. The ListenerService class is an implementation for services which are run by Android Operating System. The observers will be registered in this class which will be connected to the service of process and not to the main process of the application. With this solution will be achieved that after closing the application the two observer classes make their tasks as long as the service is run. If the service is restarted by some reason, it will check whether the listeners still live and act accordingly.

Figure 3. Class diagram.

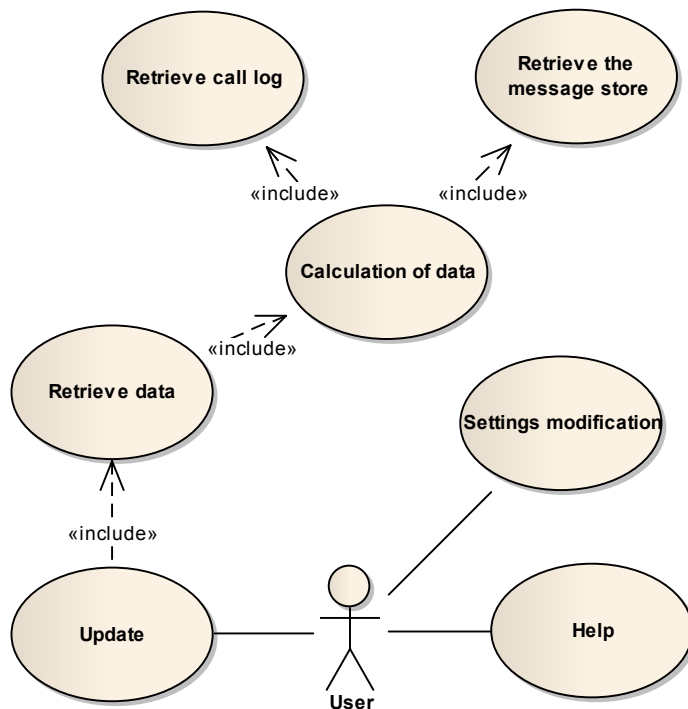


USE CASE DIAGRAM

This Use Case diagram (Figure 4.) represents the basic usage of the application. If a user starts the application, there are three possibilities:

- Update the data (it is started automatically);
- Change the settings;
- Read the help section;

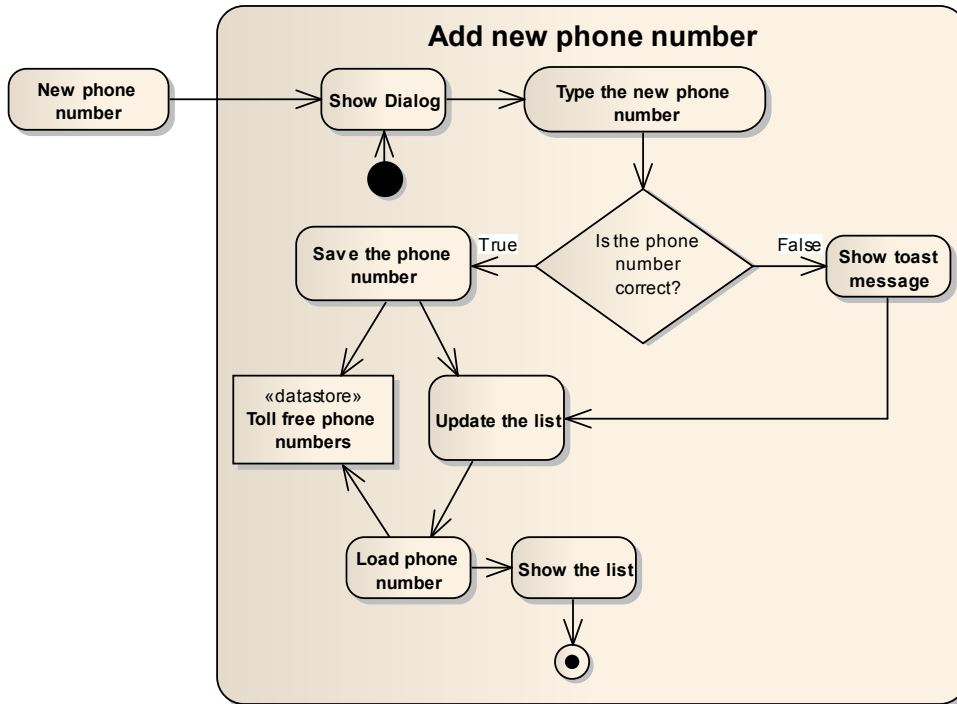
The graph clearly shows that the updated data come from event log of the device and the SMS repositories, which are calculated according to the settings and updated on the user's display.

Figure 4. Use Case diagram.

ACTIVITY DIAGRAM

The activity diagram (Figure 5.) essentially presents a particular activity of a process. In this case we can talk about a simple process, which shows an adding process of a new phone number to the database (toll-free or free text messages). The whole process will be initiated by the action itself, which event is a new phone number recording. Having started this process, the entered data correcting will be fully checked. If it is not correct (not a phone number), a toast notification will be sent to the users and the list will be updated and re-displayed without saving. If the phone number is correct, it will be saved, updated and displayed.

Figure 5. Activity diagram.



Design the Graphical User Interface

It was very important to follow the design guidelines, which was published by the creator of Android during the design of the user interface. It and the default theme of Android also allow to develop an efficient and attractive application. These guidelines can be found in the following locations: <http://developer.android.com/design/index.html>. The main interface of the application must be understandable, informative, unified and nice, which is able to display a variety of information thanks to the scroll authority without having to navigate from there.

PANORAMA OF THE MAIN SCREEN

Panorama pictures of the main screen shows the Figure 6. This main interface displays any data, which are calculated by the designed application and which are affected the phone bill. The first page is a summary page which displays the most important information (current period, the sum of the current bill, the remaining balance). Information about fee of the month can be found on the second page. The main settings of the application is found on the third tab, which is connected to the user's account (e.g. based on minutes or seconds). Statistics of the call and the text message are displayed on the last two pages.

Figure 6. Panorama of the main screen.



SCREEN DESIGN OF FUNCTIONS

The logs of the user interface wanted be designed as simple as possible. The reason of this was the transparency. The activity logs viewer has to be very clean and usable such easily. For instance, a drop-down menu allows user to choose the right category. The simplest way to solve this problem if a text and a scrolling list only appears on the screen, where the details of the currently selected category can be seen. The text shows the current period and the windows title bar, which is also a drop-down menu, shows the current category. All items of the data in the current category could be visible by an individually designed list item in the list. Within the settings can be found parts which will be implemented by individually designed screens. One of these, the screen, which is used to treat the toll free phone numbers.

The screen design of the log function shows on *Figure 7.* and the screen design of the treatment of the toll free phone numbers illustrates on *Figure 8.*

Figure 7. Screen design of the log function.



Figure 8. Screen design of the treatment of the toll free phone numbers.

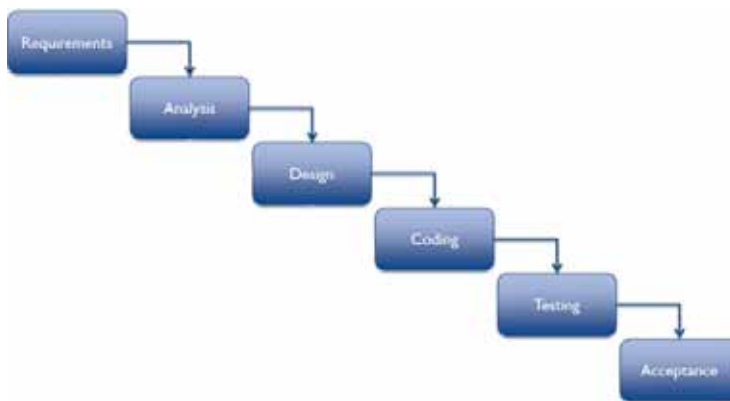


This screen is very similar to the logs screen, which have been presented above. The difference is that a category does not have to be selected so there is not a drop-down menu. Instead of a drop-down menu, there are two button on the top screen with which new data can be inserted easily into the displayed list. If the list does not even contain an element, the user's attention will be drawn by a text. In addition, a pop-up window (dialog) helps users to enter a phone number. The data edit and the delete option must be provided as well. (Figure 8.) It uses the selector, which built-in editing mode. (Screen 2). The editing is will be created by a dialog as well.

Conclusion

The article describes some of design elements of an application which want to be implemented. Before designing this application, it was specified after analysing the requirements. [6] Most of the possible factors were taken into account during the design process. With a good plan, the implementation stage could be considerably facilitated and several errors can be corrected at the beginning of the software development life cycle. These errors would be able to cause much bigger problem and could increase the costs in the future. Many software development models exist, such as waterfall (Figure9.) model, spiral model (Figure 10) or rapid application development (Figure 11.).

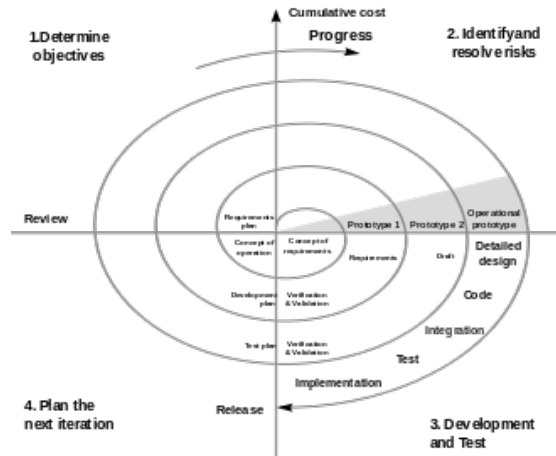
Figure 9. Waterfall development.



Source: <http://www.covalentmarketing.com/wp-content/uploads/2013/02/>

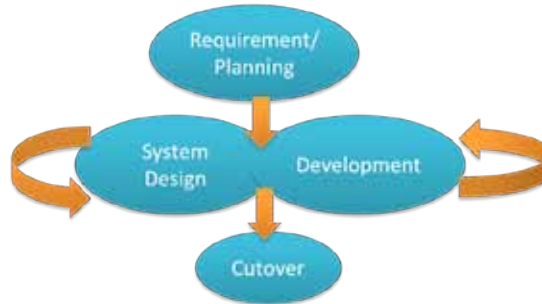
[6] Cseh, R. –Katona, J. R. (2015): Requirement analysis and specification of a Bill Monitor application based on Android Operating System. Dunakavics, Vol III., Issue II., Pp. 35–44.

Figure 10. Spiral development.



Source: <http://users.dickinson.edu/~wahlst/491/lectures/lec02/se.html>

Figure 11. Rapid application development (RAP).



Source: <http://javatechig.com/others/rapid-application-development-model>

However, these models solely are almost never used in the industry. There are a kind of “blend” of them. Most models creators agrees that the quality of the design defined the quality of the final application. The article shows and explains static, dynamic and use cases diagrams. Furthermore, screen designs are important that prognosticate the program usability.