

USER ORIENTED REENGINEERING FOR MOBILE APPLICATIONS

Ion IVAN

PhD, University Professor, Department of Economic Informatics
Faculty of Cybernetics, Statistics and Economic Informatics
University of Economics Studies, Bucharest, Romania

E-mail: ionivan@ase.ro



Alin ZAMFIROIU

PhD candidate,
University of Economics Studies, Bucharest, Romania

E-mail: zamfiroiu@ici.ro



Mihai Liviu DESPA

PhD candidate,
University of Economics Studies, Bucharest, Romania

E-mail: mihai.despa@yahoo.com



ABSTRACT:

Characteristics of mobile applications in terms of development, usage and investment process are highlighted; analysed from the point of view of the investor, target group and return on investment. Mobile applications are designed to record user behaviour and databases assembled during this process represents raw material for the reengineering process. Recording user behaviour is showcased in the context on an actual application. Indicator of user behaviour are defined and submitted for debate. Indicators are validated by using data from a real life application. A model that facilitates the user oriented reengineering process is build. An efficient way of determining the optimal period for triggering the reengineering process is submitted.

Key words: mobile application, reengineering, user, user behaviour.

1. CHARACTERISTICS OF MOBILE APPLICATIONS

Mobile applications are sequences of code that run on mobile devices with the purpose of solving user's issues.

Unlike traditional applications, mobile applications have specific features that distinguish them from traditional applications or web applications. Mobile application characteristics are grouped into three categories:

- *development characteristics* of mobile applications, regarded from the viewpoint of the person who writes the code and implements mobile applications to user; characteristics of mobile applications in terms of the development process:
 - the development cycle of mobile applications has a structure focused solely on the target group; the architecture design starts from establishing the target group;
 - portability of mobile applications is limited and therefore development must be carried out taking into account this limitation;
 - the application's interface should be similar to other applications' interfaces that users already use;
 - proper display of controllers in order for the user to have a natural interaction with the application;
 - providing information in accordance with the screen size of the mobile device;
 - power management for hardware resources;
 - testing has a different weight in the development process than in traditional applications and is achieved both through emulators and real life devices;
 - obsolescence of mobile applications is much higher than for computer applications or web applications;
 - mobile application reengineering is very dynamic insuring permanence even when migrating to a new version or when modifying applications; new applications developed following the reengineering process should uphold the precedence principle, the vocabulary used by previous application and to be closer to the user than the previous application.
- *usability characteristics* of mobile applications, viewed from the standpoint of the person who has a problem to solve and uses mobile application for handling the issue; characteristics of mobile applications in the usage process:
 - the plethora of mobile devices and their diversity makes for the application to be used in a similar way regardless of the device's, size or hardware configuration;
 - the small size of the device is considered a major problem for mobile applications because the size influence the usability degree of any device;
 - the diversity of the target group requires that the application be used by all persons included in the group, such that usage is homogenous regardless of user's personality, culture, work field, habits, hobbies or lifestyle;
 - customising mobile applications in terms of usability increases user satisfaction by creating the feeling that the application was designed especially to suit his needs;
 - free applications can be accessed by any user and thus the target group increases considerably; users who's profile does not match the application's intended target have to be reject in an appropriate way in order to accommodate users that represent the real target of the mobile application;
 - mobile applications offer natural opportunities for collecting instant digital pictures and videos because of their immediate availability to users [OLLE07].
- *characteristics of the investment process*, regarded from the viewpoint of stakeholders that provide financial resources in order to compensate programmers for their work

and from the perspective of operational costs generated by the development of the mobile application; investors aim to recover their capital and generate profit by exploiting the mobile application; characteristics of mobile applications derived from the investment process as follows:

- technologies that are employed in mobile application development are constantly and rapidly changing; investors are compelled to acquire new technologies right from their beta development stage so that when technologies hit the market mobile application that exploit the new architecture are already available; this way users are familiar with the new products right from the start;
- all mobile applications are viewed as short-term investments or long-term investments, so that the investor can recover its investment by selling the applications through online stores, like Google Play, App Store, BlackBerry World or by freely distributing it and recovering the initial investment from paid advertising; advertising banners can reduce usability though as the most downloaded mobile applications on Google Play, App Store, BlackBerry World do not display any advertisements [KHSU12].

There are open access mobile applications, based on users belonging to a dynamic community, developers seeking to include as many people in the target group and the application being developed in order to meet user requirements.

The application is designed so that users can carry out interactions that define access to specific resources that make up the *MP* set of processes:

$$MP = \{ P_1, P_2, \dots, P_j, \dots, P_{nrp} \} \quad (1)$$

where:

P_j – is the j process;

nrp – number of process types.

For mobile applications the following list of processes has been established:

- application setup;
- user login;
- account activation in order to validate existing database records;
- access account information;
- accessing information provided by the application;
- providing a solution for the issue tackled by the application;
- accessing the result or solution provided by the application.

Each process contains a set of activities, MAP_i , that defines the entire endeavour. The MAP_i set is defined by:

$$MAP_j = \{ AP_1^j, AP_2^j, \dots, AP_k^j, \dots, AP_{nrp_j}^j \}, j = \overline{1 - nrp}, \quad (2)$$

where:

AP_k^j - is the j activity from the k process;

nrp_j – number of activities of the j process.

For the AP_k^j activity a specific time frame is defined thus establishing:

$L_S^{AP_k^j}$ – the starting point of the interval that acts as a time frame for the AP_k^j activity;

$L_F^{AP_k^j}$ – the starting point of the interval that acts as a time frame for the AP_k^j activity.

with:

$$L_S^{AP_k^j} < L_F^{AP_k^j}, j = \overline{1-nrp}, k = \overline{1-nrap_j} \quad (3)$$

Within a certain process or within the entire application, activities are clustered into the following categories:

- activities performed throughout the process or throughout the mobile application's life cycle, Figure 1.

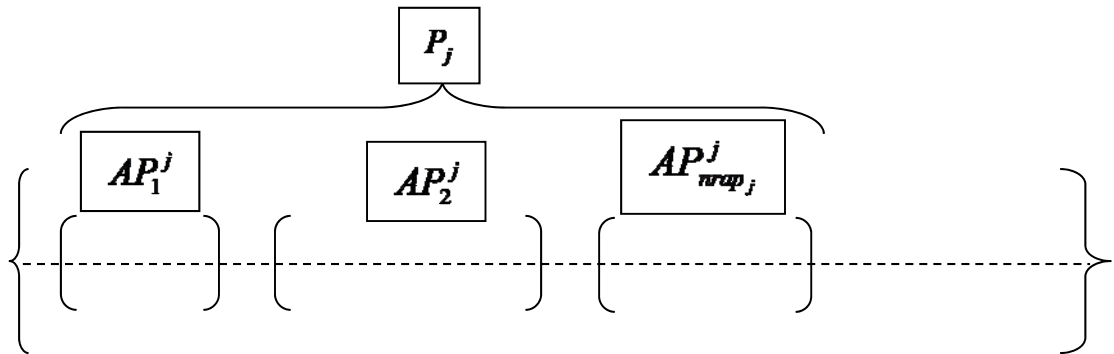


Figure 1: Activity performed during the application's life cycle

For the activities performed throughout the process there are two scenarios that have to comply with the following inequalities:

- a.** activities performed throughout the application's life cycle; all the activities are performed within the predefined application interval; AP is considered a generic activity taking place throughout the application thus:

$$L_S^{AP} \leq L_S^{AP_k^j} < L_F^{AP_k^j} \leq L_F^{AP}, j = \overline{1-nrp}, k = \overline{1-nrap_j} \quad (4)$$

- b.** activities performed throughout the entire process; all the activities are performed within the predefined process interval; AP^j is an activity from the j process, that complies with the following inequality:

$$L_S^{AP^j} \leq L_S^{AP_k^j} < L_F^{AP_k^j} \leq L_F^{AP^j}, j = \overline{1-nrp}, k = \overline{1-nrap_j} \quad (5)$$

- disjoint activities, time frames do not overlap, Figure 2.

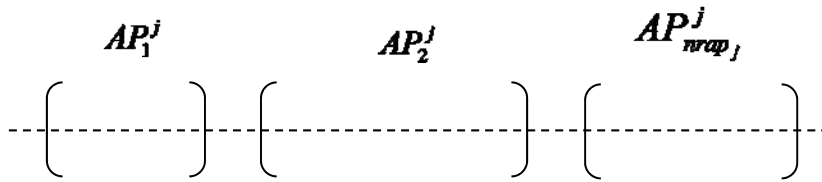


Figure 2: Disjoint activities

For these types of activities the start and completion points of an activity do not overlap, thereby:

$$\begin{aligned}
 &L_S^{AP_1^1} < L_F^{AP_1^1} < L_S^{AP_2^1} < L_F^{AP_2^1} < \dots < L_S^{AP_{nrp_1}^1} < L_F^{AP_{nrp_1}^1} < \\
 &< L_S^{AP_1^2} < L_F^{AP_1^2} < L_S^{AP_2^2} < L_F^{AP_2^2} < \dots < L_S^{AP_{nrp_2}^2} < L_F^{AP_{nrp_2}^2} < \\
 &< \dots < L_S^{AP_1^{nrp}} < L_F^{AP_1^{nrp}} < L_S^{AP_2^{nrp}} < L_F^{AP_2^{nrp}} < \dots < L_S^{AP_{nrp}^{nrp}} < L_F^{AP_{nrp}^{nrp}}
 \end{aligned} \tag{6}$$

or

$$L_S^{AP_k^j} < L_F^{AP_k^j} < L_S^{AP_{k+1}^j} < L_F^{AP_{k+1}^j}, j = \overline{1 - nrp}, k = \overline{1 - nrp_j} \tag{7}$$

• **adjoint activities**, in contrast to disjoint activities the time intervals overlap, thereby giving permission for two or more activities to take place simultaneously, Figure 3.

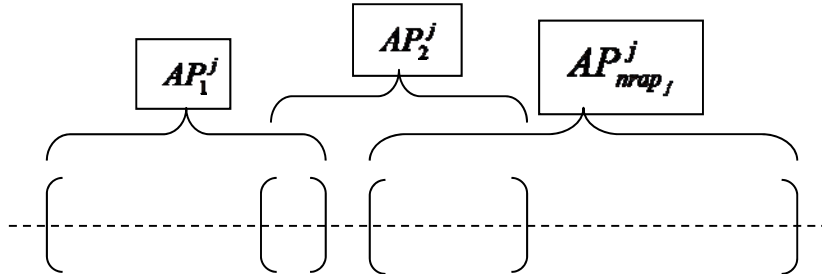


Figure 3: Adjoint activities

The interval of the AP_k^j activity is calculated, $D^{AP_k^j}$, using the following:

$$D^{AP_k^j} = L_F^{AP_k^j} - L_S^{AP_k^j} \tag{8}$$

$D^{AP_k^j}$ is the interval in which the activity is performed. Each activity has a maximum

interval that allows for proper implementation $DM^{AP_k^j}$, where:

$$DM^{AP_k^j} < D^{AP_k^j} \tag{9}$$

The interval is expressed using time units:

- seconds for activities with short intervals;
- hours for activities with medium intervals;

- days for activities with large intervals;
- months for activities with very large intervals;
- years for activities with very large intervals.

In the scenario of a process with disjoint activities:

$$L_S^{AP_{k+1}^j} = L_F^{AP_k^j}, j = \overline{1 - nrp}, k = \overline{1 - nrap_j} \quad (10)$$

Then the process's interval, D_{P_j} , is:

$$D_{P_j} = \sum_{j=1}^{nrap_j} D^{AP_k^j} \quad (11)$$

In the scenario of an application that addresses a well-defined target group, consider the set of application users, UA, defined by:

$$UA = \{UA_1, UA_2, \dots, UA_i, \dots, UA_{nru}\} \quad (12)$$

where:

UA_i – the i user of the application;

nru – the number of users within the group MUA.

For each user a specific interval is set for a certain activity. Thus the following moments are defined:

$MSU_i^{AP_k^j}$ – is the starting point of the AP_k^j activity for the i user;

$MFU_i^{AP_k^j}$ – is the completion point of the AP_k^j activity for the i user ;

where:

$$MSU_i^{AP_k^j} < MFU_i^{AP_k^j}, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru} \quad (13)$$

For each AP_k^j activity the starting point and completion point defined for a certain user fits in a predefined interval for that specific activity thus:

$$L_S^{AP_k^j} \leq MSU_i^{AP_k^j} < MFU_i^{AP_k^j} \leq L_F^{AP_k^j}, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru} \quad (14)$$

For each user the interval, $D_{UA_i}^{AP_k^j}$, for completing a certain activity is determined using the formula:

$$DU_i^{AP_k^j} = MFU_i^{AP_k^j} - MSU_i^{AP_k^j}, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru} \quad (15)$$

Depending on the moments when the user performs a task another two limits are calculated:

- starting limit, $DFS_i^{AP_k^j}$ defined as a difference between the activity's start

moment for the i user, $MSU_i^{AP_k^j}$ and the activity's start limit, $L_S^{AP_k^j}$, thus:

$$DFS_i^{AP_k^j} = MSU_i^{AP_k^j} - L_S^{AP_k^j}, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru} \quad (16)$$

- b. completion limit, $DFF_i^{AP_k^j}$, defined as a difference between the activity's completion limit and the activity's actual completion moment for the i user, thus:

$$DFF_i^{AP_k^j} = L_F^{AP_k^j} - MFU_i^{AP_k^j}, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru} \quad (17)$$

Based on the above defined limits the following are defined:

- **user's yield** for a certain activity $RU_i^{AP_k^j}$, which is calculated using the formula:

$$RU_i^{AP_k^j} = 1 - \frac{DU_i^{AP_k^j}}{DM^{AP_k^j}}, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru} \quad (18)$$

- **overall user yield of the user**, $RTU_i^{AP_k^j}$ which is calculated for all the activities performed using the application:

$$RTU_i^{AP_k^j} = 1 - \frac{\sum_{j=1}^{nrp} \sum_{k=1}^{nrap_j} \frac{DU_i^{AP_k^j}}{DM^{AP_k^j}}}{\sum_{j=1}^{nrp} nrap_j}, i = \overline{1 - nru} \quad (19)$$

Following the evaluation of the interval required to complete an activity by a user, starting limit, completion limit and yield, the following user categories are defined:

- **organised** defined by the fact that they access the application on a regular basis and perform activities with a tight starting limit;
- **directed towards critical moments** very close to the completion limit, thus users in this category have very little difference from baseline to completion;
- **random behaviour**, sometimes within limits, sometimes right on the edge; users in this category disregarding patterns or any imposed rules;
- **efficient** characterized by a very high yield and overall yield;
- **inefficient** characterized by a low yield and overall yield.

Table 1 presents the conditions required for each user category.

Table 1: User category

Category	Condition
Organised users	$DFS_i^{AP_k^j} \Rightarrow 0$
Users directed towards critical moments	$DFF_i^{AP_k^j} \Rightarrow 0$ and $RU_i^{AP_k^j} \Rightarrow 1$ or $RU_i^{AP_k^j} \Rightarrow 0$
Random behaviour users	$\exists DFF_i^{AP_k^j} \Rightarrow 0$ and $\exists DFF_i^{AP_k^j} \Rightarrow 1$
Efficient users	$RU_i^{AP_k^j} \Rightarrow 1$
Inefficient users	$RU_i^{AP_k^j} \Rightarrow 0$

Thus user sets divided by category are obtained:

- organised users' set, UO is determined based on the following formula:

$$UO = \{UA_i \mid DFS_i^{AP_k^j} \Rightarrow 0, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru}\} \quad (20)$$

- users directed towards critical moments set, UC is determined based on the following formula:

$$UC = \{UA_i \mid DFF_i^{AP_k^j} \Rightarrow 0, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru}\} \quad (21)$$

- random behaviour users' set, UCA is determined by analysing several activities and is determined based on the following formula:

$$UCA = \{UA_i \mid \exists DFF_i^{AP_k^j} \Rightarrow 0, \exists DFS_i^{AP_k^j} \Rightarrow 0, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru}\} \quad (22)$$

- efficient users' set, UEF is determined based on the following formula:

$$UEF = \{UA_i \mid RU_i^{AP_k^j} \Rightarrow 1, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru}\} \quad (22)$$

- inefficient users' set, UNE is determined based on the following formula:

$$UNE = \{UA_i \mid RU_i^{AP_k^j} \Rightarrow 0, j = \overline{1 - nrp}, k = \overline{1 - nrap_j}, i = \overline{1 - nru}\} \quad (23)$$

For users with random behaviour no condition is defined because they do not follow a predictable path and instead perform random activities.

2. USER BEHAVIOUR DATABASE

Asking the users to explicitly rate each application they use can provide an accurate picture of their application needs and requirements. This approach, however, requires manual labeling and not many people are willing to or can remember to consistently provide their input [1]. In order to determine user behavior both application and database are built on automatic acquisition of information on the actions performed by the user.

When building the database it is required to create a table that stores data about:

- actions performed by users;
- the moment when the action was performed;
- user that performed the action.

The application has to include processes dedicated to storing user related information into the database. The actions performed by the user are saved with the purpose of conducting further analysis related to user behaviour.

The moment when the action was performed is stored because it is necessary in tracking usability patterns. Actual moments of performing an action can be classified as follows:

- variation criteria:
 - fixed moments are actual time pointers that are identical for all users like starting limit or completion limit of an activity;

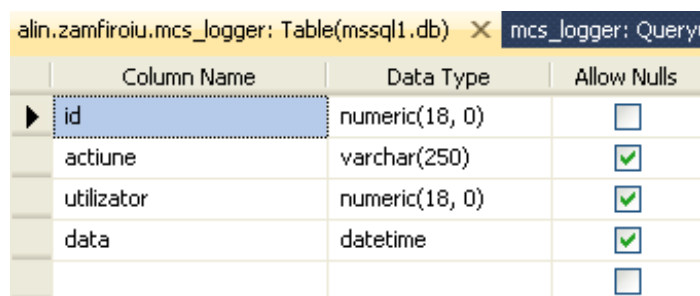
- variable moments are time points determined by each user's behaviour like the start moment or completion moment of an activity.
- affiliation criteria:
 - the moment belongs to a sub interval which defines an activity and is included between the starting limit and completion limit of an activity;
 - the moment is comprised of the entire process and is not proprietary to a specific activity; for a scenario where the activity is scheduled for the entire length of the process the moment is comprised of the activity interval;
 - the moment represents the entire life cycle of the application; the moment is not proprietary to a certain activity except in the case in which the activity is schedule to take place during the application's entire life cycle;
- moment presentation criteria
 - as an integer that represents the position on an interval where 0 is the starting moment of the application and X is the number that defines the end of a certain activity represented in years, months, days, hours, minutes and seconds;
 - as a calendar data and time option *dd.mm.yyyy hh:mm:ss*.

The user performing the action is stored in database in a table as a link or foreign key to the users table.

The information stored in the database is used to determine the following:

- user behaviour;
- user's efficiency in using the application;
- application's efficiency in documenting user behaviour;
- determining the category of a user is based on the moments when the user is accessing the application;

The database contains a table designed to store actions performed by the user when interacting with the application, activities performed and the exact moment when activities were performed. Figure 4 illustrates the structure of the *mcs_logger* table.



Column Name	Data Type	Allow Nulls
id	numeric(18, 0)	<input type="checkbox"/>
actiune	varchar(250)	<input checked="" type="checkbox"/>
utilizator	numeric(18, 0)	<input checked="" type="checkbox"/>
data	datetime	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Figure 4: The structure of the *mcs_logger* table

Data acquisition is done automatically while the user is interacting with the application without disturbing him with questionnaires and polls.

3. USER BEHAVIOUR WITHIN THE M-LEARNING APPLICATION

MCSAM application was designed for automatic acquisition of information on user behaviour.

The MP set of processes is defined, $MP = \{\text{account activation, establishing profile, accessing courses, accessing project homework, self-testing}\}$. For each process the MAP_i activity set is defined within the process. Table 2 illustrates activities within each process.

Table 2: Activities of a process within the MCSAM application

Process	Activities
Account activation	<ol style="list-style-type: none"> 1. Login 2. Fill-in inputs 3. Save form
Establishing profile	<ol style="list-style-type: none"> 1. Answer question No. 1 2. Answer question No. 2 3. 4. Answer question No. n 5. Save answers 6. Save test result
Accessing courses	<ol style="list-style-type: none"> 1. Accessing menu 2. Accessing submenus 3. Accessing a course 4. Reading the course
Accessing project homework	<ol style="list-style-type: none"> 1. Accessing menu 2. Accessing submenu seminar 3. Accessing project homework 4. Reading project results
Self-testing	<ol style="list-style-type: none"> 1. Accessing application menu 2. Accessing self-testing submenu 3. Check for test existence 4. Specify test details 5. Take test 1 6. Take test 2 7. Take test 3 8. Take test 4 9. Take test 5 10. Access test results

Within certain processes, activities' time span is very short and therefore only one activity will be considered for the entire process length.

In order to analyse the information stored in the database a table for defining the application is created. The table list activities as entries and also stores starting limits and completion limits.

Table 3: Activities starting limit and completion limit

Activity	Starting limit	Completion limit
Account activation	15.10.2013 00:00:00	10.11.2013 23:59:00
Establishing profile	15.10.2013 00:00:00	10.11.2013 23:59:00
Accessing courses	15.10.2013 00:00:00	10.01.2014 23:59:59
Accessing homework	15.10.2013 00:00:00	10.01.2014 23:59:59
Take test 1	12.11.2013 07:00:00	19.11.2013 23:59:00
Take test 2	19.01.2014 07:00:00	22.01.2014 15:00:00
Take test 3	19.01.2014 07:00:00	22.01.2014 15:00:00
Take test 4	19.01.2014 07:00:00	22.01.2014 15:00:00
Take test 5	19.01.2014 07:00:00	22.01.2014 15:00:00

The *Account activation* activity is analysed and the *ac* acronyms attributed to it. This activity consists of filling the fields with relevant account information, saving the information and changing the original password.

Account activation was achieved during the interval defined by the starting limit and completion limit for that specific activity:

$$L_S^{ac} = 15.10.2013 \quad 07:00:00$$

$$L_F^{ac} = 10.11.2013 \quad 23:59:00$$

where:

$$L_S^{ac} < L_F^{ac} \tag{24}$$

Using the same approach the *Establishing profile* activity, *sp*:

$$L_S^{sp} = 15.10.2013 \quad 07:00:00$$

$$L_F^{sp} = 10.11.2013 \quad 23:59:00$$

where:

$$L_S^{sp} < L_F^{sp} \tag{25}$$

The two activities are disjoint and consecutive. After *Account activation* activity is completed the *Establishing profile* activity starts.

The length of the interval associated to the *Account activation* activity, D^{ac} is determined using the following formula:

$$D^{ac} = L_F^{ac} - L_S^{ac} \tag{26}$$

$$D^{ac} = 10.11.2013 \quad 23:59:00 - 15.10.2013 \quad 07:00:00$$

$$D^{ac} \cong 27 \text{ days} \cong 641 \text{ hours} \cong 38.459 \text{ min}$$

The length of the interval associated to the *Establishing profile* activity is the same as the length of the interval associated to the *Account activation* activity:

$$D^{sp} = D^{ac} \cong 27 \text{ days} \cong 641 \text{ hours} \cong 38.459 \text{ min} \tag{27}$$

For the UA set with $nru = 70$ for each user the application has to store in the database:

- start moment for *Account activation* activity;
- completion moment for *Account activation* activity;
- completion moment for *Establishing profile*.

Information extracted from the database is shown in Table 4.

Table 4: Moments for *Account activation* and *Establishing profile* activities

User	Start moment <i>Account activation</i> MSU_i^{ac}	Completion moment <i>Account activation</i> MFU_i^{ac}	Completion moment <i>Establishing profile</i> MFU_i^{sp}
UA1	20.10.2013 10:11	20.10.2013 10:12	20.10.2013 10:19
UA2	25.10.2013 09:57	25.10.2013 09:57	25.10.2013 10:00
UA3	15.10.2013 15:59	15.10.2013 16:01	15.10.2013 16:02
...
UA69	23.10.2013 15:48	23.10.2013 15:49	23.10.2013 15:52
UA70	07.11.2013 10:23	07.11.2013 10:25	07.11.2013 10:26

Because activities are consecutive the start moment for the *Establishing profile* activity is the same as the completion moment for the *Account activation* activity:

$$MSU_i^{sp} = MFU_i^{ac}, i = \overline{1-70} \quad (28)$$

For all users following restrictions are mandatory:

- the start moment for *Account activation* has to be previous to the activity's completion moment:

$$MSU_i^{ac} < MFU_i^{ac}, i = \overline{1-70}; \quad (29)$$

- the start moment for *Establishing profile* has to be previous to the activity's completion moment:

$$MSU_i^{sp} < MFU_i^{sp}, i = \overline{1-70}; \quad (30)$$

- the two moments concerning *Account activation* have to reside within the interval defined by the activity's start limit and completion limit:

$$L_S^{ac} \leq MSU_i^{ac} < MFU_i^{ac} \leq L_F^{ac} \quad (31)$$

where:

$$i = \overline{1-70};$$

- the two moments concerning *Establishing profile* have to reside within the interval defined by the activity's start limit and completion limit:

$$L_S^{sp} \leq MSU_i^{sp} < MFU_i^{sp} \leq L_F^{sp} \quad (32)$$

where:

$$i = \overline{1-70};$$

Because the two activities are disjoint and consecutive the following statements have to be true:

$$L_S^{ac} = L_S^{sp}, L_F^{ac} = L_F^{sp} \text{ and } MSU_i^{sp} = MFU_i^{ac}, i = \overline{1-70}, \quad (33)$$

By noting the two activities' start limit L_S^{comuna} and the completion limit L_F^{comuna} :

$$L_S^{comuna} \leq MSU_i^{ac} < MFU_i^{ac} = MSU_i^{sp} < MFU_i^{sp} \leq L_F^{comuna}, i = \overline{1-70} \quad (34)$$

Based on data selected form the database:

- length of the *Account activation* activity interval, $D_{UA_i}^{ac}$;
- difference between *Account activation* start moment and *Account activation* start limit, DFS_i^{ac} ;
- difference between the *Account activation* completion limit and *Establishing profile* completion moment, DFF_i^{ac} ;
- user yield in executing the *Account activation* activity, RU_i^{ac} ; it is determined in comparison with the maximum time spent on the *Account activation* activity;
- the length of the *Establishing profile* activity interval, $D_{UA_i}^{sp}$;
- difference between *Establishing profile* start moment and *Establishing profile* start limit, DFS_i^{sp} ;
- difference between the *Establishing profile* completion limit and *Establishing profile* completion moment, DFF_i^{sp} ;
- user yield in executing the *Establishing profile* activity, RU_i^{sp} ; it is determined in comparison with the maximum time spent on the *Establishing profile* activity.

The above defined information is systematized in Table 5.

Table 5: Account activation and Establishing profile data

User	$D_{UA_i}^{ac}$ (sec)	DFS_i^{ac} (hours)	DFF_i^{ac} (hours)	RU_i^{ac}	$D_{UA_i}^{sp}$ (sec)	DFS_i^{sp} (hours)	DFF_i^{sp} (hours)	RU_i^{sp}
UA1	48	123,19	517,78	0,87	431	123,21	517,66	0,5
UA2	26	242,96	398,02	0,93	166	242,97	397,97	0,81
UA3	85	9	631,96	0,77	101	9,02	631,94	0,88
...
UA69	42	200,81	440,16	0,89	168	200,82	440,11	0,8
UA70	120	555,4	85,55	0,67	20	555,43	85,55	0,98

By analysing information provided in Table 5 the user categories for the *Account activation* activity are define as follows:

$$\bullet UO^{ac} = \{UA_i \mid DFS_i^{ac} < 24, i = \overline{1-70}\} = \{UA3, UA8, UA10, UA18, UA28, UA37, UA38, UA43, UA48, UA49, UA56, UA58, UA67\}; \quad (35)$$

$$\bullet UC^{ac} = \{UA_i \mid DFF_i^{ac} < 24, i = \overline{1-70}\} = \{UA46, UA54, UA60, UA68\}; \quad (36)$$

$$\bullet UEF^{ac} = \{UA_i \mid RU_i^{ac} > 0.75, i = \overline{1-70}\} = \{UA1, UA2, UA3, UA6, UA7, UA8, UA9, UA10, UA11, UA12, UA15, UA16, UA17, UA20, UA21, UA22, UA24, UA25, UA26, UA27, UA28, UA30, UA31, UA32, UA33, UA34, UA35, UA36, UA37, UA39, UA40, UA45, UA46, UA47, UA48, UA49, UA50, UA51, UA52, UA53, UA55, UA56, UA57, UA58, UA59, UA61, UA62, UA63, UA64, UA68, UA69\}; \quad (37)$$

$$\bullet UNE^{ac} = \{UA_i \mid RU_i^{ac} < 0.3, i = \overline{1-70}\} = \{UA5, UA41\}. \quad (38)$$

Using the same method the categories for the *Establishing profile* activity are defined:

$$\bullet UO^{sp} = \{UA_i \mid DFS_i^{sp} < 24, i = \overline{1-70}\} = \{UA3, UA8, UA10, UA18, UA28, UA37, UA38, UA43, UA48, UA49, UA56, UA58, UA67\}; \quad (39)$$

$$\bullet UC^{sp} = \{UA_i \mid DFF_i^{sp} < 24, i = \overline{1-70}\} = \{UA46, UA54, UA60, UA68\}; \quad (40)$$

$$\bullet UEF^{sp} = \{UA_i \mid RU_i^{sp} > 0.75, i = \overline{1-70}\} = \{UA2, UA3, UA5, UA6, UA8, UA13, UA15, UA21, UA23, UA24, UA30, UA32, UA34, UA35, UA37, UA39, UA44, UA45, UA46, UA51, UA53, UA55, UA57, UA58, UA59, UA60, UA61, UA64, UA66, UA67, UA68, UA69, UA70\}; \quad (41)$$

$$\bullet UNE^{sp} = \{UA_i \mid RU_i^{sp} < 0.3, i = \overline{1-70}\} = \{UA7, UA18, UA22, UA28, UA42, UA43, UA62\}. \quad (42)$$

The new activity that is subject to analysis is *Take test 1*, represented using the acronym *st1*.

Users can take Test No. 1 within the interval defined by the start limit and completion limit:

$$L_S^{st1} = 12.11.2013 \quad 07:00:00$$

$$L_F^{st1} = 19.11.2013 \quad 23:59:00$$

where:

$$L_S^{st1} < L_F^{st1} \quad (43)$$

The *Take test 1* activity is disjoint from other test taking activities thus:

$$L_S^{st1} < L_F^{st1} < L_S^{st2} < L_F^{st2} < L_S^{st3} < L_F^{st3} < L_S^{st4} < L_F^{st4} < L_S^{st5} < L_F^{st5} \quad (44)$$

The length of the *Take test 1* activity's interval, D^{st1} is determined by using the formula:

$$D^{st1} = L_F^{st1} - L_S^{st1} \quad (45)$$

$$D^{st1} = 19.11.2013 \quad 23:59:00 - 12.11.2013 \quad 07:00:00$$

$$D^{st1} = 7days = 185hours = 11.100min$$

The maximum length of the *Take test 1* activity is 15 minutes, $DM^{st1} = 15\text{min}$

The user set, $nru = 70$, is defined and for every user the following information is extracted from the data base:

- check for the existence of *Test 1* moment, MVU_i^{st1} ;
- start moment of *Take test 1* activity, MSU_i^{st1} ;
- completion moment of *Take test 1* activity, MFU_i^{st1} .

Information extracted from the database is shown in Table 6.

Table 6: Moments for *Take test 1* activity

User	MVU_i^{st1}	MSU_i^{st1}	MFU_i^{st1}
UA1	16.11.13 9:19	16.11.13 9:20	16.11.13 9:22
UA2	17.11.13 7:27	17.11.13 7:27	17.11.13 7:30
UA3	17.11.13 21:46	17.11.13 21:47	17.11.13 21:52
...
UA69	19.11.13 13:30	19.11.13 13:30	19.11.13 13:41
UA70	15.10.13 7:00	10.11.13 23:59	7.11.13 10:25

For all users the following restriction apply:

- the start moment of the *Take test 1* activity is prior to the completion moment of the *Take test 1* activity:

$$MSU_i^{st1} < MFU_i^{st1}, i = \overline{1-70}; \quad (46)$$

- the two moments concerning *Take test 1* have to reside within the interval defined by the activity's start limit and completion limit:

$$L_S^{st1} \leq MSU_i^{st1} < MFU_i^{st1} \leq L_F^{st1} \text{ with: } i = \overline{1-70} \quad (47)$$

Based on data extracted from the database for each user specific indicators are calculated. Indicators regard the length of the interval within the activity was performed, the difference between the start moment and the start limit of the activity, the difference between the completion limit and the completion moment of the activity and user's yield in performing the *Take test 1* activity. Indicators are presented in Table 7.

Table 7: Difference between moments of the *Take test 1* activity

User	Activity's interval length $D_{UA_i}^{st1}$ (sec)	Difference from activity's start limit DFS_i^{st1} (hours)	Difference from activity's completion limit DFF_i^{st1} (hours)	User's yield RU_i^{st1}
UA1	179	98,33	86,6	0,8
UA2	190	120,46	64,48	0,79
UA3	331	134,79	50,11	0,63
...
UA69	630	174,51	10,3	0,3
UA70	332	180,56	4,33	0,63

By analysing information provided in Table 7 the user categories for the *Take test 1* activity are define as follows:

$$\bullet UO^{st1} = \{UA_i \mid DFS_i^{st1} < 24, i = \overline{1-70}\} = \{UA30, UA31, UA37, UA39, UA49, UA63\}; \quad (48)$$

$$\bullet UC^{st1} = \{UA_i \mid DFF_i^{st1} < 24, i = \overline{1-70}\} = \{UA4, UA5, UA14, UA27, UA34, UA43, UA55, UA65, UA66, UA69, UA70\}; \quad (49)$$

$$\bullet UEF^{st1} = \{UA_i \mid RU_i^{st1} > 0.75, i = \overline{1-70}\} = \{UA1, UA2, UA4, UA5, UA8, UA21, UA24, UA31, UA33, UA44, UA53, UA55\}; \quad (50)$$

$$\bullet UNE^{st1} = \{UA_i \mid RU_i^{st1} < 0.3, i = \overline{1-70}\} = \{UA6, UA7, UA11, UA16, UA19, UA22, UA23, UA25, UA28, UA29, UA36, UA41, UA42, UA48, UA58, UA62, UA66, UA69\}. \quad (51)$$

Based on user sets defined according to category for the three activities that have been subject to analysis global user categories are constructed. In order to determined global user categories the above mention categories are intersected and also users with random yield are identified. As $UO^{ac} = UO^{sp}$ and $UC^{ac} = UC^{sp}$ in order to construct the global set only UO^{ac} will be used. Thus:

$$\bullet UO = UO^{ac} \cap UO^{st1} = \{UA37, UA49\}; \quad (51)$$

$$\bullet UC = UC^{ac} \cap UC^{st1} = \{\}; \quad (52)$$

$$\bullet UEF = UEF^{ac} \cap UEF^{sp} \cap UC^{st1} = \{UA2, UA8, UA21, UA24, UA53, UA55\}; \quad (53)$$

$$\bullet UNE = UNE^{ac} \cap UNE^{sp} \cap UC^{st1} = \{\}. \quad (54)$$

$$\bullet UCA = \{UA_i \mid \exists DFF_i^m < 24, \exists DFS_i^n < 24, i = \overline{1-nru}, m \neq n\},$$

$m, n \in \{ \text{Account activation, Establishing profile, Taking test 1} \}.$

(55)

$UCA = \{UA43\}$

If $UA43$ $DFF_i^{st1} < 24$ and $DFS_i^{ac} < 24$, random behaviour is established.

4. ESTABLISHING THE REQUIREMENTS FOR USER-ORIENTED REENGINEERING

Establishing user behaviour leads to formalizing usability patterns. Usability patterns are used to fundamentally change an application in order to better suit the user's needs. A radical change in an application requires a reengineering process. According to [2] Reengineering information systems represents improving intermediate results and providing a significant leap forward in terms of quality by optimizing resource usage, reducing maintenance costs, and achieving objectives with more accuracy. A reengineering process

takes as input an existing computer application, basic, from which a new computer application with superior performance parameters is obtained.

Reengineering aims to transform applications by combining existing elements with new entities. The application that results from the reengineering process incorporates superior quality standards and a lower risk of error occurrence.

Reengineering costs will impact the developer and the user as well:

- the developer has development, coding and implementation costs;
- the user has costs related to setting up and installing the application and also for training personnel in using the application.

Reengineering is achieved when substantial changes have to be implemented and cannot be achieved through maintenance or development of a new version of the application. The differences between reengineering and maintenance consists in the depth of the changes that need to be implemented.

Reengineering mobile applications is achieved by upholding the principles of customization and the user orientation so that application-user interaction is as natural as possible.

Reengineering occurs when:

- data acquisition must be made not only from the keyboard or from databases, created by entering data from the keyboard, but by scanning through voice recordings and by measurements with special devices;
- selecting new option is done using the mouse or arrows and not by inserting text;
- a new technology that facilitates interactions emerges;
- new features or new modules that do not support the old technology are introduced;
- tools developed by third parties that offer certain features and reengineering requires that those specific tools be included into the new application.

New technologies related to data storage or displaying information emerge. Users enter numeric data, images, movies in their shared areas and develop their own applications.

Reengineering involves radical changes in hardware, data acquisition and increased quality functionalities offered to the user.

The result of the maintenance process is only a mended application using old technology and struggling to find relevant information.

The application that results from the reengineering process is new generation product superior in every way to previous applications.

The optimal time for reengineering is when required changes are not achievable by using maintenance procedures [3]. Minor changes do not require reengineering but instead can be performed using maintenance. According to [2] reengineering is required when the software system no longer meets the objectives for which it was created.

The time for reengineering is also determined according to the volume of information collected from users on the existing application. It is important to gather a lot of information from users so that reengineering will be based on direct feedback.

The effects of the reengineering phase are the results of an extensive process.

The application prior to the reengineering process has the following disadvantages:

- long waiting time;
- the application is difficult to manage;
- poor management of hardware resources;
- high complexity;
- users are offered too only a fraction of what they require;
- interaction with the users is often ambiguous and unnatural.

All the disadvantages, through the reengineering filter, are turned into positive effects in new application:

- managing the application is easier;
- more efficient management of hardware resources;
- application becomes easier to use;
- user interface is natural and intuitive because it is built based on specifications collected from actual users of the application.

All these disadvantages turned into advantages represent the effects of reengineering a mobile application.

5. CONCLUSIONS

Mobile applications are different from other types of application due to particular usability characteristics, specific investment process and distinct development and testing practices. User interaction with the mobile application is documented in order to establish usability patterns. Data is automatically collected from users and stored in a database. Analysing user behaviour information is the starting point for defining reengineering processes. Reengineering is required when the software system no longer meets the objectives for which it was created. The application that results from the reengineering process incorporates superior quality standards and a lower risk of error occurrence. Return on investment is only achieved if the target group perceives an increase in satisfaction by using the new application:

- increasing the frequency of visits by old users
- increasing the number of users by registering new users

Increased return on investment is obtained by optimizing resource usage, reducing maintenance costs, and achieving objectives with more accuracy.

REFERENCES

1. Yan, B., Chen G., **AppJoy: Personalized Mobile Application Discovery**, Proceedings of the 9th international conference on Mobile systems, applications, and services, 28 Jun. – 01 July 2011, Bethesda, MD, USA, Publisher: ACM, 2011, pg. 113-126.
2. Tomozei, C., **Reingineria aplicațiilor distribuite**, PhD Thesis, 2012, Bucharest
3. Andone, I. - **Enterprise Reengineering and Expert Systems Challenges for Managers of 21 Century Organizations** - Informatica Economică, nr. 3(31)/2004, pg. 7-15, <http://revistaie.ase.ro/content/31/andone.pdf>.
4. Boja, C., Popa M., Nițescu I., **Characteristics for Software Optimization Projects**,

Informatica Economică, nr. XII, vol. 1, pp. 46-51, 2008,
<http://revistaie.ase.ro/content/45/7%20-%20catalin%20boja.pdf>

5. Roșcan P., Roșcan D., Popper M., Sava I., Păzitor T., Stelea N., Vădan M., Tudor A., Sofronie C., Roșcan P.V.C., **Software Reengineering Modeled In Concept Evolution Systems**, Analele Universității din Oradea Fascicula de Energetică, Vol. 15, 2009, <http://www.energy-cie.ro/archives/2009/p4-13.pdf>
6. Olsson T., Lehtonen M., Pavel D., Väänänen-Vainio M.K., **User-centered design of a mobile application for sharing life memories**, Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology, 10-12 Sep. 2007, Helsinki, Finland, Publisher: ACM, 2007, pp. 524-531
7. Khan A. J., Subbaraju V., Misra A., Seshan S., **Mitigating the True Cost of Advertisement- Supported Free Mobile Applications**, Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications, 28 – 29 Feb. 2012, San Diego, CA, USA, Publisher: ACM, 2012, article no. 1