Self-Healing for Mobile Applications

Ion IVAN, Cătălin BOJA, Alin ZAMFIROIU

Bucharest University of Economic Studies
ROMANIA
ionivan@ase.ro, catalin.boja@ie.ase.ro, zamfiroiu@ici.ro

Abstract: The reliability and security of a software application are two of the most important software quality characteristics because they describe the ability of the software to run without failures and to protect user data. Mobile applications concur with desktop applications in terms of rich interfaces and functionalities and are becoming one the most used type of software applications. Based on the “anytime, anywhere” paradigm, mobile applications must provide special measures to avoid failures and to preserve a high level of reliability and security because mobile operating systems provide limited or none access to administrative or system tools that will allow a user with an IT background to access temporary or persistent data. A high level of software reliability is directly influenced by a low level of failures. We there-fore describe self-healing as a required quality characteristic for mobile applications and we propose a metric for measuring it. This approach is included in the general context of mobile applications quality and the papers describes types of mobile applications, their development cycle and features that are particular to mobile applications quality.

Key-Words: mobile, application, self-healing, quality, software metric, regeneration

1. Mobile applications and devices

Mobile applications plays an important role in today’s digital era because mobile devices are becoming more powerful in terms of processing power, more rich in functionalities and more affordable for the general public. They concur with desktop applications in terms of rich interfaces and functionalities and are becoming one the most used type of software applications and they have an advantage, their mobility.

Mobile applications are software applications developed to run on mobile devices that have resource restrictions, both hardware and software. Based on [1] and [2] a mobile device is a portable small, hand-held computing device with independent data processing capacity, having a display screen with touch input and/or a miniature keyboard, which can communicate with other IT systems by a wireless connection.

In [3] and [2] mobile devices are divided into distinct categories based on performance, usage, communication capabilities, operating system, input modes, software feature:

- basic terminals or simple phones performed basic functions such as send / receive SMS, receive / initiate call; they are not able to use any wireless data communication and multimedia characteristics are limited;
- feature phones are mobile low-end mobile devices that have additional functions over and above standard mobile phones like camera, touchscreen and Wi-Fi capabilities; because technology changes rapidly features phones may be considered the low-end smartphones;
- smartphones are high-end feature phones with advanced business functions, office applications, high-speed Internet connection, specific multimedia terminals functions, high-resolution touchscreens; they evolved from classic personal digital assistants (PDAs);
- multimedia terminals in addition to basic functions present additional functions such a powerful camera, large storage media, audio and video player;
- fashion terminals includes performance functions like multimedia terminals and has a special design;
- mobile Internet devices represent an intermediary class of mobile devices that are more than a smartphone and less than a tablet PC in terms of features; they are used mostly for Web
browsing based on a wireless network connection; modern tablets are becoming the mobile Internet devices of the moment;

- **mobile standard PCs** are mobile computers that have all the features of a personal computer; Tablet-PCs, laptops/notebooks and ultra mobile PCs (UMPC) distinguish from a classic desktop computer by size, display, weight and peripherals.

From the user point of view, we can divide them into the following categories:

- **normal user** uses the phones for basic functions like to send and receive messages, making and receiving calls;
- **gadgets-oriented user** uses the multimedia terminals and likes to have phone with many tools such as camera, music player or image editing programs;
- **fashion users** not give so much attention to functions performed by the mobile device but is more interested of the device design. This user uses the mobile phone accessories such as bracelet, watch or an accessory for clothes;
- **business user** uses the phone for basic functions and other business-type activities. The phone has text editing tools, graphics tools, statistics and conference presentations tools. Through phone network user connects to the Internet to check e-mail or communicate with business partners;
- **Internet user** relies on his mobile device to browse Internet pages for different information, to use social Internet based services and to stay connected with the network of friends, RSS feeds and other personal interests; this class of users are mostly teenagers who use their mobile device only for personal use;

Operating systems (OS) for mobile devices are divided into two categories, [4]:

- **open OS** provides an interface for programming the applications for that platform, access to kernel, other OS features and source code;
- **closed OS** do not provide interfaces for applications programming but provides support for independent technologies and mobile platforms for running and developing applications; an example of mobile platform that represents an intermediary layer between the application and OS is Java Micro Edition (J2ME).

Mobile OS features are similar to the personal computer functions, but provide additional and specific features of mobile devices in terms of hardware and software management, memory management, process management, control the input/output of device, files management, graphical interface, application programming interface, management of phone features.

In the last years the market for mobile operating systems recorded major changes moving from a monopoly of relative few platforms to a dynamic market where several major platforms compete for a bigger share. Figure 1 shows the dynamic of the top eight mobile operating systems, recorded between March 2011 and March 2012.

Applications for mobile devices have special features that make them more attractive than the desktop counterparts, but also more sensitive in the development:

- must provide the same user experience across a large diversity of terminals; this requires that the application maintain the same quality level for all types of terminals; in terms of software quality, is important to provide a high level of portability from one terminal to another; applications must adapt and provide the same user experience for any display size;
- users are numerous and heterogeneous in terms of cultural and social background, experience, age; this diversity requires the application to be user-friendly and to be easily accessed by all types of users;
applications manage personal or other sensitive data that requires high security; mobile devices are more vulnerable to theft or loosing and for that, the information used by mobile applications require a higher degree of security;

- real-time decisions based applications requires accuracy and speed in mobile applications;
- availability on a 24 hours base;
- mobile applications relies on Internet connections for data access, updating sessions and even download; almost all mobile applications are downloaded from Internet based application markets; because their input comes entirely from Internet connections, most mobile applications are in fact mobile Web applications; the special relations between mobile applications and the availability of a Internet connection is depicted by the exponential growth of mobile Internet use versus desktop one, described by figure 2; since January 2009, mobile Internet went from 0.6 % to near 9% in March 2012 [5]; this evolution is related to the evolution and availability of mobile devices, especially smartphones and to the one of wireless data communications.

Another factor used to characterize mobile applications is their type, based on their utility [6]:

- entertainment applications – games and applications which are used for entertain the free time of the users;
- social application – are applications through which users communicate with other remote users and share personal information about them;
- utilities applications – are software used for office activities and for business activity, personal managers like an agenda, in health domain or weather information. In [3] is presented the application AccuWeather;
- news applications – are used to inform the users and are available to anyone wishing to get information or new news in a particular area;
productivity applications – are very used and concepts like M-banking, M-learning, M-tourism or M-commerce are very familiar. They are used for productive actions and are specialized in that domain;

search applications – are used for search information, pictures or words to translate. All these applications are based on internet connection. The users make a photo with the device and then search information about the photographed place.

These features impose strict requirements on the quality of applications and a high quality of quality is directly related to their degree of fulfillment. In opposition, non-quality leads to catastrophic problems due to failures generated by the mobile applications.

In order to manage this complex construction of quality characteristics, features and restrictions, applications are divided into modules which have a hierarchical structure, as in figure 3. This mode of organization of the applications allows a better management of application code and modules can be developed by different programming teams and later merged to form the entire system.

![Figure 3. The hierarchical k-tier structure of the applications](image)

2. Mobile applications quality

The reliability and security of a software application are two of the most important software quality characteristics because they describe the ability of the software to run without failures and to protect user data. Based on the “anytime, anywhere” paradigm, mobile applications must provide special measures to avoid failures and to preserve a high level of reliability and security because mobile operating systems provide limited or none access to administrative or system tools that will allow a user with an IT background to access temporary or persistent data. A high level of software reliability is directly influenced by a low level of failures.

Software quality is an essential concept for software applications because it provides tools to quantify something that for most users is subjective and provides data used to evaluate and compare items in the same category.

The system of software quality characteristics can be classified based on two types of features [7]: internal and external. Internal features are those which examine the software development process and the external features analyze the software application that interacts with the end user.

In use of computers the users develops activities with long duration in time, and in use of mobile devices they develop activities with short time duration, from several seconds to several minutes, when they are in other activities.

Based on the particularities of the mobile environment, a sub set of quality characteristics are considered more important for mobile applications [8]:

- **start time** - a short starting time for the application is very important because most mobile applications are used for activities with short duration and frequency; if the start time is long, the user need to wait too much time for that application; because of the importance of a minimized start time, most mobile applications are not closed but minimized;

- **receptivity** - if the application does not respond in time to restore the user can control, and so there will be two or more tasks to be solved, which will further aggravate the application activity. It is recommended that when an order is received, the user is notified that the order is processed, because they do not give another command;

- **interface** - user interaction with the application should be minimal and the results to be those expected. The
application must not contain many modules, modules that will not be used or are used by very few users. An application must be designed for a specific target group and to solve their problems.

Software quality is a set of technical characteristics, economic and social attributes of the application. These attributes or characteristics can be grouped in three categories:

- **economic characteristics** are represented by costs (design, development, implementation, use), savings (material resources, human, financial), development efficiency and productivity, viewed the costs and savings and analyzed together to determine whether product realization;
- **social characteristics** show the impact on user product; a product may improve the routine, lifestyle, behavior of the user;
- **technical characteristics** have the largest share and are discussed extensively in the literature. They are subject to the definition of international standards by specialized organizations.

In terms of quality characteristics, the set of set of quality factors defined in the ISO 9126-1 [9] quality model can be used to define the mobile application quality:

- **functionality** is a set of functions that satisfy specific application needs in terms of appropriateness, the accuracy, interoperability, security and compliance with standards in the area;
- **reliability** describes the application ability to work correctly in difficult or special conditions. High tolerance to errors and recoverability are two of the requirements that the application in question must have;
- **usability** is defined by the effort required to use the software by users. Operability is high because the time it produces the desired result is very short;
- **efficiency** of a software application is by the correlation optimum between the consumption of resources and complexity or difficulty of the problem to solve. Response time is very important in an application under review;
- **portability** of an application is possibility to use the same application on more devices. If the user change his device with another one who have another operating system is necessary to have his application on this device too. In [10] portability actions after three scenarios:
  - mobile device to mobile device, is realized easy between the same mobile device with the same operating systems;
  - PC to mobile device, is a difficult scenario because the mobile device hardware is not so powerful like computer hardware;
  - mobile device to PC, is the same with the previous one but in this case the modifications are less.
- **maintainability** is needed effort to make the necessary changes in that software application. The software must be designed as an open product, easily updated and developed by addition, making the product easy to maintain. Considered important elements to ensure the maintainability of the application are:
  - classes, variables, methods and functions with descriptive names;
  - a change of one functionality involved the updated of this functionality name;
  - methods of a class doesn't have a large number of lines of code, usually a maximum matching on screen;
  - each method and function is documented within a phrase or two;
  - if a method overrides another method is explained why.

In [6] are determinates the weights of eleven important characteristics for mobile applications.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality of instructions sequences</td>
<td>0,05</td>
</tr>
<tr>
<td>Number of written instructions</td>
<td>0,01</td>
</tr>
<tr>
<td>Respecting the deadline</td>
<td>0,05</td>
</tr>
</tbody>
</table>
Quality of testing process 0.07
The cost to the developer 0.06
The cost for user 0.07
User satisfaction 0.19
Complexity, reliability and maintainability 0.19
The risks and vulnerabilities 0.09
The user security 0.10
Security of application 0.12

For every characteristic described in table 1 is defined an indicator to calculate the value. Using them it is calculated the aggregate indicator value for an application using the weights determinate in table 1 and figure 4.

Should be considered that the effort to write documented code, orderly and easy to understand and to change is very important to avoid problems that may arise in the future;

- portability is defined as a facility of a software application to be transferred from one medium to another, running on multiple platforms. The application must be easily installed on multiple types of computing machines and can coexist with other programs installed on that machine;

- complexity is a feature on the evaluation of difficulty of developing an program be defined mathematically, linked to the algorithm itself or empirically through volumes of text, different indicators, qualitative.

If classic applications must be equipped with quality features such as reliability, maintainability, portability, as specified, online applications must be friendly, accessible and stable. In contrast to the classic applications, the mobile applications must be robust, correctly, to guide the user, non excess resources consuming, with memory to return back.

3. Self-healing

In many socio-economic fields, mobile software has allowed software developers and users to reach the “anytime, anywhere” paradigm in terms software usability. Because of the extent use of mobile devices and applications, software services are highly available despite the user location. As a result of this special feature, mobile applications are included in socio-economic dependable systems like critical infrastructures, transportation and medical systems. In these systems, high availability, security and reliability are critical characteristics which have to be satisfied all the time. In order to accomplish the requirements regarding availability the applications and the system must be able to self-heal. Based on the concept definition given by [11], [12],
[13], [14] and [15] the system is able to reconfigure by autonomously redeploying or restarting software components affected by a hardware or software crash failure. Self-Healing is the capability of discovering, diagnosing, and reacting to disruptions [16]. Accordingly, a self-repair action consists of a set of components scheduled for re-deployment and an appropriate new deployment for each of these components, [15]. Self-healing is only one component from a set of properties that defines a self-adaptive software, described in figure 5, which is able to adapt to any situation, to enhance its performance and to protect its data [14], [17].

For self-healing, the main objective is to maximize the availability, survivability, maintainability, and reliability of the system [18]. Four aspects are very important for self-healing [17]:

- monitoring, the system should be monitored always to observe the normal comportment of the application;
- adaptation or reconfiguration is the final phase of self-healing and consist in implementing the mechanism for recovery the damaged modules of the system.

A real self-healing system begun from the system design phase when is decided how is model the system with self-healing characteristics. Multiple techniques used to implement self-healing capabilities are described by existing research, each with its own advantages and disadvantages:

- the system provides an automatic self-management routine which detects if one of the services crashed at runtime; in this case, it restarts the crashed service taking into account the given deployment restrictions [15];
- the system generates an antibody when is detected an error or an attack, which is distributed to all vulnerable hosts; these hosts can verify if the attack exists and if the antibody can stop it [19];
- information of a site is copied in another place and when the site crashes, all data of this site has a safety copy and must to be able to make an optimal choice and solve the problem by reconfiguration his data [13];
- after detection of an attack or a fault, is invoked a localized recovery mechanism that seeks to recognize and prevent the specific failure in future executions of the application; is verified if the problem has been solved re-running the application against [12];
- the application is monitored continuously for failures; after the attack is selected a strategy and the system dynamically modifies the application, using binary injection, so that it is able to detect and recover from the same fault in the future [20];
- the fault software is detected by the diagnostician through the information collected by one or more monitors and repaired by the repairer through one or more effectors, figure 6 [21].
Because of the particularities of a mobile device, like limited processing power and low available resources, complex self-healing techniques are not available. Even the previous described techniques have been tested on server and desktop devices.

In [22] are presented software architectures for self-healing. For development of self-healing systems two elements are required, an automated decision when to repair the system and the infrastructure of the systems.

The proposed model of self-healing is to develop applications on modules as in [16] and when the application is attacked is damaged only few modules which can be very easy restored to initial form. In that way it can be two types of self-healing: horizontal self-healing and vertical self-healing.

Let be an application A, which has the executable source code text stored in a file E and in a file F, which is used. In time due of attacks the F file became F'. If $E \neq F'$ means that there was an attack on the application and the F' file need to be recovered through E file.

If the application is totally in an original state then the application have the self-healing characteristic. Self-healing of an application is the ability to recover from an attack and to restore to the form taken before that attack.

Figure 7 shows the structure of the two applications E and F'. For application F' was attacked and damaged a module on the level 1 of the application. The application F' is different like a structure that the standard application E. Restoring damaged module and bring the application in the form before the attack is considered property of horizontal self-healing.
Figure 8 describes the structures of another two applications. For application F1' in an attack were damaged modules from level 2. This application differs from the E1 standard application. Vertical self-healing is the application ability to restore the modules in depth. The self-healing property is specific to open applications that are built as modules interconnected.

4. Software metric for self-healing

Software metric is a mathematical model of indicators developed around an equation, which is designed to measure a characteristic of a software product, taking into account factors influencing the measured characteristic. Properties of indicators are:

- **sensitivity**, means that at small variations of the variables, there are small variations of the indicator, and the large variations of the variables, there are large variations of indicator value;
- **non-catastrophic character**, implies the absence of situations in which the variations of the variables involve the large variations of the indicator value;
- **compensatory character**, required to obtain different values for the indicator at different levels of variables values involved in calculating the indicator;
- **representative**, so that the indicator to represent the reality expressed by its value;
- **operational**, assumes that the indicator can be applied in any situation.

For self-healing characteristic is very important period the time that the application F was attacked and became F' and when F' is brought to its original state through the standard application E. In this period there were a number of ae accesses the application. Accesses to the fact that the application was attacked are not completed successfully. Thus self-healing indicator suggests I_R is defined by the formula:

\[
I_R = 1 - \frac{ae}{at},
\]

where:

- ae  – number accesses since the attack until the total regeneration of the application;
- at  – total number of accesses of the application;
- h  – number of accesses since the last moment.

\[
I_R(ae_0) = 1 - \frac{ae_0}{at}
\]

\[
ae_1 = ae_0 + h
\]

\[
I_R(ae_1) = 1 - \frac{ae_1}{at} = 1 - \frac{ae_0 + h}{at} =
\]

\[= 1 - \frac{ae_0}{at} + \frac{h}{at} = I_R(ae_0) + \frac{h}{at}
\]
Equation (4) highlights the strong relation between variations of \( I_R \) values. Thus this indicator is sensitive. For different values of \( a_e \) the indicator \( I_R \) have different values, thus this indicator have the compensatory character. IR have catastrophic character only if the \( a_t \) is equal with 0, but this can’t be associated with a real scenario. The indicator value is representative because it shows the speed at which the application is heal and the number of calls the application is minimized destroyed. Formula is applicable to any situations, so the indicator is operational.

5. Conclusions

In our days, mobile applications and devices are complex and for developers is difficult to satisfy the performance requirements required by users and by industry. An efficient approach is to develop applications using hierarchical modules. The paper presented the self-healing concept as quality characteristic of mobile applications. It has described horizontal self-healing by restoring the modules on the existing level and vertical by restoring the modules on a new level. In order to measure the efficiency of self-healing, a metric is needed to calculate self-healing of applications. It is recommended that mobile applications to be self-able to update/repair, thus avoiding the inconvenience caused by attacks on them and damage of the source code.

Acknowledgement

This work was supported from the European Social Fund through Sectored Operational Programmer Human Resources Development 2007-2013, project number POSDRU/89/1.5/S/59184, “Performance and excellence in postdoctoral research in Romanian economics science domain”.

Parts of this research have been presented at 19th International Economic Conference – IECS 2012, "The Persistence of The Global Economic Crisis: Causes, Implications, Solutions", 2012.

References


