ESTIMATING THE EFFORT OF MOBILE APPLICATION DEVELOPMENT

Laudson Silva de Souza[1] and Gibeon Soares de Aquino Jr.[1]

[1]Department of Informatics and Applied Mathematics,
Federal University of Rio Grande do Norte, Natal, Brazil
{laudyson,gibeon}@gmail.com

ABSTRACT

The rise of the use of mobile technologies in the world, such as smartphones and tablets, connected to mobile networks is changing old habits and creating new ways for the society to access information and interact with computer systems. Thus, traditional information systems are undergoing a process of adaptation to this new computing context. However, it is important to note that the characteristics of this new context are different. There are new features and, thereafter, new possibilities, as well as restrictions that did not exist before. Finally, the systems developed for this environment have different requirements and characteristics than the traditional information systems. For this reason, there is the need to reassess the current knowledge about the processes of planning and building for the development of systems in this new environment. One area in particular that demands such adaptation is software estimation. The estimation processes, in general, are based on characteristics of the systems, trying to quantify the complexity of implementing them. Hence, the main objective of this paper is to present a proposal for an estimation model for mobile applications, as well as discuss the applicability of traditional estimation models for the purpose of developing systems in the context of mobile computing. Hence, the main objective of this paper is to present an effort estimation model for mobile applications.

KEYWORDS

Software Engineering, Software Quality, Estimating Software, Systematic Review, Mobile Applications, Mobile Computing

1. INTRODUCTION

Computing is becoming increasingly present in people's lives and currently in a much more intense and accelerated way due to the rise of the use of mobile technologies in the world, such as mobile phones, smartphones and tablets, all connected to mobile networks, which are increasingly more present in many places and with better speeds. We are facing a new technological scenario that is changing old habits and creating new ways for the society to access information and interact with computer systems [1], [2] and [3].

The ITU (International Telecommunication Union) estimates that there are more than 6 (six) billion mobile clients worldwide. According to Gartner, 1.75 billion people own mobile phones with advanced capabilities; he also foresees further growth in the use of this technology in the upcoming years [4]. There is a global trend towards the increase of the number of users connected to the network via mobile devices which, consequently, will create an increasing demand for information, applications and content for such equipments. New ways to use existing information systems are emerging. In particular, systems that were once accessed via web interfaces through personal computers physically located in offices, universities or homes are providing new ways to access from mobile devices which, in turn, have different requirements and capabilities than the personal computers.

Thus, we realize that traditional information systems are undergoing a process of adaptation to this new computing context. Current developments, including the increase of the computational power of these new devices, in addition to the integration of multiple devices on a single one and lined up with the change of the users' behavior, actually create a new environment for the development of computing solutions. However, it is important to note that the characteristics of this new context are different. They present new resources and, thereafter, new possibilities [5], [6], [7] and [8], as well as introduce non-existing restrictions in conventional systems [9] and [10].

The fact is that this new technological scenario that is emerging with new requirements and restrictions requires a reevaluation of current knowledge about the processes of planning and building software systems. These new systems have different characteristics and, therefore, an area in particular that demands such adaptation is software estimation. The estimation processes, in general, are based on characteristics of the systems, trying to quantify the complexity of implementing them. For this reason, it is important to analyze the methods currently proposed for software projects estimation and evaluate their applicability to this new context of mobile computing.

Hence, the main objective of this paper is to present a proposal for an estimation model for mobile applications, as well as discuss the applicability of traditional models used in estimation of information systems for the purpose of the development of systems in the context of mobile computing. In this work, the main estimation methods that exist now will be analyzed, the specific characteristics of mobile systems will be identified and an adaptation of a estimation method that exists in this context will be proposed. For this, the paper is organized as follows: Section II discusses the main estimation methods with the purpose of identifying those that address development in this new scenario of mobile computing. Section III summarizes the main articles that discuss and point out the particular characteristics of mobile computing systems. In Section IV, the summary of the results of a survey on the characteristics of mobile development will be presented. In Section V, the problem addressed will be discussed. In Section VI, the objective of the work will be presented. Finally, in Section VII, the final considerations resulting from this research.

2. Main Estimation Methods

In order to identify how the traditional estimation methods could address the characteristics of the systems, a literature review on the main estimation methods was performed. The methods identified in the survey can be seen in Table 1.

Table 1. Main Estimation Methods.

Year	Method	Author
1979	Function Point Analysis (FPA)	Albrecht [11]
1981	COnstructive COst MOdel (COCOMO)	Barry W. Boehm 's [12]
1982	DeMarco's Bang Metrics	Tom DeMarco [13]
1986	Feature Points	Jones [14]
1988	Mark II FPA	Charles Symons [14]
1989	Data Points	Harry Sneed [15]
1990	Netherlands Software Metrics Users Association (NESMA) FPA	The Netherlands Software Metrics Users Association [16]
1990	Analytical Software Size Estimation Technique-Real-Time (ASSET-R)	Reifer [17]
1992	3-D Function Points	Whitmire [18]
1993	Use Case Points UCP	Gustav Karner [19]
1994	Object Points	Banker et al. [20]
1994	Function Points by Matson, Barret and Mellichamp	Matson, Barret e Mellichamp [21]
1997	Full Function Points (FFP)	University of Quebec in cooperation with the Software Engineering Laboratory in Applied Metrics [18]
1997	Early FPA (EFPA)	Meli, Conte et al. [22]
1998	Object Oriented Function Points – (OOFPs)	Caldiera et al. [23]
1999	Predictive Object Points – (POPs)	Teologlou [24]
1999	Common Software Measurement International Consortium (COSMIC) FFP	Common Software Measurement International Consortium (COSMIC) [25]
2000	Early & Quick COSMIC-Full Function Points (E&Q COSMIC FFP)	Meli et al. [26]
2000	Kammelar's Component Object Points	Kammelar [27]
2001	Object Oriented Method Function Points – (OOmFP)	Pastor and his colleagues [28]
2004	Finnish Software Metrics Association FSM	The Finnish Software Metrics Association (FiSMA) [29]

Table 1 displays in chronological order the main estimation methods, showing the year of creation, the name of the method and the author of it. For each of the methods identified, a summary of the description and characteristics was developed, as can be seen in the following items.

- Function Point Analysis (FPA) is a method of measurement of size in function points based on what the user notices, taking into account the implemented functionality. This method is independent of technology and was designed to estimate business information systems. Its characteristics are: estimates the functionality requested by the user, calculating its size and cost; estimates projects of development and maintenance of softwares, regardless the technology used; estimates the functionality received by the user, after its development, in order to check if its size and costs are in accordance with what was estimated [11].
- COnstructive COst Model (COCOMO) is a cost model for the process of planning and implementation of software projects. Its characteristics are: a framework for communication of business decisions between everyone involved in the project, making it possible to estimate the effort and cost and follow up on the schedule of the project [12].

- DeMarco's Bang Metrics is a method which consists in assigning an independent measure of functionality based on the structured analysis of the project and design notation. Its characteristics are: estimates the size of the software from the structured description of the components during the process of gathering requirements [13].
- Feature Points is an adaptation of Albrecht's FPA. Its characteristics are: changes the weights of the components of the original function point through an additional algorithm. This algorithm is based on a set of rules created to solve a computational problem. But due to the lack of standardization for the use of the algorithm, it has fallen into disuse [14].
- Mark II FPA is a method certified by ISO as an international standard, designed to estimate business information systems. It was created to try to remedy the shortcomings identified in the FPA method. Its characteristics are: estimates the processing of information, in which it treats a software as a set of logical operations and examines its functional size, counting the input data types, types of data entity referenced and output data elements types for each logical transaction [14].
- Data Points is a method similar to the function point. Its characteristics are: it was designed for object-oriented development and is applied based on what the user sees [15].
- Netherlands Software Metrics Users Association (NESMA) is a method certified by ISO as an international standard, designed for counting function points of the software, similar to the FPA method. Its characteristics are: External Input, External Output, External Consultation, Internal Logical File and External Interface File. The difference is that NESMA FPA gives more concrete guidelines, due to its three types of counts: detailed, estimated and indicative [16].
- Analytical Software Size Estimation Technique-Real-Time (ASSET-R) is a method designed to estimate the size of data processing in real-time systems. Its characteristics are: is similar to FPA, but takes into consideration factors, process interfaces and execution modes [17].
- 3-D Function Points is a method designed to estimate the size of real-time systems. Its characteristics are: independent of technology, is similar to the FPA but uses two new approaches, the transformations and the transitions. However, its use is impracticable in the initial design phase, for it requires a greater degree of detailing of the system [18].
- Use Case Points UCP is a method developed to measure projects in its initial phase, during the survey of requirements based on use cases of the system being developed. It was designed to measure object-oriented software projects and, to determine some factors, it takes into account the experience of the developers [19].
- Object Points is a method similar to the FPA, but it counts the objects instead of the functions. Its characteristics are: better defines the complexity adjustment factor and also adds to its count the percentage of reused code [20].
- Function Points by Matson, Barret and Mellichamp is an adaptation of the Albrecht's FPA. Its characteristics are: a linear combination of five basic components of the software, inputs, outputs, master files, interfaces and surveys [21].
- Full Function Points (FFP) is a method designed to estimate real-time and embedded systems. Its characteristics are: when the measurement is being made, he adds six more types of function in comparison to the FPA [18].
- Early FPA (EFPA) is a method similar to the FPA. Its characteristics are: estimates the functional size of the software quickly using a standard of rules different than the FPA's, which allows it to identify software objects at different levels [22].
- Object Oriented Function Points - (OOFPs) is an adaptation of the FPA, used to estimate object oriented softwares. Its characteristics are: instead of using logical files and operations like the FPA, it uses classes and methods. Its counting also takes into account the reuse of code [23].
- Predictive Object Points is a specific method for object-oriented softwares. Its characteristics are: estimates taking into account the classes, the behaviors of these classes and the effects of these behaviors on the rest of the software [24].

- Common Software Measurement International Consortium (COSMIC) FFP was designed based on the FPA, but directed to applications in real time and multi-platform softwares. Its characteristics are: estimates the development effort, evolves software quality, compares specified systems in different languages, in terms of productivity and cost maintenance, considering concepts such as functional requirements of users, software users, layers and boundaries [25].
- Early & Quick COSMIC - Full Function Points (E&Q COSMIC FFP) is an estimation method for the size of the software, which was based on COSMIC FFP. Its characteristics are: classification of the types of processes into functional process, general process or macro-process [26].
- Kammelar's Component Object Points was designed based on the FPA but was directed to object-oriented systems. Its characteristics are: takes into account two types of counting elements, users' domain elements (users' functional requirements) and system elements, including services, classes, operations and transformations [27].
- Object Oriented Method Function Points - (OOmFP) was designed based on the rules of the FPA, but was directed to object-oriented systems. Its characteristics are: the classes are considered internal logical files and concepts such as inheritance and aggregation are also relevant to the estimation [28].
- Finnish Software Metrics Association FSM, or just FiSMA, is a method of sizing the software. Its characteristics are: it is service-oriented rather than process-oriented, in other words, all services are identified to calculate the functional size of the software [29].

At first glance, one realizes that the main existing methods were not designed to consider the requirements of mobile applications. Indeed, the very creation of most of them precedes the emergence of mobile devices as we know today. This suggests that the use of these methods to estimate the effort of the development of projects involving systems or applications for mobile devices would cause a possible failure to quantify the complexity of some features and, therefore, would not produce adequate estimates.

3. Characteristics of Mobile Applications

In order to identify characteristics that are inherent to systems and mobile applications, a surveying of the characteristics of these types of software was accomplished through a systematic review. Conducting a systematic review is relevant because most searches begin with some kind of review of the literature, and a systematic review summarizes the existing work fairly, without inclinations. So the surveys were conducted according to a predefined search strategy, in which the search strategy should allow the integrity of the research to be evaluated. The planning and accomplishment of the methodology discussed were directed by Procedures for Performing Systematic Reviews [30].

3.1. Planning The Systematic Review

In the context of research questions, the following research question was formulated: “What are the characteristics of Mobile Applications?”, based on the issue about the proposed study.
Search Strategies - The search strategy was divided into three parts: sources, keywords and search strings.
Sources - the researches were directed to the following databases: ACM DL Digital Library (http://dl.acm.org/), Google Scholar (http://scholar.google.com.br/) and IEEE Xplore Digital Library (http://ieeexplore.ieee.org/Xplore/).

Keywords - the keywords were defined and based on the research question elicited previously and on their synonyms, as follows: Mobile; Applications; Computing; Features; Characteristics; Attribute; Aspect; Property; Factors; Individuality; Differential; Detail; Software; System;

Search string - based on the keywords defined previously and according to the sources to be used, the following search string was prepared: "((“Mobile Applications”) OR (“Mobile Computing”) OR (“Mobile System”) OR (“Mobile Software”)) AND (Features OR Characteristics OR Attribute OR Aspect OR Property OR Factors OR Individuality OR Differential OR Detail)”.

The results obtained through the researches made with the string search string defined previously in the three databases mentioned above were analyzed according to the following criteria:

- Inclusion Criteria: The returned result should be available in English or Portuguese; The returned result should be available in PDF or HTML format; The returned result should answer the research question directly;
- Exclusion Criteria: The returned result has already been found in previous research; The returned result has not been published in conferences, books, newspapers or magazines; The returned result has no relation to the research question; The access to the result is not available through agreements with CAPES or UFRN; The returned result was not published between 2002 and 2013;

Procedures for The Evaluation of the Articles: the articles will be analyzed considering its relation with the issues addressed in the research questions, inclusion criteria and exclusion criteria, and their respective situation will be assigned with either “Accepted” or “Rejected”. The evaluation will follow the following procedure: read the title and abstract and, should it be related with the research question, also read the whole article.

3.2. Implementation of the Systematic Review

The implementation of the systematic review was performed almost in line with its planning, except for the need to adjust the syntax of the proposed search string due to the particularities of the research bases. 234 articles were analyzed, of which 40 were selected and considered “Accepted” according to the inclusion criteria; 194 were considered “Rejected” according to the exclusion criteria. The list with all the articles can be accessed at the following address: http://www.laudson.com/sr-articles.pdf. The 40 articles that were accepted were fully read, thus performing the data extraction. All the characteristics found during this extraction phase were described in the following subsection.

3.3. Completion of Systematic Review

Given the results extracted from the systematic review, it's is possible to identify 29 kinds of characteristics in 100% of the articles evaluated and considered accepted in accordance with the inclusion criteria. However some of these are a mixture of characteristics of mobile devices and characteristics of mobile applications, such as the characteristic called “Limited Energy”, which is a characteristic of the device and not the application, however the articles that mention this type of characteristic emphasize that in the development of a mobile application, this “limitation” must be taken into account since all the mobile devices are powered by batteries, which have a limited life, depending completely on what the user operates daily. Applications requiring more hardware or software resources will consume more energy. In Figure 1, the 23 types of characteristics mentioned the most in the selected articles can be observed.

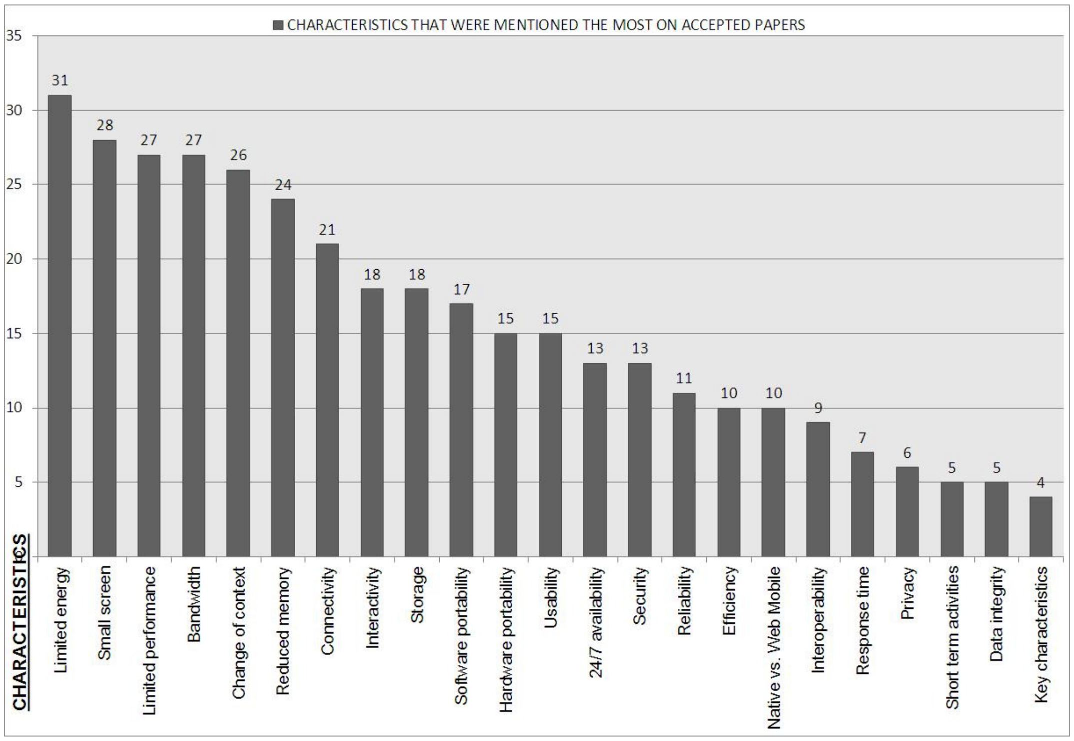


Figure 1. Characteristics that were mentioned the most on accepted papers

The other six types of characteristics identified are mentioned only three times, which are: "Complex integration of tasks in real time" and "Constant interruption of activities"; and, finally, are mentioned only once, which are: "Functional area", "Price", "Target public" and "Type of provider Type".

Following, there is a description of each characteristic identified in the review:

- Limited energy: every mobile device is powered by battery and, because of this, it has a certain lifetime period [31].
- Small screen: mobile device screens are pretty small and, because of this, interface design is limited [31].
- Limited performance: due to its size and technological advancement all mobile devices, even the most advanced in its class, have limitations of specific resources such as processing power, memory and connectivity. Because of this, the performance is limited [32].
- Bandwidth: given an application that requires the maximum, the minimum or a reasonable bandwidth, one must consider its enormous variation [32].
- Change of context: the change of context occurs in accordance with the environment [32].
- Reduced memory: due to its size and technological advancement, all mobile devices, even the most advanced in its class, have limitations of specific resources, including the size of its memory [31].
- Connectivity: the kind of connectivity that the application will use, such as 3G, bluetooth, infrared and Wi-Fi [33].
- Interactivity: what will be the type of input that the user will use to interact with the application [32].
- Storage: the applications have to take into consideration how it is going to be done [32].

- Software portability: the application should be performed on all types of operating systems [32].
- Hardware portability: the application should be performed on all types of devices [32].
- Usability: is a set of attributes which affect the effort needed for the use, in which it must be intuitive and as natural as possible to make or receive a call or text message [33].
- 24/7 availability: the application must be available to access anywhere, anytime [33].
- Security: must prevent accidental or deliberate unauthorized access to the applications and data [33].
- Reliability: is a set of attributes which affect the application's ability to maintain its level of performance under stated conditions for a stated period of time [34].
- Efficiency: is a set of attributes that relate to the relationship between the application's level of performance and the amount of resources used, under stated conditions [35].
- Native vs. Web Mobile: it must be defined if the application will be designed to be installed on the device itself, which is known as native applications, or used on the web [33].
- Interoperability: the application should be able to interact with other specific systems. In other words, it must have interoperability with other services [32].
- Response time: the applications must be initialized and finalized immediately [36].
- Privacy: the application must demonstrate to the user how his or her personal information are being collected, used and shared, and let the user exercise his or her choice and control over their use [33].
- Short term activities: activities in mobile applications tend to have a short duration, ranging from several seconds to several minutes [37].
- Data integrity: making sure that in an accidental shutdown of the application or of the device itself, the application will ensures data integrity [36].
- Key characteristics: mobile applications tend to be more focused or, in other words, they have specific key characteristics rather than offer the exploratory environment commonly used [37].
- Complex integration of real-time tasks: mobile applications should provide integration between application of different sources (native or web) [38].
- Constant interruption of activities: when using a mobile application, the activities are constantly interrupted, like when you receive a call, lose connection or have a low battery, which are examples of such interruptions [37].
- Functional area: data, collaboration and communication services, information services and productivity services such as business and office applications [39].
- Price: free, less than five euros and more than five euros [39].
- Target audience: applications for final private consumer or business applications [39].
- Provider type: businesses, professionals or other service providers [39].

After this survey, a refinement was made and a mix of characteristics was elicited with the purpose of defining which characteristics would be emphasized. Of a total of 23 types of characteristics that were most mentioned in the selected articles, a common denominator of 13 characteristics was reached, some of which had their names redefined, like “Interactivity”, which became “Input Interface”.

4. CHARACTERISTICS OF MOBILE APPLICATIONS SURVEY

With the conclusion of the systematic review, a survey was carried out among experts in mobile development with the purpose of ratifying the characteristics previously raised and to prove their respective influence on mobile development. The disclosure of the survey was conducted in more than 70 locations, among them universities and businesses, through e-mails, study groups and social groups.

In general, of all 117 feedbacks received through the survey, 100% of the experts confirmed the characteristics; among them, an average of 72% indicated a greater effort and complexity regarding the characteristics during development, an average of 12% indicated less effort and complexity and, finally, an average of 16% indicated they did not perceive any difference in mobile development, even though they confirmed the presence of the characteristics.

5. PROBLEM ADDRESSED

As noted in Section II, there is no estimation method developed for mobile applications projects. Moreover, some of the characteristics elicited in Section III aggravate the complexity and, thereafter, the effort in the development of mobile applications.

From the analysis that follows, with the characteristics of applications on mobile devices elicited in Section III, it is clear that they are different from the characteristics of traditional systems and directly influence its development. A clear example, which is different from the information or desktop systems, is the characteristic that the mobile devices have “Limited Energy”. As mobile devices are powered by battery, which have a limited lifetime period, the applications must be programmed to require the minimal amount of hardware resources possible, since the more resources consumed, the greater amount of energy expended. This characteristic makes it necessary for the solution project to address this concern, generating a higher complexity of development and, thereafter, a greater effort and cost.

Another specific characteristic of this context is the “Graphical Interface”. Due to the reduced screen size, the interface design is limited. Therefore, a greater complexity and, thereafter, a larger effort is required in the development of the graphical interface. Another characteristic related to the screen is the “Input Interface”, which defines how the user will interact with the application, in other words, if the user will interact via keypad, stylus, touch screen or voice and image recognition. The latter makes the task of developing applications that offers all these interaction options more complex, thus requiring a bigger effort.

Regarding connectivity, the characteristic “Bandwidth” was identified, wherein a mobile application might have the maximum band at times and the minimum in other moments. Some types of applications need to realize this and act differently in each situation. Another related feature is the “Connectivity Type”. Mobile applications can be developed to support different types of connectivity such as 3G, bluetooth, infrared, Wi-Fi, Wireless, NFC and others. In addition, a single application can support multiple types of connectivity simultaneously. These behaviors directly affect the complexity of the software and therefore require a larger development effort.

The “Change in Context” is also another characteristic inherent in mobile applications, which should take into account not only the data entries explicitly provided by users, but also the implicit entries concerning the physical and computational context of the users and the environments that surround them. In addition, the “Constant Interruption of Activities” is a much more common characteristic in this context, as well as the need for some applications to be developed to work offline and therefore be able to synchronize. Mobile applications should be prepared for different scenarios because the activities are interrupted constantly. Receiving a call, lack of connection and low battery are examples of such interruptions, which makes the applications become much more complex.

Despite the advances related to the computational ability of these devices, their hardware must still be considered as limited, especially when compared to desktops and servers. Two characteristics related to this issue are “Limited Performance” and “Reduced Memory”. Besides these, a characteristic inherent to the use of mobile devices is the “Response Time”, that is directly related to the power of “Processing”. Mobile applications must be initialized and finalized immediately, in other words, any development should be focused in the time variable. These characteristics require the applications to be developed with a possible resource optimization for a better efficiency and response time, requiring more effort.

The “Portability” is also a required characteristic of these applications. It can be divided into two characteristics: the “Hardware Portability” and the “Software Portability”. Regarding the first one, nowadays there is a large number of different devices with different capabilities and resources. A mobile application should be able to run on the largest number of devices possible. This requires an increased effort in the development. Moreover, a greater effort in testing this kind of portability is required. Regarding “Software Portability”, it is necessary to develop specific applications for each existing platform should the application be native. With this, more effort is required for replications of the same software product, including the tests.

Finally, mobile applications can be separated into two types: “Native or Web Mobile”. The first one has higher performance and easiness in accessing the hardware, while the second has lower performance since it is web, but it is easier to achieve portability. In addition, there are some applications that are considered hybrids. Depending on the type of application, the issues that must be considered and the complexity can be different, requiring different development efforts.
From the survey of the most popular estimation methods cited in Section III, it was found that these characteristics are not covered by the current estimation methods for two explicit reasons: first, none of the existing methods was designed to perform project estimation in mobile applications development; and second, all the characteristics discussed in this section are exclusive to mobile applications, with direct interference in their development, thereby generating a greater complexity and, thereafter, a greater effort. However, to consider any of the existing estimation methods to apply to the process of development of mobile applications is to assume that this kind of development is no different than the project of developing desktop applications, in other words, an eminent risk is assumed.

6. PROPOSAL: Estimation in Mobile Application Development Project

A solution to solve this problem would be to create a new estimation method or even to adapt some existing estimation method, in which would be added all of the characteristics identified that directly affect the mobile application development project, taking into account whatever is needed to reach the minimum efficiency in the estimates.

The approached proposed is an adaptation of an existing method, based exclusively on methods recognized as international standards by ISO. Among the most popular estimation methods mentioned in Section III, the method used to base the proposal below on is known as “Finnish Software Metrics Association (FiSMA)”. The model is one of the five methods for measuring software that complies with the ISO/IEC 14143-1 standard, is accepted as an international standard for software measuring [29] and nowadays over 750 software projects are completed being estimated by FISMA. However, the difference between this and other methods that are in accordance with the above standard, which are the Common Software Measurement International Consortium Function Points (COSMIC FP) [25], the International Function Point Users Group (IFPUG) FPA [11], MarkII FPA [14] and the Netherlands Software Metrics Association (NESMA) WSF [40], is that the method used is based in functionality but is service-oriented. It also proposes in its definition that it can be applied to all types of software, but this statement is lightly wrong since in its application, the method does not take into account the characteristics elicited in Section IV.

The COMISC FP [25], the MarkII FPA [14] and the NESMA [40] were created based on the FPA [11], in other words, they assume the counting of Function Point (FP), but considering the implemented functionality from the user's point of view. With this, it is clear that the methods mentioned above do not take into account the characteristics of mobile applications because they are not noticed by the user. The methods are independent of the programming language or technology used. And, unlike FISMA, they do not bring in their literature the information that they can be applied to all types of software.

Overall, the FISMA method proposes that all services provided by the application are identified. It previously defines some services, among which stands out the user's interactive navigation, consulting services, user input interactive services, interface services for other applications, data storage services, algorithmic services and handling services. Finally, after identifying all the services, the size of each service is calculated using the same method and thus obtaining a total functional size of the application by adding the size of each service found [41].

6.1. Approaching the Chosen Model

The FiSMA method in its original usage proposes a structure of seven classes of the Base Functional Component or BFC (Base Functional Component) type, which is defined as a basic component of functional requirement. The seven classes used to account for the services during the application of the method are [41]: interactive navigation of the end user and query services (q); interactive input services from end users (i); non-interactive outbound services for the end user (o); interface services for another application (t); interface services for other applications (f); data storage services (d) and algorithmic manipulation services (a).

The identification for each class name BFC previously mentioned, with a letter in parenthesis, is used to facilitate the application of the method during the counting process, because each of the seven classes BFCs are composed of other BFC classes which, at the time of calculating, these BFCs "daughter" classes are identified by the letter of their BFC "mother" class followed by a numeral, as can be seen in Figure 2.

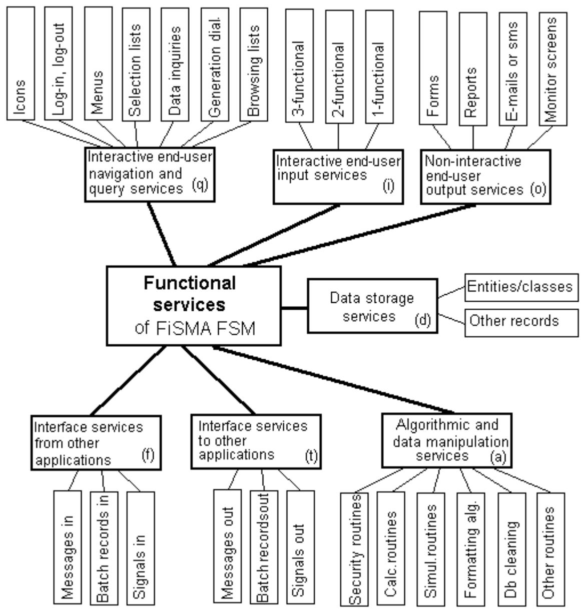


Figure 2. Types of BFCs classes of the base model [41]

The unit of measurement is the point of function with the letter “F” added to its nomenclature to identify the “FiSMA”, resulting in FfP (FiSMA Function Point) or Ffsu (FiSMA functional size unit). The measurement process generally consists of measuring the services and end-user interface and the services considered indirect [41], as can be seen in Figure 3.

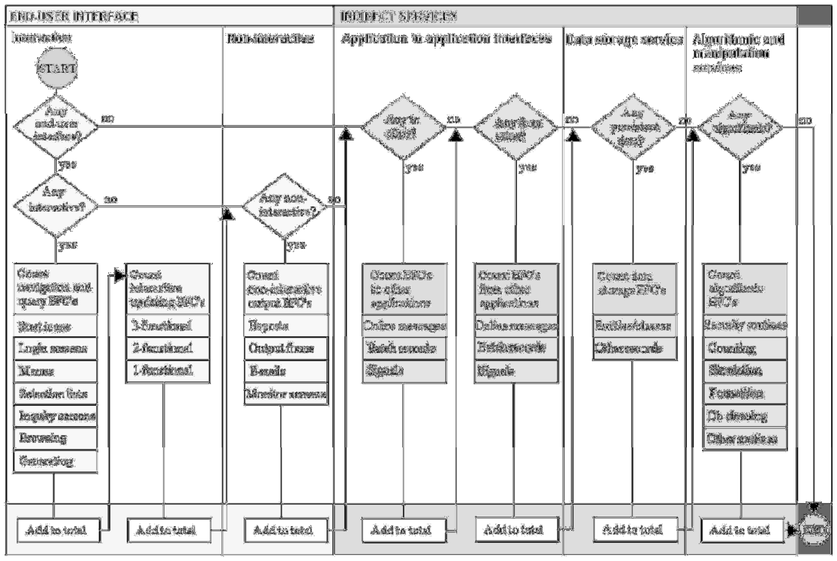


Figure 3. Representation of the measurement process of the base model [41]

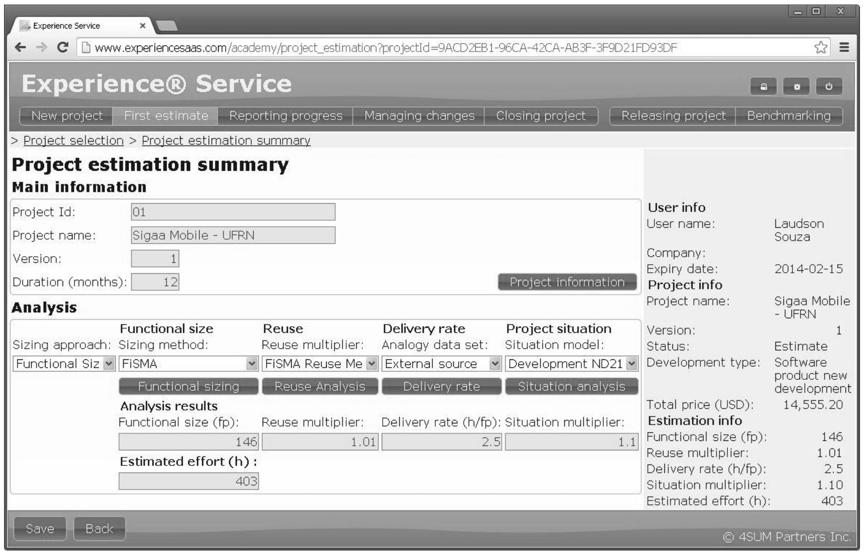
Figure 3 shows the process of measuring the base model, in which it defines each step and sum of each BFC class of the model. Briefly, the process of counting should be done as follows. Identify: 1-How many types of BFCs does the software have? 2-Which are they? (identify all) 3-What are they? (provide details of each BFC identified)
After doing this, it is necessary to add each BFC root using the formulas pre-defined by the method and their assignments. Finally, the formula of the final result of the sum is the general sum of all the BFCs classes.

6.2. Applying the Chosen Model

The FiSMA method can be applied manually or with the aid of the Experience Service[1] tool, which was the case, provided by FiSMA itself through contact made with senior consultant Pekka Forselius and with the chairman of the board Hannu Lappalainen.

When using the tool, it is necessary to perform all the steps of the previous subsection to obtain the functional size. Figure 4 shows the final report after the implementation of the FiSMA on a real system, the Management of Academic Activities Integrated System (Sigaa) in its Mobile version, developed by the Superintendence of Computing (SINFO) of the Federal University of Rio Grande do Norte (UFRN).

[1] http://www.experiencesaas.com/



Project estimation summary

Main information

Project Id:	01
Project name:	Sigaa Mobile - UFRN
Version:	1
Duration (months):	12

Analysis

	Functional size	Reuse	Delivery rate	Project situation
Sizing approach:	Sizing method:	Reuse multiplier:	Analogy data set:	Situation model:
Functional Siz	FiSMA	FiSMA Reuse Me	External source	Development ND21

Analysis results

Functional size (fp):	Reuse multiplier:	Delivery rate (h/fp):	Situation multiplier:
146	1.01	2.5	1.1

Estimated effort (h) : 403

User info

User name:	Laudson Souza
Company:	
Expiry date:	2014-02-15

Project info

Project name:	Sigaa Mobile - UFRN
Version:	1
Status:	Estimate
Development type:	Software product new development
Total price (USD):	14,555.20

Estimation info

Functional size (fp):	146
Reuse multiplier:	1.01
Delivery rate (h/fp):	2.5
Situation multiplier:	1.10
Estimated effort (h):	403

Figure 4. Final Report of FiSMA applied to Sigaa Mobile

After the application of FiSMA, the functional size of the software is obtained and from this it is possible to find the effort using the formula: Estimated effort (h) = size (fp) x reuse x rate of delivery (h/fp) x project status; the latter is related to productivity factors that are taken into account for the calculation of the effort. However, of the factors predefined by the FiSMA regarding the product, only 6 (six) are proposed, in which the basic idea of the evaluation is that "the better the circumstances of the project, the more positive the assessment". The weighting goes from - - to + +, as follows:

Caption:

- "+ +" = [1.10] Excellent situation, much better circumstances than in the average case
- "+" = [1.05] Good situation, better circumstances than in the average case
- "+ / -" = [1.0] Normal situation

- "-" = [0.95] Bad situation, worse circumstances than in the average case
- "- -" = [0.90] Very bad situation, much worse circumstances than in the average case

Productivity factors:

Functionality requirements: compatibility with the needs of the end user, the complexity of the requirements.

- (- -) Complex and critical application area (thousands of FPs), multiple users and multicultural system.
- (-) Interoperable application area with some complex characteristics, requiring special understanding from users and developers.
- (+ / -) Partly automated, integrated application area and a medium size application (between 600 and 1000 FPs) with standard security requirements.
- (+) Application area mostly automated and application with less than 5 interfaces with other systems; there are specific security requirements.

- (+ +) Very mature application area, simple and easy, a small stand-alone application (less than 200 FPs) for a small group of users.

Reliability requirements: maturity, tolerance to faults and recovery for different types of use cases.

- (- -) Malfunctions may put in danger human lives and cause significant economic or environmental losses.}
- (-) The software is part of a large real-time system where all the failures of operation will cause problems to many other applications.}
- (+ / -) Not more than 2 hours of downtime is acceptable, but the system recovery routines are appropriate.
- (+) Need for non-continuous operation, but daily.
- (+ +) Need for periodic operation. Pausing for a few days will not cause any damage to the organization.

Usability requirements: understandability and easiness to learn the user interface and workflow logic.

- (- -) A large number of different types of end users around the world.
- (-) 2 or 3 different types of users with different skills.
- (+ / -) A large number of end users with equal abilities.
- (+) No more than tens or hundreds of homogeneous users in perhaps more than one location.
- (+ +) Only a few users, all located on one site.

Efficiency requirements: effective use of resources and adequate performance in each use case and under a reasonable workload.

- (- -) Complex database with millions of data records and transactions per day, thousands of simultaneous end users.
- (-) Large database, hundreds of simultaneous end users, critical response most of the time.
- (+ / -) Large database, less than millions of data records and less than hundreds of simultaneous end users.
- (+) Medium database in volume and structure, simple and predictable data requests from some simultaneous end users.
- (+ +) Simple and small database without simultaneous end users or complex data requests.

Maintainability requirements: lifetime of the application, criticality of fault diagnosis and test performance.

- (- -) Very large strategic software (over 20 years of lifetime) in a volatile area of business, with frequent changes in laws, regulations and business rules.
- (-) Large software (10-20 years of lifetime), and frequent changes in laws, regulations and business rules.

- (+ / -) Medium size software (5-10 years of lifetime), monthly changes in laws, regulations and business rules.
- (+) Small software, rarely changes (2 to 5 years of lifetime).
- (+ +) Temporary software (less than 2 years of lifetime), without modifications.

Portability requirements: adaptability and instability to different environments, to the architecture and to structural components.

- (- -) Software users are located in many types of organizations, with various platforms (hardware, browsers, operating systems, middleware, protocols, etc), various versions and various update frequencies.
- (-) The software must operate on some different platforms (hardware, browsers, operating systems, middleware, protocols, etc) and in various versions of each of them.
- (+ / -) Each version of the software must run on multiple versions of a given platform (hardware, browser, operating system, middleware, protocols, etc), and the frequencies of update of the users are quite predictable.
- (+) The software must run on a given platform (hardware, browser, operating system, middleware, protocols, etc), but the use of system-level services is limited because the upgrade process is partial.
- (+ +) Software must be run on a particular platform (hardware, browser, operating system, middleware, protocols, etc), but the upgrade process is completely controllable.

Among the productivity factors mentioned above, only the "Portability Requirement" factor fits in harmony with the "Portability" characteristic regarding both hardware and software. However, none of the other factors discusses the characteristics of mobile application, in other words, after obtaining the functional size of the software and applying the productivity factors related to the product to estimate the effort, this estimate ignores all of the characteristics of mobile applications, judging that the estimate of traditional information systems is equal to the mobile application. However, with the proposal of the creation of new productivity factors, which would be the specific characteristics of mobile applications, this problem will be solved, as presented below.

Performance Factor:

- (-) The application should be concerned with the optimization of resources for a better efficiency and response time.
- (+ / -) Resource optimization for better efficiency and response time may or may not exist.
- (+) Resource optimization for better efficiency and response time should not be taken into consideration.

Power Factor:

- (-) The application should be concerned with the optimization of resources for a lower battery consumption.
- (+ / -) Resource optimization for lower battery consumption may or may not exist.
- (+) Resource optimization for a lower battery consumption should not be taken into consideration.

Band Factor:

- (-) The application shall require the maximum bandwidth.
- (+ / -) The application shall require reasonable bandwidth.
- (+) The application shall require a minimum bandwidth.

Connectivity Factor:

- (-) The application must have the maximum willingness to use connections such as 3G, Wi-fi, Wireless, Bluetooth, Infrared and others.
- (+ / -) The application must have reasonable predisposition to use connections such as 3G, Wi-Fi and Wireless.
- (+) The application must have only a predisposition to use connections, which can be: 3G, Wi-fi, Wireless, Bluetooth, Infrared or others.

Context Factor:

- (-) The application should work offline and synchronize.
- (+ / -) The application should work offline and it is not necessary to synchronize.
- (+) The application should not work offline.

Graphic Interface Factor:

- (-) The application has limitations due to the screen size because it will be mainly used by cell phone users.
- (+ / -) The application has reasonable limitation due to the screen size because it will be used both by cell phone and tablet users.
- (+) The application has little limitation due to the screen size because it will be mainly used by tablet users.

Input Interface Factor:

- (-) The application must have input interfaces for touch screen, voice, video, keyboard and others.
- (+ / -) The application must have standard input interfaces for keyboard.
- (+) The application must have any one of the types of interfaces, such as: touch screen, voice, video, keyboard or others.

The proposed factors take into account the same weighting proposed by FiSMA, but only ranging from - to +, in other words:

- “+” = [1.05] Good situation, better circumstances than in the average case
- “+ / -” = [1.0] Normal Situation
- “-” = [0.95] Bad situation, worse circumstances than in the average case

The functional size remains the same, thus affecting only the formula used to obtain the effort, which will now consider in its "project situation" variable the new productivity factors specific for mobile applications.

7. Conclusion

Given the results presented, based on the literature review of estimation methods and on the systematic review of the characteristics of mobile applications, it was observed that this sub-area of software engineering still falls short. Basically, it's risky to use any existing estimation method in development projects for mobile applications, as much as there are some models already widespread in industry, such as the Function Point Analysis, the Mark II and the COSMIC-FFP, which are even approved by ISO as international standards. They all fall short by not taking into account the particularities of mobile applications, which makes the method partially ineffective in this situation.

With the common emergence of new systems, experts always find a barrier when using one of the current methods of software measurement. This barrier can be on the effectiveness of the method, on what type of method should be used, when it comes to a software that is considered unconventional and, mostly, when it is required to apply it in completely atypical scenarios. This whole situation is aggravated further when it comes to mobile applications.

Based on this study, it is concluded that the proposal presented in this work is entirely appropriate and viable and that this proposal should take into account all the peculiarities of such applications, finally creating a belief that there actually are considerable differences in the development project for mobile applications.

Regarding future work, it is possible to accomplish the same systematic review with other purposes, increasing the number of research questions and, thereby, increasing the number of selected articles. Finally, exhaustively applying the presented proposal in several development projects for mobile devices in order to attain a perfect validation and a solid foundation in the industry.

References

[1] L. Naismith, M. Sharples, G. Vavoula, P. Lonsdale *et al.*, "Literature review in mobile technologies and learning," 2004.

[2] G. Macario, M. Torchiano, and M. Violante, "An in-vehicle infotainment software architecture based on google android," in *Industrial Embedded Systems, 2009. SIES '09. IEEE International Symposium on*, 2009, pp. 257–260.

[3] T. Liu, H. Wang, J. Liang, T.-W. Chan, H. Ko, and J. Yang, "Wireless and mobile technologies to enhance teaching and learning," *Journal of Computer Assisted Learning*, vol. 19, no. 3, pp. 371–382, 2003.

[4] I. GARTNER. (2013) Gartner says worldwide mobile phone sales declined 1.7 percent in 2012. egham, uk: Gartner, 2013. [Online]. Available: http://www.gartner.com/newsroom/id/2335616

[5] C.-C. Yang, H.-W. Yang, and H.-C. Huang, "A robust and secure data transmission scheme based on identity-based cryptosystem for ad hoc networks," in *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference*, ser. IWCMC '10. New York, NY, USA: ACM, 2010, pp. 1198–1202, aCM. [Online]. Available: http://doi.acm.org/10.1145/1815396.1815670

[6] I. Ketykó, K. De Moor, T. De Pessemier, A. J. Verdejo, K. Vanhecke, W. Joseph, L. Martens, and L. De Marez, "Qoe measurement of mobile youtube video streaming," in *Proceedings of the 3rd workshop on Mobile video delivery*, ser. MoViD '10. New York, NY, USA: ACM, 2010, pp. 27–32. [Online]. Available: http://doi.acm.org/10.1145/1878022.1878030

[7] S.-Y. Yang, D. liang Lee, and K.-Y. Chen, “A new ubiquitous information agent system for cloud computing - example on gps and bluetooth techniques in google android platform,” in *Electric Information and Control Engineering (ICEICE), 2011 International Conference on*, 2011, pp. 1929–1932.

[8] R. Lowe, P. Mandl, and M. Weber, “Context directory: A context-aware service for mobile context-aware computing applications by the example of google android,” in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, 2012, pp. 76–81.

[9] N. Husted, H. Sadi, and A. Gehani, “Smartphone security limitations: conflicting traditions,” in *Proceedings of the 2011 Workshop on Governance of Technology, Information, and Policies*, ser. GTIP ’11. New York, NY, USA: ACM, 2011, pp. 5–12. [Online]. Available: http://doi.acm.org/-10.1145/2076496.2076497

[10] A. Shabtai, Y. Fledel, U. Kanonov, Y. Elovici, S. Dolev, and C. Glezer, “Google android: A comprehensive security assessment,” *Security Privacy, IEEE*, vol. 8, no. 2, pp. 35–44, 2010.

[11] S. Oligny, J.-M. Desharnais, and A. Abran, “A method for measuring the functional size of embedded software,” in *3rd International Conference on Industrial Automation*, 1999, pp. 7–9.

[12] B. Boehm, R. Valerdi, J. Lane, and A. Brown, “Cocomo suite methodology and evolution,” *CrossTalk*, vol. 18, no. 4, pp. 20–25, 2005.

[13] C. Jones and T. C. Jones, *Estimating software costs*. McGraw-Hill New York, 1998, vol. 3.

[14] C. Symons, “Come back function point analysis (modernized)–all is forgiven!),” in *Proc. of the 4th European Conference on Software Measurement and ICT Control, FESMA-DASMA*, 2001, pp. 413–426.

[15] M. Lother and R. Dumke, “Points metrics-comparison and analysis,” in *International Workshop on Software Measurement (IWSM 2001), Montréal, Québec*, 2001, pp. 155–172.

[16] J. Engelhart, P. Langbroek *et al.*, *Function Point Analysis (FPA) for Software Enhancement*. NESMA, 2001.

[17] D. J. Reifer, “Asset-r: A function point sizing tool for scientific and real-time systems,” *Journal of Systems and Software*, vol. 11, no. 3, pp. 159–171, 1990.

[18] M. Maya, A. Abran, S. Oligny, D. St-Pierre, and J.-M. Desharnais, “Measuring the functional size of real-time software,” in *Proc. of 1998 European Software Control and Metrics Conference, Maastricht, The Netherlands*, 1998, pp. 191–199.

[19] S. Kusumoto, F. Matukawa, K. Inoue, S. Hanabusa, and Y. Maegawa, “Estimating effort by use case points: method, tool and case study,” in *Software Metrics, 2004. Proceedings. 10th International Symposium on*, 2004, pp. 292–299.

[20] R. Banker, R. Kauffman, C. Wright, and D. Zweig, “Automating output size and reuse metrics in a repository-based computer-aided software engineering (case) environment,” *Software Engineering, IEEE Transactions on*, vol. 20, no. 3, pp. 169–187, 1994.

[21] J. Matson, B. Barrett, and J. Mellichamp, “Software development cost estimation using function points,” *Software Engineering, IEEE Transactions on*, vol. 20, no. 4, pp. 275–287, 1994.

[22] R. Meli, “Early and extended function point: a new method for function points estimation,” in *Proceedings of the IFPUG-Fall Conference*, 1997, pp. 15–19.

[23] M. Morisio, I. Stamelos, V. Spahos, and D. Romano, “Measuring functionality and productivity in web-based applications: a case study,” in *Software Metrics Symposium, 1999. Proceedings. Sixth International*, 1999, pp. 111–118.

[24] G. Caldiera, G. Antoniol, R. Fiutem, and C. Lokan, “Definition and experimental evaluation of function points for object-oriented systems,” in *Software Metrics Symposium, 1998. Metrics 1998. Proceedings. Fifth International*, 1998, pp. 167–178.

[25] C.-C. S. M. I. Consortium *et al.*, “The cosmic functional size measurement method-version 3.0 measurement manual (the cosmic implementation guide for iso/iec 19761: 2003),” 2007.

[26] R. Meli, A. Abran, V. T. Ho, and S. Oligny, “On the applicability of cosmic-ffp for measuring software throughout its life cycle,” in *Proceedings of the 11th European Software Control and Metrics Conference*, 2000, pp. 18–20.

[27] J. Kammelar, “A sizing approach for oo-environments,” in *Proceedings of the 4th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering*, 2000.

[28] S. Abrahão, G. Poels, and O. Pastor, “A functional size measurement method for object-oriented conceptual schemas: design and evaluation issues,” *Software & Systems Modeling*, vol. 5, no. 1, pp. 48–71, 2006.

[29] P. Forselius, "Finnish software measurement association (fisma), fsm working group: Fisma functional size measurement method v. 1.1." 2004.

[30] B. Kitchenham, "Procedures for performing systematic reviews," *Keele, UK, Keele University*, vol. 33, p. 2004, 2004.

[31] J.-H. Sohn, J.-H. Woo, M.-W. Lee, H.-J. Kim, R. Woo, and H.-J. Yoo, "A 50 mvertices/s graphics processor with fixed-point programmable vertex shader for mobile applications," in *Solid-State Circuits Conference, 2005. Digest of Technical Papers. ISSCC. 2005 IEEE International*, 2005, pp. 192–592 Vol. 1, google.

[32] H. Mukhtar, D. Belad, and G. Bernard, "A model for resource specification in mobile services," in *Proceedings of the 3rd international workshop on Services integration in pervasive environments*, ser. SIPE '08. New York, NY, USA: ACM, 2008, pp. 37–42, aCM. [Online]. Available: http://-doi.acm.org/10.1145/1387309.1387318

[33] H. Feng, "A literature analysis on the adoption of mobile commerce," in *Grey Systems and Intelligent Services, 2009. GSIS 2009. IEEE International Conference on*, 2009, pp. 1353–1358, iEEE.

[34] A. Kumar Maji, K. Hao, S. Sultana, and S. Bagchi, "Characterizing failures in mobile oses: A case study with android and symbian," in *Software Reliability Engineering (ISSRE), 2010 IEEE 21st International Symposium on*, 2010, pp. 249–258, iEEE.

[35] J. Al-Jaroodi, A. Al-Dhaheri, F. Al-Abdouli, and N. Mohamed, "A survey of security middleware for pervasive and ubiquitous systems," in *Network-Based Information Systems, 2009. NBIS '09. International Conference on*, 2009, pp. 188–193, iEEE.

[36] K. Hameed *et al.*, "Mobile applications and systems," 2010, google.

[37] I. R. D. E. A. G. Alekhya Mandadi, Deepti Mudegowder, "Mobile applications: Characteristics & group project summary," *Mobile Application Development*, 2009, google.

[38] M. Hayenga, C. Sudanthi, M. Ghosh, P. Ramrakhyani, and N. Paver, "Accurate system-level performance modeling and workload characterization for mobile internet devices," in *Proceedings of the 9th workshop on MEmory performance: DEaling with Applications, systems and architecture*, ser. MEDEA '08. New York, NY, USA: ACM, 2008, pp. 54–60, aCM. [Online]. Available: http://-doi.acm.org/10.1145/1509084.1509092

[39] A. Giessmann, K. Stanoevska-Slabeva, and B. de Visser, "Mobile enterprise applications–current state and future directions," in *System Science (HICSS), 2012 45th Hawaii International Conference on*, 2012, pp. 1363–1372, google.

[40] C. Gencel, R. Heldal, and K. Lind, "On the conversion between the sizes of software products in the life cycle."

[41] F. S. M. A. FiSMA. (2004) Fisma functional size measurement method version 1-1. [Online]. Available: http://www.fisma.fi/in-english/methods/

Authors

Laudson Silva de Souza is masters student the Department of Informatics and Applied Mathematics, Federal University of Rio Grande do Norte, Brazil.

Gibeon Soares de Aquino Jr. is PhD teacher the Department of Informatics and Applied Mathematics, Federal University of Rio Grande do Norte, Brazil.