Proposing New Model for Effort Estimation of Mobile Application Development

Nidhi Singh
Department of Computer Science
Jaypee Institute of Information Technology
Noida (U.P)

Devpriya Soni, PhD
Department of Computer Science
Jaypee Institute of Information Technology
Noida (U.P)

ABSTRACT
There have been many developments in the field of mobile technology in the last few years. Mobile technology grown very fast. Initially mobiles were only for rich but now all household have mobiles and using mobile technology because of great development in this field. Which is possible due to research in this area. Not only big companies but individuals also been engaged in this field. Billions of mobile application has been developed by software developers. In the recent year's mobile application are developed by software developers to considering the demands of our society. Mobile application developers faces many challenges like size, memory and the functionality. So in this paper we focused on the differences between traditional estimation model and mobile estimation model and hereby, proposed a new model for mobile applications and partially validated the same.

Keywords
Software Engineering, Mobile Applications, Estimating Software, Software Quality.

1. INTRODUCTION
Mobile devices have become a crucial part of our daily life. Nowadays, people are fully dependent on technology. We can say that mankind became slave of technology. Mobile phones are small computer with the limited processing power, small enough to occupy handful or less space and contain operating system having the ability to run applications. The mobile devices have a screen, virtual keyboard and icons on the screen. A mobile device communicates with the help of internet and is interconnected to other mobile/ non-mobile devices through Wi-Fi and Bluetooth etc. Mobile devices find place in several key areas such as businesses, social networking, gaming, health, entertainment [1]. Mobile applications are intensive programs developed to work on tablets, smart phones and other mobile devices. In recent there is a great development in the field of mobile applications and devices. It is increasing day-by-day and there is large increment in number of mobile applications every year [2]. Mobile application estimation model is much different than the traditional software estimation model. Mobile application estimation model takes much less time, it takes round about two to three months to develop a mobile application. But in case of traditional software estimation model, it takes three to five years to develop. Mobile application estimation model requires team comprising of three to five members including programmer, tester and developer. But the traditional estimation model requires team size up to 30 people and the most important factor is budget. Mobile application estimation model budget is around 100$ but traditional software estimation model has budget in the range of 2 to 5 million dollars. Every mobile device has unique set of the mobile apps, representation of new features and methods for development process, including testing, coding, designing and maintaining mobile apps [2]. In this paper we focus on the differences between traditional estimation model and mobile estimation model and further propose a new model for mobile application estimation and partially validate the same. Rest of the paper is organized as follows: Section 2, describes the main estimation methods. Section 3, discusses the literature survey of the traditional and mobile estimation model. The characteristics of mobile application has been discusses in Section 4. The proposed model has been introduced in Section 5. Section 6, describes the counting rules. Finally Section 7, conclusion.

2. ESTIMATION METHODS
In order to identify how the traditional estimation methods could address the characteristics of the systems.

Table 1. Main estimation methods

Year	Method	Author
1979	Function point Analysis	Albrecht [3]
1981	COCOMO Model	Barry Boehm's [4],[5],[6]
1988	Mark II FPA	Charles Symons [7]
1990	NESMA	The Netherlands Software Metrics User Association [8]
1999	COSMIC Full Function Point	Common Software Measurement International Consortium [9]
2004	FiSMA FSM	The Finnish Software Metrics Association [10]

Table 1. Display the main estimation methods, showing the year of creation, the name of the methods and the author is associate it. We discuss the charterstics. The use of these methods to estimate the effort of the mobile application developments.

3. LITERATURE SURVEY
There are some traditional estimation methods COCOMO, FPA, MARK- II, NESMA FPA, and the mobile estimation methods are COSMIC FFP and FiSMA FSM. So the traditional estimation methods are: COCOMO is developed in 1980 and the COCOMO-II is developed in 1981 both

methods is introduced by Berry Boehm [4]. The COCOMO model is stands for constructive cost model. This model is evaluate the effort, cost and the schedule for the software project. COCOMO model have three levels basic, intermediate and detailed. Basic model is give the quick and rough estimation of the software project and calculate the effort in the KSLOC (thousand source line of code). Basic model is divided into three classes organic, semi-detached and embedded. The intermediate model is the enhancement of the basic model. This model calculated the software development effort by using the 15th cost drivers. The cost drivers are attributes, software attributes, hardware, personnel attributes, product attributes and project attributes. The detailed model is the enhancement of intermediate model. This model is a phase sensitive model. In this model add the effort multiplier in each phase and cost drivers in each steps. COCOMO-I model have some drawbacks, they calculate the size of the software in KDSI. (KDSI is thousand delivery source instruction) This KDSI is not measuring the size of the software. It is measuring the length of the software [4], [5]. COCOMO-II model is a successor of COCOMO-I model. It was built because the COCOMO-I model was not compatible with the more recent practices in the software development. COCOMO-II model is fully concentrated on the three phases of the spiral life cycle design. The phases are, the application composition, the earlier design phase and the last phase is the post architecture phase. There are some differences in the size measurement. COCOMO-II model measure the size of the software in SLOC (Source line of code). Source line of code is measure the size of software by counting a number of line in the given program. It is very simple process, but it can be calculated accurately only after the development of the code. There are some disadvantages of SLOC as it is language dependent. SLOC is difficult to measure the size in the earlier stage and no industrial standard for measuring the SLOC [6]. Function point analysis was developed by Allan Albrecht in 1979. This method overcome some of the difficulties associated with the SLOC as measure the size of software. According to Albrecht *"unit of measurement to express the amount of business functionality provided to the user"*. Function point analysis is a measure of size for software in terms of functionality and provided to user. Function point analysis have five factors. But these five factors have divided into two parts. The first part is data function and the second part is transaction function. Data function is further divided into two parts internal logical file, external interface file. Transaction function have three parts external input, external output and the external inquires. The function point analysis have some drawbacks, It measures the size of the software in a single total value obtain by a specific function of size measurement method [3]. Mark-II method is designed to estimate the business information systems and certified by the ISO standard. This method was created to improve the shortcomings of the function point analysis method [7]. The Netherlands Software Metrics User Association method was designed for counting the function points of the software same as FPA method [8]. There are some models proposed by researchers for the mobile application estimation models. COSMIC is one of them, which is a second generation function size measurement method. Functional size measurement methods are used in many areas of the software application developments. There are four types of data movement Entry, Exit, Read and Write. In the Entry phase data is measured from functional user to the functional process. In the Exit phase data is measured from a functional process to a functional user. In the Write phase data is measured from a functional process to persistent storage. In the Read phase data is measured from persistent storage to a functional process. The cosmic measurement process consist of three phases (i) Measurement strategy (ii) Mapping phase (iii) Measurement phase. In the last they took sum of all the data movement [9]. FiSMA function size measurement method measures the size of software. FiSMA method was developed in the working group of "finnish software measurement association" that is called FiSMA. This method was applied in Finland. FiSMA method measure more than 600 software development project in between 1997 to 2003. FiSMA functional size measurement method is based on the user requirement. Here user requirements is functional and nonfunctional requirements. FiSMA function size measurement methods are service oriented. Which can be identification of all the different services given by a software. FiSMA method is based on the functional user requirements. It measure the size of the software from the perspective of user. Functional user requirements specifies as a function i.e. Base Function Components (BFC). Base function type are sub attributes which are divided into many small component then they are measured as a separate domain and further the functional size is added to it. The BFC is divided into the base function classes and the base function classes is further divided into base function types. There are seven classes of BFC: *(i) Interactive end-user navigation and query services (ii) End-user input services (iii) End-user output services (iv) Interface services to other application. (v) Interface services from the other application. (vi) Data storage services. (vii) Algorithm and manipulation services.* These BFC classes are further decompose into the BFC types. There are 28 BFC types [10]. The FiSMA Model is calculating the functional size of the mobile application. But there are some limitation. The rules which calculate the function size is very complicated. Because there are lots of components and measurement of all the components are very typical. Some components are not quantifiable. That's why we introduce the new model with less components and which easily calculate the size of the mobile application.

4. CHARTERSTICS OF MOBILE APPLICATION

There are some charterstics that are distinguish mobile apps from the traditional application. These charterstics are divided into three parts hardware, software and the communication.

4.1 Charterstics of hardware

4.1.1 Limited power – Mobile application has less processing power, and relativity small in size by LOC. Mobile application is less memory space than the traditional application.

4.1.2 Screen size – Mobile phone screen size is small and varies from one devices to the other devices.

4.1.3 Start-up time – Mobile devices user use the mobile phone in short duration and mobile phone has an ability to quickly start a mobile application [2].

4.2 Charterstics of software

4.2.2 User-Interface – Mobile application should be designed to match the target mobile environment. The

target environments standards are important to the user with the pleasant application.

4.2.2Interaction with the information sources – In the mobile application data is transfer from one application to other application [2].

4.3 Charterstics of communication

4.3.1 Network communication – Network is very important factor for the communication from one device to other device. Mobile devices are connected to the internet, GPS system by the network communication [2].

5. IMPLEMENTATION DETAILED

We proposed a new model for mobile application, with the name MEstimation model. MEstimation model is proposed for the evaluation of the effort required for the development of mobile application. The MEstimation model measure the size of the application from the user perspective. Base function component is a unit of measurement. Base function components are based on the assessment of functional user requirements. Base function type is sub divided into the small components and sum of all the component added together. MEstimation model is divided into five components. These components are called Base Function Class (BFC) and the Base Function Classes are further divided into Base Function Types. So the five components are (i) End-user interface services (ii) End-user input services (iii) End-user output services (iv) Data storage services (v) Interaction to the other application.

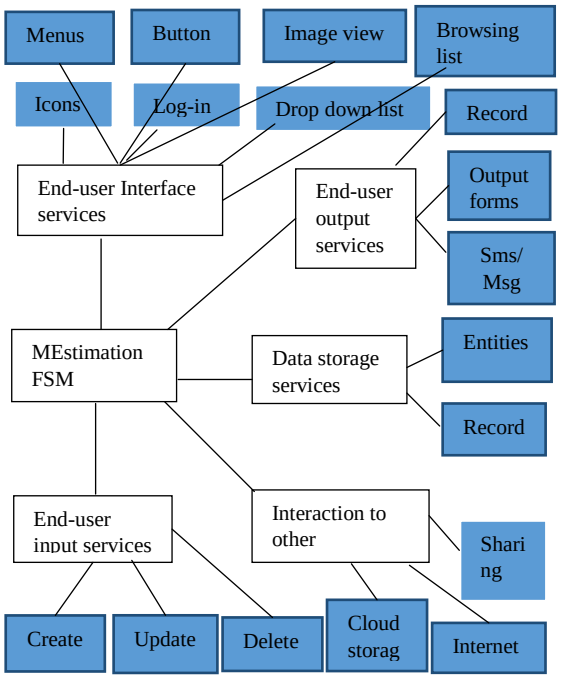

Fig. 1: MEstimation FSM Model

5.1 End- user interface (e) - The parts where is no maintenance of the data storage of the system of interactive user interface are specified by end-user. When data is created, updated or deleted as a result of some service, it is called maintenance services. Here we calculate the functions which are totally dependent on number of needed reading references to entity types and amount of data elements of measured BFC.

End-user interface services are divided into seven types

5.1.1 Icons- Icon is small representation of program file. Icons doesn't store the much information. When we click on the icon then the associated file is opened.

5.1.2 Log-in and log-out window - This is main screen to open an application. It is used for the security purpose. They are used for restricting access to the user and the preventing to illegal uses and log-out function data will not change and updated by the end- user.

5.1.3 Menus- Menus are used for selection the next operation. In the menus the data will not updated and changed by the end-user.

5.1.4 Drop down list- It is a part of interactive end-user interface. Drop-down list have many choices to display in the limited space. Drop-down list have many names selection list, combo list and list boxes etc. The data will not change and updated by the end- user.

5.1.5 Button- Button is used to save the information provided by the user. (Ex- Save, Search, Cancel)

5.1.6 Image view- It is user-interface widget that is used to display the images in your application.

5.1.7 Browsing list- It is high list of the data. On which we can further filter results and check the details. In the browsing list the data will not updated by the end- user.

5.2 End-user input services (i) - The parts where is maintenance of the data storage of the system of interactive user interface are specified by end-user input services. When data is created, updated or deleted as a result of some service, it is called maintenance services. The data storage means entity. Here we calculate the functions which are totally dependent on number of needed reading references and writing references to entity types and amount of data elements of measured BFC.

End- user input services are divided by three BFC types

5.2.1 Create- It can maintain only one maintenance services.

5.2.2 Update - It can maintain only two maintenance services.

5.2.3 Delete - It can maintain all the types of maintenance create, update and delete.

5.3.3 End-user output services (o) - The parts where do no maintenance of the data storage of the system of interactive user interface are specified by end-user output services. Here we calculate the functions which are totally dependent on number of needed reading references to entity types and amount of data elements of measured BFC. There are three BFC types

5.3.1 Reports- Report is a documents, whose structure is vary according to the framework.

5.3.2 Output forms- Output forms are also documents. Which are represent the standard structure.

5.3.3 Text messages and Sms – It is a transmitted document, which have standard structure. Structure contains data fields, text fields, subject field and optional attachments.

5.4 Data storage services (d) - Data storage services are the collection of related and self- contained data in the real world about which the user requires the software to provide persistent storage. Here we calculate the function size depends on the data elements of the BFC.

Data storage services are divided into two types

5.4.1 Classes – An entities contain similar types of data. Entities is logical components of data storage services. Entities represent thing according to the user, in which information to be stored.

5.4.2 Records- It stores the records in the table. Records are many kind of technical records table records (text table).

5.5 Interaction to the other application (s) Interaction to other application determine all the data transfer functions moving the data from one application to the other application. Here we calculate the functions are totally dependent on number of needed reading references to entity types and amount of data elements of measured BFC.

5.5.1 Cloud Storage – Cloud Storage is just like data storage. In cloud we store the digital data in a logical pools. We can share the data, that provide shared computer processing resources and data to computers and other devices.

5.5.2 Internet - Means connecting to one device to any other device anywhere in the world. It can easily send and receive any kind of information.

5.5.3 Sharing to other application – We can share the data from one application to other application by the network.

6. COUNTING RULES

6.1 End-user Interface (e)

Se = n/ De + r/ Re
Where Se = screen of interface screen.
n = No's of data elements.
r = No's of reading references to entities.
De = Data elements are found in the user interface BFC Class.
Re = Reading references are found in the user interface class.
Fsu = functional sizing unit.

6.2 End-user input services (i)

Si = m * (n/ Di + w/ Wi + r/ Ri)
Where Si = size of input.
n = No's of data elements.
w = No's of writing reference's to entities.
r = No's of reading reference's to entities.
Di = Data elements found in the input BFC class.
Wi = Writing references are found in the input BFC class.
Ri = Reading references are found in the input BFC class.
m = here m is a functionality multiplier. The value of 1, 2 and 3 are depends on how many function is create, update and delete.

6.3 End-user output service (o)

So = n/ Do + r/ Ro
Where So = output size.
n = No's of data elements.
r = No's of reading references to entities.
Do = Data elements are found in the output BFC class.
Ro = Reading references are found in the output BFC class.
Fsu = functional sizing unit.

6.4 Data storage services (d)

Sd = n/Dd
Where Sd = entities size
n = No's of data elements (attributes).
Dd = Data elements are found in data storage BFC class.
Fsu = functional sizing unit.

6.5 Interaction to the other application (s)

Ss= n/Ds + r/Rs
Where Ss = size of interface to other Application.
n = No's of data elements.
r = No's of reading references.
Ds = Data elements are found in the BFC class.
Rs = Reading references are found in the BFC class.
Fsu =functional sizing unit.

6.6 Calculate the functional size of the application

The sum of all the BFC's classes are

S = Se + Si + So + Sd + Ss

6.7 Calculate the Estimated Effort

Estimated Effort = a_b (Functional size) b_b

Here a_b and b_b are constant. So the value of a_b = 2.4 and the value of b_b = 1.05. The value is used according to the model.

We have to scale the value of data elements (D) reading references (R) and writing references (W) according to the application.

We have validated the proposed model i.e. MEstimation model by comparing the effort involved in developing SIGRH mobile application[11] through estimating the effort using MEstimation model and FiSMA method and thereby calculating both the efforts with the actual effort required for the development of SIGRH mobile application. The effort required for the FiSMA method and the real effort is referred from [1]. Analysis of estimated effort of MEstimation model and the FiSMA model is given in table.

Table 2. Comparison of effort required for SIGRH application for various models

Real effort	FiSMA	MEstimation
103 h	152 h	104.4h

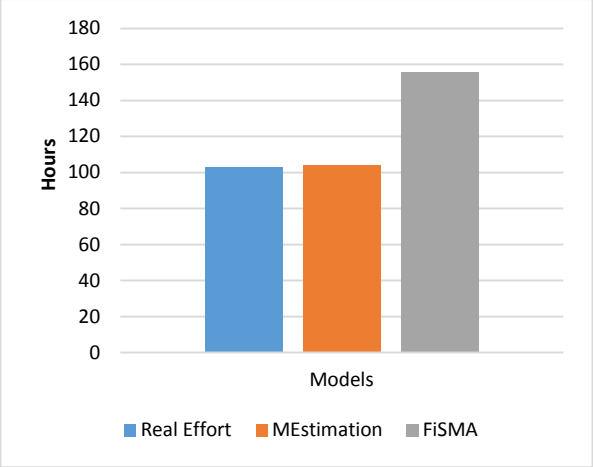

Fig. 2: Comparison graph

Table. 2 and Fig. 2, Shows that the effort required for the development of SIGRH application is closer the real effort required for the development of the application with MEstimation model rather than the FiSMA model. Therefore, we can conclude that MEstimation model is a better predicator of effort than the FiSMA model.

7. CONCLUSION

The results presented, are based on the literature review of the effort estimation methods and some areas of software engineering. There were some traditional estimation models - COCOMO, Functional point analysis, Mark-II. The traditional estimation models were very much complex, therefore measuring the size of software was also very difficult. It took lot of time and cost of development was also very high. So, we developed the mobile application models. The characteristics of mobile applications are distinguished from the traditional desktop applications. According to this paper, we have measured the functional size of the mobile application and calculate the estimated effort of the mobile application in the proposed model. Also, we have performed a comparison of the proposed model (MEstimation model), Real effort and FiSMA model in terms of effort.

Finally it is concluded that the proposed model (MEstimation model) shows the better predictions for the effort estimation as compared to the FiSMA model. There are many directions of the future work that are worth analyzing. We will do the evaluation of the new project with the different size of the application so that the proposed solution would be scalable. The improvisations can be done in the proposed model. Further, the scaling produced can be validated by taking opinion from experts in the field through questionnaire method.

8. REFERENCES

[1] Dehlinger, Josh, and Jeremy Dixon. "Mobile application software engineering: Challenges and research directions." *Workshop on Mobile Software Engineering*, 2011.

[2] Flora, Harleen K., Xiaofeng Wang, and Swati V. Chande. "An investigation on the characteristics of mobile applications: A survey study." International journal of information technology and computer science (Ijitcs) 6.11 2014.

[3] Longstreet, David. "Fundamentals of function point analysis." Longstreet Consulting, Inc. 2002.

[4] Boehm, B., Software Engineering Economics, Prentice-Hall, 1981.

[5] Putnam, L. and W. Myers, Measures for Excellence, Yourdon Press, 1992.

[6] Boehm, Barry, et al. "COCOMO II model definition manual." The University of Southern California, 1997.

[7] 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.

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

[9] 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.

[10] FiSMA, F. S. M. A. "Fisma functional size measurement method version 1-1." 2004.

[11] Maciaszek, Leszek A., and Joaquim Filipe, eds. Evaluation of Novel Approaches to Software Engineering: 9th International Conference, ENASE 2014, and Lisbon, Portugal, 2015.