On the applicability of COSMIC-FFP for measuring software throughout its life cycle

[1]Roberto Meli, Alain Abran[2], Vinh T. Ho[2], Serge Oligny[2]
[1]DPO Srl, Italy
[2]Dept. of Computer Science, Université du Québec à Montréal, Canada
roberto.meli@iol.it, abran.alain@uqam.ca, vho@lrgl.uqam.ca, oligny.serge@uqam.ca

Abstract

Software measurement plays a key role in software engineering and, to improve its performance, an organisation needs to measure software at each stage of the development life cycle. Recently, the COSMIC-FFP measurement method has been developed to improve the measurement of the functional size of a large array of software types. By quantifying software's functional user requirements, the method makes it possible to measure software from the user's viewpoint. The COSMIC-FFP measurement method has been designed based on a software functional model that can represent the functional user requirements at many levels of functional abstraction, such as software layers, functional processes and data movement sub-processes.

Developers in general, however, need to know the size of the software early in the development process to support the estimation and project planning process. While the measurement rules of the COSMIIC-FFP method have been designed to be applied when the details of the software functions are known, the method has the required flexibility to capture an estimate of the functional size of software early in the life cycle and to offer added value to the software engineers preparing the development plans. This paper investigates the applicability of COSMIC-FFP for measuring the size of software at early stages of the development life cycle.

1. Introduction

A requirement for software productivity analysis according to classic business models is the availability of the size of a software product from the user's viewpoint, that is, from a functional perspective rather than from a technical perspective [3]. Function Points Analysis (FPA) published in 1979 [4] is an example of a functional measure designed for MIS domain software, where it has been used extensively in productivity analysis and estimation [3].

When compared to technical measures of software, based, for example, on Lines of Code, functional size measures can be determined precisely when details of functional user requirements are identifiable. This is much earlier in the life cycle than technical measures, which cannot be determined until software coding has been done, a point much later in the development life cycle when only testing remains. Currently, functional size can be precisely obtained after functional specification is completed, by which time anywhere between 15 and 40% of the total work effort might have already been expended. There is, therefore, an obvious need to obtain such information earlier on for forecasting and planning purposes.

In this context, the Early Function Point Analysis technique has been originally developed in 1997 and subsequently refined [5] [6] [10] [11] to:

- Estimate, in the early stages of the software project life cycle, the functional size to be developed or maintained;
- Reduce significantly the time and cost involved in developing an estimate of functional size, in comparison with the time and cost required to performing detailed and precise measurements.

This technique, referred to as Early-FPA, has been successfully used for estimation in many organisations and countries. However, even though applicable to MIS software, there is still a need to extend it to a larger array of software types. The COSMIC-FFP measurement method has been developed to address this issue of functional size of various software types: real-time, technical, system and MIS software [1][2]. As a Functional Size Measurement method, the COSMIC-FFP method measures the size of software based on the functionality to be delivered to the user. It was designed based on a software functional model that can represent software at many levels of functional abstraction, such as software layers and functional processes. Software Functional User Requirements (FURs) are represented by a set of functional processes, each of which is a unique and ordered set of data movement sub-processes implementing a cohesive set of FURs.

The design of the COSMIC-FFP offers the flexibility that makes it possible to be applied much earlier in the development life cycle for estimation purposes. Inspired by the usefulness of the Early FPA method for measuring MIS software, we were interested in designing a new size estimation technique, Early & Quick COSMIC-FFP, based on the present COSMIC-FFP design to help estimate functional size at early stages of the development life cycle. Unlike the Early FPA method, which can be effectively applied only to measure MIS software, the proposed Early & Quick COSMIC-FFP can be applied to measuring a wide range of software (i.e. real-time, technical, system and MIS software).

This paper presents exploratory research towards the development of the new Early & Quick COSMIC-FFP size estimation technique based on the current COSMIC-FFP method. It is organised as follows: section 2 presents an overview of the COSMIC-FFP method; section 3 presents key concepts of the proposed Early & Quick COSMIC-FFP estimation method; and section 4 follows with a discussion on future research directions.

2. Overview of the COSMIC-FFP method

This section presents some of the key concepts of the COSMIC-FFP method [1][2]. The COSMIC-FFP measurement method consists of the application of a set of rules and procedures to a given piece of software in order to measure the functional size of the software. Two distinct and related phases are necessary to measure the functional size of software: mapping the functional user requirements embedded in the artefacts of the software to be measured onto the COSMIC-FFP software model and then measuring the specific elements of this software model (Figure 1).

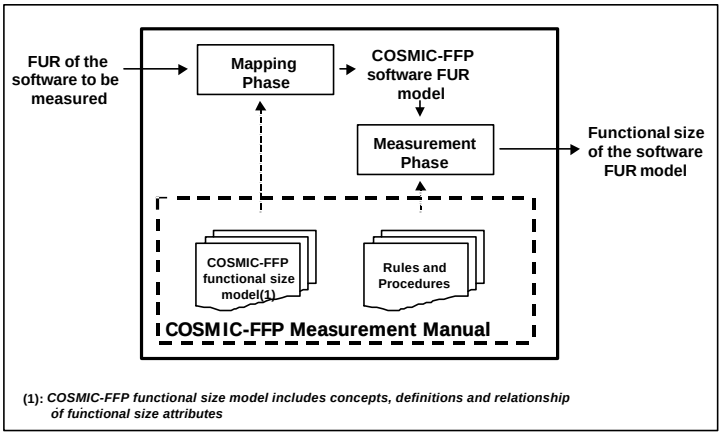

Figure 1. *COSMIC-FFP measurement process model [2]*

Prior to applying the measurement rules and procedures, the software to be measured must be mapped onto a generic model (the COSMIC-FFP software model – Figure 3) which captures the concepts, definitions and relationships (functional structure) required for a functional size measurement exercise. The COSMIC-FFP method measures the size of software based on identifiable functional user requirements. Depending on how these requirements are allocated, the resulting software might be implemented in a number of pieces. While all the pieces exchange data, they will not necessarily operate at the same level of abstraction. The COSMIC-FFP method introduces the concept of the software layer to help differentiate levels of abstraction of the FURs. An illustration of software layer configurations is presented in Figure 2. The functionality of a layer may be composed of a number of functional processes.

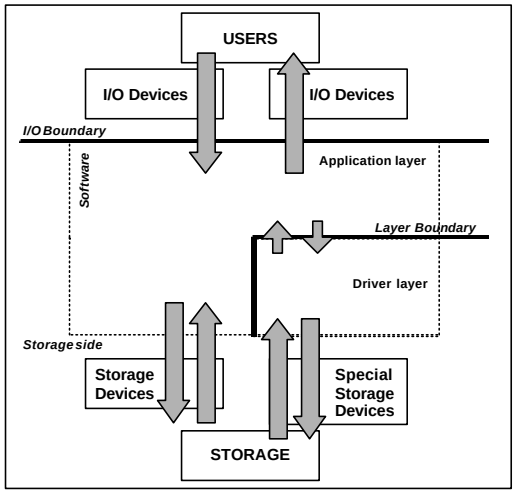

Figure 2. *An example of a configuration of functional layers [2]*

Software functional requirements are then implemented by a set of functional processes. Each functional process is a unique and ordered set of data movement sub-processes implementing a cohesive set of FURs. As indicated in Figure 3, the COSMIC-FFP generic software model distinguishes four types of data movement sub-process: in the "front end" direction, two types of movement (ENTRIES and EXITS) allow the exchange of data attributes with the users; in the "back end" direction, two types of

movement (READS and WRITES) allow the exchange of data attributes with the storage hardware ((Figure 3).

The COSMIC-FFP measurement rules and procedures are then applied to the software model in order to produce a numerical figure representing the functional size of the software. The unit of measurement is 1 data movement, referred to as 1 COSMIC Functional Size Unit, e.g. $1C_{FSU}$.

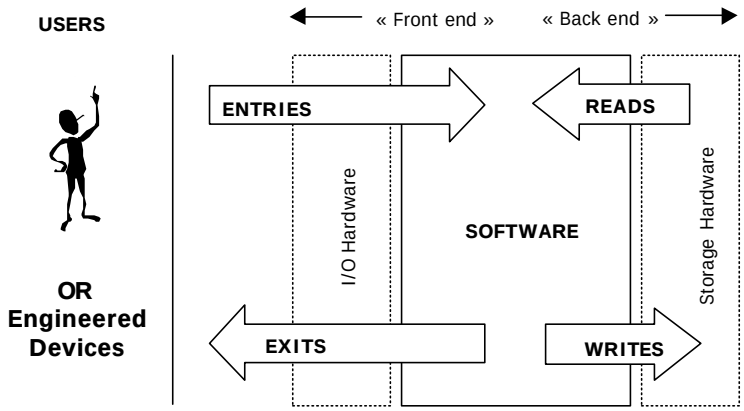

Figure 3. *COSMIC-FFP software model and data movement types [2]*

Conceptually, the mapping phase of the COSMIC-FFP method can be considered as a process of "viewing" a software from different levels of functional detail. First, the software is viewed at the highest level as composed of software layers, if applicable. Then, each software layer is viewed at a lower level of detail, i.e. functional processes. Finally, each functional process is in turn viewed at the lowest level of detail of interest for measurement with COSMIC-FFP, that is, sub-processes composed of data movement types. In the next section, we propose the key concepts enabling estimation, using the COSMIC-FFP model, of the functional size of software at early stages of the development cycle.

3. Basic principles of the Early & Quick COSMIC-FFP method

3.1. Size estimation at early stages of development

Functional size of software to be developed can be measured precisely after functional specification stage. However, functional specification is often completed relatively late in the development process and a significant portion of the budget has already been spent. In fact, if we consider the importance of an estimation with respect to project management, we face a curious and maybe paradoxical phenomenon: measurement would be very useful when we do not have enough elements to obtain it (feasibility studies), but when we can measure with the greatest accuracy (just before the final product is ready) it is no longer necessary for effort and duration prediction purposes. For estimation purposes, a trade-off between timeliness and accuracy is needed. If we need the functional size earlier, we must accept a lower level of precision since it can only be obtained from less precise information. Therefore, there is an urgency to finding a forecasting method for an early estimation of functional size (with any measurement method such as Cosmic-FFP) which can already be usable after the feasibility study phase. In this text, the term "estimation" refers to forecasting techniques, while the term

"measurement" is used for the application of the detailed measurement rules found in the COSMIC-FFP Measurement Manual.

3.2. Principles of the Early & Quick COSMIC-FFP method

The **Early & Quick COSMIC-FFP method** (E&Q COSMIC-FFP) has been designed to provide practitioners with an early and quick forecast of the functional size which can be used for preliminary technical and managerial decisions. Of course, a precise standard measure must always be carried out in the later phases to confirm the validity of decisions already taken. Here, "Early" means that we may obtain this value before having committed a significant amount of resources to a project; "Quick" means that we may also use the technique when the software is an existing asset and when constraints (such as costs and time) prevent a precise measurement.

It must be stressed, however, that an Early & Quick estimation must not be considered as an accurate measure of the software functional size, but only an estimation of that measure. In all cases where an inspection is requested, litigation is a possibility, an exact measure of the software is necessary, then the formal standards, such as COSMIC-FFP Measurement Manual, must be used.

The E&Q COSMIC-FFPA technique has been derived, with some important differences, from the analogous Early Function Point Analysis, already established in the area of estimation for the IFPUG standard.

The starting point for the reference framework of E&Q analysis is acknowledgement of the hierarchical structure in the functional requisites for a software application. This is not always true, but still very common at high levels of specification of requisites (sometimes software may be better logically modelled as a network of functions; if so, this must be taken into account when making adjustment to the size estimation model we are proposing).

When we document a software structure, we usually name the root as the application level and then we go down to defining single nodes, each one with a name that is logically correlated to the functions included. We reach the leaf level when we don't think it is useful to proceed to a further decomposition. An important property of this structure (actually a product breakdown structure) is that all the functions provided by the application must be in the leaves, that is, there must be no functionality in a node that is not present in an explicit component of that node (a leaf or another node). This implies that if we have a measure of all the leaves we have a measure of the whole tree.

In the COSMIC-FFP model [2], the leaves are the Functional Processes which are defined as: "…a unique and ordered set of data movements (entry, exit, read, write) implementing a cohesive set of Functional User Requirements. It is triggered by an event and, once performed, must leave the software in a coherent state with respect to the triggering event."

Considering those leaves, we apply the measurement rules and then we obtain the final measure for the software item under measurement.

On the one hand, in the early stages it is not possible to distinguish the single data movements (BFC – Base Functional Component) because the information is not available at this level of detail. On the other hand, however, the preliminary hierarchical structure of the software shows as leaves what are actually nodes in the detailed version. What is required early on in the life cycle is, then, to assign forecasts of average process size, in

C_{FSU}, at the intermediate and top levels in such a way that the final result will be obtained by the aggregation of the intermediate results.

If we may use a metaphor, the C_{FSU} is to software what the square meter is to buildings, the entire application is like an apartment and the intermediate nodes are like rooms or even aggregations of rooms (i.e. the night zone). We may then determine the dimensions of the apartment in two different ways: using a tape measurer and obtaining a precise measure of the whole; or counting the number of rooms and how many there are of each, and estimating the total area using only average measures for each type of room. This is what is done when assigning C_{FSU} weights to each kind of software object in the hierarchical structure.

The estimation is based on the capability of the estimator to "recognize" a software item as belonging to a particular topology; an appropriate table, then, allows the estimator to assign a C_{FSU} weight (an average, but it could also be a minimum or a maximum) for that item.

The software items to estimate are identified in the same manner as with the standard measurement method (which we will not discuss here); then, boundary and layer details are considered. After the identification has been completed, it will be possible to apply the concepts of E&Q COSMIC-FFP.

One advantage of the technique is that estimates can be based on different and non homogeneous levels of detail in the knowledge of the software structure. If a part of the software is known at a detail level, this knowledge may be used to estimate it at the Functional Process level, and, if another part is only superficially known, then a higher level of classification may be used. This property is better known as multilevel estimation.

3.3. Similarities and differences between IFPUG EFPA and E&Q COSMIC-FFP

Let us look at similarities and differences between the IFPUG EFPA (referred to as Early IFPUG) and the E&Q COSMIC-FFP (referred to as E&Q COSMIC):

Similarities:

- Software is modeled as a hierarchy of functions (processes)
- A direct weight in terms of "points" may be assigned to those functions
- The technique is based either on an analytical approach (the use of a composition table) and on an analogy-based one (the identification phase)
- Analogy may be used to identify/classify hierarchical items
- Analogy may be used with respect to an abstract model or to a concrete set of software objects actually collected and classified

Differences:

- IFPUG: the IFPUG-defined Elementary Processes measured are the EI, EO and EQ
- COSMIC: the data movements measured are components of a Functional Process
- IFPUG: the IFPUG Elementary Processes are step functions in terms of FP, with lower and upper bounds (weights from 3 to 9).

- COSMIC: Functional Processes can be assigned smaller to larger size (from 1 C_{FSU} to no theoretical limit - they are not bounded, but in practice they will have some sort of "natural" upper boundary)
- Early IFPUG: Logical data groups are modeled and counted
- E&Q COSMIC: Logical data are not explicitly taken into account

These differences have lead to the implementation of some differences in the E&Q COSMIC technique:

- The term Process is used extensively across the levels of the hierarchy
- Functional Processes do not need to be classified as EI, EO or IQ
- Functional Processes are classified as Small, Medium or Large, depending on their estimated number of sub-processes
- Logical Data are not taken into account

The following considerations have also been taken into account:

- The numerical relationship among the different software objects at each level of the hierarchy has been kept stable to facilitate the analogy among the EFPA and the new E&Q COSMIC – i.e. a Small Macro Process is composed by 2-3 General Processes. The number of C_{FSU} in the corresponding boxes will be determined from case studies.
- It has been necessary to implement an adjustment factor applicable at each level of the structure (above the Functional Processes) to calibrate the C_{FSU} with respect to the estimated mixture of Functional Processes contained in the software object considered. The basic estimation assumption is that the mixture is small FP 33%, medium FP 33%, large FP 33%. This assumption may vary as soon as empirical data are collected. In any case, if the estimator suspects that the mix may be significantly different from the proposed one, then the C_{FSU} assigned to that object must be adjusted by a corrective factor based on the distance between the mixes.
- In the future, we will perhaps be able to assign a standard mix to any particular software domain, thereby improving the quality of the basic estimation.

3.4. The types of functions in E&Q COSMIC-FFP

A Function is a set of actions performed by the software system, enabling the user to fulfil a logical objective. A Functions is classified, in order of increasing magnitude, as a Functional Process, a General Process, or a Macro-Process:

a) A Functional Process (FP) is the smallest process, performed with the aid of the software system, with autonomy and significance characteristics. It allows the user to attain a unitary business or logical objective at the operational level. It is not possible, from the user's point of view, to proceed to further useful decomposition of a Functional Process without violating the principles of significance, autonomy and consistency of the system. A Functional Process can be Small, Medium or Large, depending on its estimated number of Base Functional Components (E,X,R,W).

b) A General Process (GP) is a set of medium Functional Processes and can be likened to an operational sub-system of the application in order to allow for an organised whole responding to the user's objectives to be achieved. A

General Process can be Small, Medium or Large, depending on its estimated particular number of Functional Processes (FP).

c) A Macro-Process (MP) is a set of medium General Processes, and sometimes it can itself constitute a whole development project. It may be likened to a relevant sub-system of the overall Information System of the user's organisation. A Macro-Process can be Small, Medium or Large, depending on its estimated number of General Processes (GP).

In the preceding set of processes, the second level is built up on the basis of the first one, and the third level is built up on the basis of the second one. There is a 4th type of process which is, in a certain sense, off-line from the hierarchical structure outlined. We call it a Typical Process.

d) A Typical Process (TP) is just the set of the four frequently used Functional Processes, which are: Create, Retrieve, Update and Delete (CRUD) data in one or more logical files. Usually, these aggregations are associated with the term "Management of …", meaning that any particular file or group of files should be subject to all the described operations to make it possible to usefully employ the software application.

From a theoretical point of view this is just a particular case of the General Process (because it is made up of Functional Processes) but it may be usefully considered on its own since we frequently explicitly meet that kind of aggregation in the user requirements.

3.5. Numerical assignments

Each E&Q COSMIC-FFP element is defined in terms of sub-components (see Table 1,2 & 3 where the first one is still in a trial phase) and is associated with three values in terms of C_{FSU} (minimum, most likely and maximum). These assignments are currently subject to definition and trial on the basis of the data collection activity and statistical analysis for actual projects in the Field Trial Phase of the COSMIC-FFP method. The ultimate aim of the E&Q COSMIC-FFP technique is to attain the closest approximation to CFSU values and therefore improvements are to be expected.

***Table**. 1*

FP (Functional Process)	**# BFC in C_{FSU}**
Small	N.A.
Medium	N.A.
Large	N.A.

***Table**. 2*

GP (General Process)	**# FP (Functional Processes)**
Small	6-12
Medium	13-19
Large	20-25

***Table**. 3*

MP (Macro-Process)	**# GP (General Process)**
Small	2-3
Medium	4-7
Large	8-12

4. Conclusions

4.1. Expected results

The E&Q COSMIC technique proposed here is at early stage of development. Further work will be require to pursue the following goals:

- To produce estimates closely related to the precise measurement results later in the life cycle when more functional details are available;
- To be robust enough, in the sense that for the same item different estimators would produce estimate within the same range;
- To be as simple as possible for large uptake by practitioners;
- To be really quick in its practical implementation;
- To be inexpensive, for its intended purpose and range of accuracy.

4.2. Research directions

4.2.1. Validation of the basic assumption and principles

From the practical application of the E&Q COSMIC-FFP technique we expect to validate the basic principles shown in this research paper. In particular, we will collect feedback about the expected results quoted above.

4.2.2. Fine-tuning of the basic numerical values

Further data collection is required for fine-tuning of the numerical values assigned to the different elements, in particular for the minimum, most likely and maximum values for each type of process and for the appropriate mix of functional processes.

4.2.3. Eventual quantification of the relationships among BFCs

Data are currently being collected and analysed to investigate the potential correlation among different BFCs in specific software domains. For example, some correlation may be observed between number of Entries and Exits and number of Reads and Writes for a particular type of process. If they exist, such correlations may be included in the technique to achieve better results.

4.2.4. Eventual correlations between software domains and the basic mix of processes

From empirical evidence, we might draw some conclusions about the eventual correlation between software domains and the basic mix of processes in such a way that high-level estimates may be more precise and more tuned to the specific object of interest.

5. References

[1] Abran, A., “FFP Release 2,0: An Implementation of COSMIC Functional Size Measurement Concepts”, Presented at FESMA 99, Amsterdam, October 4-7, 1999.

[2] Abran, Desharnais, Oligny, St-Pierre, Symons, “COSMIC-FFP Measurement Manual, version 2.0”, Ed. S. Oligny, Software Engineering Management Research Lab., Université du Québec à Montréal (Canada), October, 1999.

[3] Abran, A., Robillard, P. N., “Function Points Analysis, An Empirical Study of its Measurement Processes”, IEEE Transactions on Software Engineering, vol. 22, no. 12, pp. 895-909, December 1996.

[4] IFPUG, “Counting Practices Manual rel. 4.1”, International Function Point Users Group

[5] Meli, R., “Early Function Points: a new estimation method for software projects”, ESCOM 97, May 1997, Berlin.
[6] Meli, R., “Early and Extended Function Points: a new method for Function Points Estimation”, IFPUG Fall Conference, Scottsdale, Arizona USA, 15-19 September 1997.
[7] Meli, R., “Risks, requirements and estimation of a software project”, ESCOM-SCOPE 99, Herstmonceux Castle, East Sussex, England, April 27-29, 1999.
[8] Meli, R., L. Santillo, “Function point estimation methods: a comparative overview”, FESMA 98 - The European Software Measurement Conference – Amsterdam, October 6-8, 1999.
[9] Meli, R., “Human factors and analytical models in software estimation: an integration perspective”, ESCOM-SCOPE 2000, Munich, April 18-20, 2000.
[10] Santillo, Luca and Meli, Roberto, “Early Function Points: some practical experiences of use”, ESCOM-ENCRESS 98, Roma (Italy), May 18, 1998.
[11] Santillo, Luca, “Early Function Point Estimation and the Analytic Hierarchy Process”, ESCOM-SCOPE 2000, Munich, April 18-20, 2000.