A Case Study on Functional Size Measurement Based Effort Estimation for Embedded Systems

Ali Nazima Ergun¹, Cigdem Gencel²

¹KAREL ARGE, Cyberpark, Bilkent, ²Middle East Technical University, Ankara, Turkey
{ali.ergun@karel.com.tr, cgencel@metu.edu.tr}

Abstract. There have been many attempts at measuring software size based on the amount of functionality laid out on project requirements. Since projects that triggered the need for proper software management have been typically database driven enterprise software, most estimation methods were well suited to analyzing such systems and the question of how to analyze embedded systems has only begun to be explored since late nineties. This study discusses the results of the implementation of two ISO certified Functional Size Measurement (FSM) methods; MkII FPA and COSMIC FFP in a multiple-case study involving four projects; one data-intensive and three embedded projects completed at a software company in Turkey. Based on the functional size figures and the actual development effort, the productivity delivery rates of the company are derived and discussed.

1. Introduction

Since the early 1980s great effort has been put forth to identify and fine tune the “software process” and its proper management. Software community has developed unique tools and techniques for size, effort, and cost estimation to address challenges facing the management of software development projects [23][32][33].

Since software size is one of the key measures for most effort and cost estimation methods, a significant amount of these efforts have been put on software size measurement. Earlier size measurement methods are based on Source Lines of Code (SLOC) of software, however, this information is not available until after the project is completed.

Recent methods attempt at measuring project size by trying to capture the amount of functionality laid out on project requirements, which are available earlier on in the project lifecycle. The topic of Function Point Analysis (FPA) evolved quite a bit since the introduction of the concept by Albrecht in 1979 [1]. Many variations and improvements on the original idea were suggested [32], some of which proved to be milestones in the development of Functional Size Measurement (FSM) (see Section 2)
In 1996, International Organization for Standardization (ISO) established the common principles of FSM methods and published ISO/IEC 14143 standard in order to promote the consistent interpretation of FSM principles.

Currently, MkII FPA, the International Function Point Users Group (IFPUG) FPA, the Common Software Metrics International Consortium Full Function Points (COSMIC FFP) and the Netherlands Software Metrics Association (NESMA) FSM methods have been certified as international FSM standards by the International Organization for Standardization (ISO).

Due to these constructive progresses, FSM has begun to be applied more and more worldwide. The number of benchmarking data on the projects which were measured by FSM methods has significantly increased in well-known and recognized benchmarks such as the one by ISBSG with more than 4,100 projects.

On the other hand, one of the major uses of software size is its use in software effort estimation. There already exist a number of studies with an emphasis on the relationship between software size and effort (see Section 2). However, although different estimation models are reported to be successfully used in certain environments, none of them have been deemed universally applicable by the software community.

Therefore, until such a widely accepted estimation model is developed, it is usually recommended to use the conventional approach to estimate effort, which takes into account two key components: the size of software and the rate of work, also called the ‘Productivity Delivery Rate (PDR)’. The effort is estimated based on the size of the software and the PDR. However, this brings the software organizations a challenging requirement to identify their PDR values for specific types of projects or use industrial average values taken from the benchmarking datasets. However, in order to use the second choice, the benchmark dataset shall involve a very similar project, which is usually not available.

This paper draws upon our case study on four projects completed at Karel Arge, which we used to pit Mark II FPA, a popular method in Turkey and widely required in defense contracts, against COSMIC FFP, to explore the suitability of these methods for the company, which predominantly develops real-time embedded applications. The average PDR values for the company are derived based on the results of the case study.

The related research on software functional size measures and methods are briefly summarized in the second section. In the third section, the description of the case projects, the application of Mk II FPA and COSMIC FFP to these case projects, the results obtained and the comparisons of these two methods are presented. The discussions on the results of the case study are presented in the last section...

2. Background

The 1970s saw the emergence of effort estimation as a serious challenge in software engineering.

COCOMO, or Constructive Cost Model, an SLOC based size, effort and cost analysis is proposed by Boehm [3]. COCOMO was soon followed by other methods...
such as COSYSMO [38], Parametric Review of Information for Costing and Evaluation – Software (PRICE-S) [26], and Software Evaluation and Estimation of Resources – Software Estimating Model (SEER-SEM) [9].

SLOC based estimation models relate SLOC of software to effort spent at development, via an exponential equation whose parameters are derived from statistical studies of relevant project development data sets. However there are certain disadvantages with using SLOC as a basis of estimation; it can not be normalized across platforms, languages or organizations, and most importantly it can be measured only after the code is available.

Albrecht’s 1979 proposal [1] for estimating the functional size became a serious contender for software size measurement and hence effort estimation. During the following years, variations of the original method have been developed. Some of them either provided unique viewpoints different from the dominant method of their time or extended the applicability of FSM methods to different functional domains in addition to business application software. Other methods narrowed down the functional domain measured to better analyze specific software, such as object oriented methods. Regardless, currently only the four methods are certified by the ISO, which are MkII FPA [34], IFPUG FPA [35], COSMIC FFP [36] and NESMA FSM [37] methods.

Albrecht’s original idea has become the basis for IFPUG FPA [35], one of the earliest ISO standardized FSM methods [19]. IFPUG FPA enjoys widespread popularity and large publicly available data sets for those who wish to train their own company-specific IFPUG model, or to compare their measurements with others. It is based on the idea of measuring the amount of functionality delivered to users in terms of Function Points (FP). IFPUG FPA was mainly developed to measure data-strong systems such as Management Information Systems (MIS).

MkII FPA was developed by Symons in 1988 in order to improve the original FPA method [31]. He brought some suggestions to reflect the internal complexity of a system. Currently, the Metrics Practices Committee (MPC) of the UK Software Metrics Association (UKSMA) is the design authority of the method [34]. It was mainly designed to measure business information systems. It can also be applied to other application domains such as scientific and real-time software with some modifications. MkII FPA has been approved as being conformant to ISO/IEC 14143 and become an international ISO standard in 2002 [18]. This method views the system as a set of Logical Transactions (LT) and calculates the functional size by counting the Input Data Element Types (DETs), Data Entity Types Referenced and Output Data Element Types for each LT.

NESMA FPA [37] is also based on the principles of the IFPUG FPA method. The function types used for sizing the functionality are the same as IFPUG FPA. The difference between these two methods is that the guidelines of NESMA give more concrete guidelines, hints and examples. It was certified by ISO in 2005 [21].

COSMIC FFP [36], on the other hand, is a fairly recent method, gaining ground in the international community, thanks to its ability to measure real-time systems, as opposed to earlier variants which shined in measuring data intensive MIS software. COSMIC FFP has been approved as being an international ISO standard in 2003 [20]. COSMIC FFP was designed to measure a functional size of software based on the

---

1 Please refer to [8][8] for a comprehensive summary of these methods.
count of four types of data movement, i.e. the number of Entry, Exit, Read, and Write operations.

In parallel to these developments in FSM, significant research has been going on about software effort estimation models and techniques, which are based on software size [3][4][10][22][24][25]. COCOMO II [4], the revised version of the original COCOMO, provide for measuring functional size and converting this result to SLOC. However, this technique, called ‘backfiring’ still cannot account for the extra uncertainty introduced by adding another level of estimation [7][29][30].

In a number of studies such as [2][5][6][27], the related works on estimation models are assessed and compared. However, the common conclusion of these studies is that although different models are successfully used by different groups and for particular domains, none of them has gained general acceptance by the software community. Effort estimation based on the functional size figures have just begun to emerge as more data are collected in benchmarking datasets, which would help identifying the relationship between functional size and effort.

3. Case Study

In this case study, we explore the suitability of the two ISO certified FSM methods; MkII FPA and COSMIC FFP for measuring the software projects of Karel Arge. We evaluate the chosen methods in measuring the functional size of one data-intensive and three embedded projects already completed. We also explore the success of the effort estimation by examining the usefulness of the relationship between the functional size measured by the individual methods and the actual effort spent.

3.1 Description of the Software Organization

Karel Arge, founded in 1986, is a local and regional telecommunications company, which designs and manufactures PBX systems. Karel Arge was spun-off from Karel in 2005, and performs R&D of PBX systems and telecommunication products. Its R&D capability is also available for outsourcing in electronic control and military projects. A majority of the projects are real-time embedded systems and are often accompanied by data intensive GUI or client/server applications. In 2006, Karel Arge decided to define its software processes to conform to CMMI (Capability Maturity Model Integrated) level 3.

3.2 Description of the Case Projects

We have picked four projects undertaken at Karel Arge for our case study. Since the majority of the projects within the company are embedded in nature, we selected three embedded projects already completed. Since database type projects are not completely absent, one such project is also included along with the embedded projects, to
compare how well these estimation methods apply to non-embedded projects undertaken at Karel Arge.

It is important for the company to also accurately estimate these non-embedded kinds of end-user projects as they are usually a sub-component of the software product together with an embedded part, and estimating only the embedded part of a project will not be useful.

A short description of each project is as follows:

**Project A - SMS Gateway (Data Intensive):** Database centric GUI running on a PC that manages sending and receiving of SMS messages through specifically designed gateway hardware. All stages of this project (from requirements analysis to design, coding and testing) were completed by a single senior software designer using Borland Delphi 2005 utilizing 646.9 person-hours.

**Project B – Fridge and Freezer Combo (Real-time):** A microprocessor monitors certain switches and temperature sensors to control heaters and compressors within various cooling and defrost cycles. The program cycles are rigorously algorithmic. All stages of this project were completed by a single experienced software designer using C utilizing 538 person-hours.

**Project C – Combo Boiler (Real-time):** Simple, user programmable residential heater. User can program multiple on and off schedules via buttons and a simple display. All stages of this project were completed by a single senior software designer using C utilizing 132 person-hours.

**Project D – Shower Control (Real-time):** Microcontroller based system controls shower heat and water flow per user input. Monitors sensors, controls heaters and relays. System behavior is explained in the requirements over a state machine, which points to a considerable algorithmic element. All stages of this project were completed by a single senior software designer using C utilizing 40 person-hours.

All projects were completed by different engineers; however the engineers who have worked on Projects B, C and D work together on many other projects and most likely have similar design practices.

For measuring the functional sizes of the projects, we picked MkII FPA and COSMIC FFP. COSMIC FFP is the first ISO standardized method with a clear emphasis on measuring real-time systems and MkII FPA counts are widely required from defense contractors, which set the bar in Turkish software development practices nationwide.

### 4. Case Study Conduct and Results

All the functional size measurements were taken by the authors of this study, one of which is working for the company and has the domain knowledge.

**Project A - SMS Gateway:** The development effort utilized for the project is 646.9 person-hours and the length of code is 20610 SLOC. The functional size measurement results by COSMIC FFP and MkII FPA, the PDR values and SLOC-Functional size ratio for the project are given in Table 1.
Table 1. SMS Gateway Measurement Results

<table>
<thead>
<tr>
<th>FSM Method</th>
<th>Functional Size</th>
<th>PDR (Effort/Functional Size)</th>
<th>SLOC/Functional Size</th>
<th>Measurement Effort (person-hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MkII FPA</td>
<td>223.84</td>
<td>2.89</td>
<td>92.07</td>
<td>2.33</td>
</tr>
<tr>
<td>COSMIC FFP</td>
<td>168.00</td>
<td>3.85</td>
<td>122.68</td>
<td>0.83</td>
</tr>
</tbody>
</table>

As a project structured in a well defined database with entities and relationships, the SMS Gateway’s MkII counting was easy and intuitive. COSMIC analysis was also easy to do, however counts only included Read and Write accesses to a data type but not the number of members modified. This ignores complexities especially evident in database systems, as the number and the type of members modified change the complexity of the database query associated with the given transaction.

Total effort spent for COSMIC implementation was one and a half hours quicker as transactions were already laid on during MkII analysis.

Project B – Fridge and Freezer Combo: The development effort utilized for the project is 538 person-hours and the length of code is 2044 SLOC. The functional size measurement results by COSMIC FFP and MkII FPA, the PDR values and SLOC-Functional size ratio for the project are given in Table 3.

Table 2. Fridge and Freezer Combo Measurement Results

<table>
<thead>
<tr>
<th>FSM Method</th>
<th>Functional Size</th>
<th>PDR (Effort/Functional Size)</th>
<th>SLOC/Functional Size</th>
<th>Measurement Effort (person-hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MkII FPA</td>
<td>75.98</td>
<td>7.08</td>
<td>26.90</td>
<td>2.00</td>
</tr>
<tr>
<td>COSMIC FFP</td>
<td>119.00</td>
<td>4.52</td>
<td>4.52</td>
<td>0.33</td>
</tr>
</tbody>
</table>

Modeling an embedded system with an E-R diagram turned out to be a major difficulty. A useful analogy assumed was considering the microprocessor as the user and sensors, triggers etc as the interface. Timers and similar asynchronous interrupts frequent in embedded applications still proved to be problematic. Defining entities, as part of the E-R diagram, were highly subjective; there is no record based database, hence everything could be considered within a single entity. And it would not matter in reality, as an inter-entity complexity does not exist as it exists in databases. Hence, MkII FPA’s counting of entities referenced (which increases complexity significantly) does not really apply well to embedded systems.

Applying COSMIC on the other hand was very easy and intuitive.

Although Project B was smaller than Project A, MkII FPA implementation took six times longer as modeling an embedded system as a database proved to be quite a challenge.
**Project C - Combo Boiler:** The development effort utilized for the project is 132 person-hours and the length of code is 1151 SLOC. The functional size measurement results by COSMIC FFP and MkII FPA, the PDR values and SLOC-Functional size ratio for the project are given in Table 3.

<table>
<thead>
<tr>
<th>FSM Method</th>
<th>Functional Size (person-hours)</th>
<th>PDR (Effort/Functional Size)</th>
<th>SLOC/Functional Size</th>
<th>Measurement Effort (person-hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MkII FPA</td>
<td>54,40</td>
<td>2,43</td>
<td>2,43</td>
<td>0,60</td>
</tr>
<tr>
<td>COSMIC FFP</td>
<td>79,00</td>
<td>1,67</td>
<td>1,67</td>
<td>0,30</td>
</tr>
</tbody>
</table>

E-R modeling for MkII FPA was very counter-intuitive and the analysis failed to capture much of the algorithm stated in the requirements. Along with a proper data structure modeling, COSMIC analysis yielded functional size of 80 Cfsu. COSMIC FPA was suitable to analyzing this project. However, being a small project, Project C had little movement of data, and so the algorithmic component had more prominence. Excluding the algorithmic complexity arguably resulted in an oversimplification of size measurement.

**Project D - Shower Control:** The development effort utilized for the project is 40 person-hours and the length of code is 432 SLOC. The functional size measurement results by COSMIC FFP and MkII FPA, the PDR values and SLOC-Functional size ratio for the project are given in Table 4.

The E-R modeling required for an MkII FPA was very hard and failed at capturing the algorithm. However MkII FPA was better than COSMIC analysis in capturing interface actions (counting actual number of interface parameters changed, i.e. two motors controlled rather than counting one motor control access). COSMIC FP functional size was obtained as 53 Cfsu. The analysis was simple and straightforward.

<table>
<thead>
<tr>
<th>FSM Method</th>
<th>Functional Size (person-hours)</th>
<th>PDR (Effort/Functional Size)</th>
<th>SLOC/Functional Size</th>
<th>Measurement Effort (person-hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MkII FPA</td>
<td>65,04</td>
<td>0,62</td>
<td>0,62</td>
<td>0,80</td>
</tr>
<tr>
<td>COSMIC FFP</td>
<td>53,00</td>
<td>0,75</td>
<td>0,75</td>
<td>0,30</td>
</tr>
</tbody>
</table>

Fig. 1 represents the relationship between the logarithmic transformations of Functional size vs. actual development effort in all projects.
The results show that, Functional Size measured in COSMIC FFP results yield a desirable, approximately linear graph, and are promising for further analysis. MkII FP counts, on the other hand, show an irregular graph. In this case project, the number of data points is very few to draw conclusions on the strength of the relationship between functional size and effort, which is statistically significant. Therefore, we evaluate the methods based on each project.

The irregular shape of MkII resulted from Project 4- Shower control application. This might be attributed to the “advantage” we thought MkII had on measuring the shower control project, that it was more intuitive than COSMIC FFP on counting actual number of interface parameters changed, i.e. two motors controlled rather than counting one motor control access. Embedded implementations are likely to minimize data structures by concatenating similar information into one structure, for example two motors always controlled together will be represented by a single status, which might have four possible on/off combinations for both motors, instead of having two separate on/off status information for each motor. This implementation decision reflects upon MkII results while it is irrelevant in terms of data movement and hence does not affect final CFSU. This is a good example on how COSMIC FFP is more easily applied to real-time embedded system analysis; the results are independent of how data is stored, which is indeed irrelevant in terms of functional complexity in embedded systems as opposed to MIS systems where number of records and overall database structure effect processing of all queries.

Table 5 summarizes the PDR Values for all the projects.

<table>
<thead>
<tr>
<th>Table 5. The PDR Values for Karel Arge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>PDR (Effort/MkII FP)</td>
</tr>
<tr>
<td>PDR (Effort/CFSU)</td>
</tr>
<tr>
<td>PDR (Effort/SLOC)</td>
</tr>
</tbody>
</table>
We based our results on merely four samples. However, in order to use the PDR values to estimate effort in Karel Arge, more projects’ data is required to come to an average PDR value.

5. Discussion and Conclusion

We have examined measuring three real-time projects and one data-intensive project using MkII FPA and COSMIC FFP. Applying MkII FPA typically requires at least a simple entity-relationship analysis. Forcing such a database oriented model on real-time projects’ requirement set is counterintuitive and the E-R models derived are highly subjective. COSMIC FFP, on the other hand, can be applied to both types of projects; however COSMIC FFP does look very much like MkII FPA when used on database-centric projects since data movements mimic the underlying record accesses. The final “counts” would be approximately identical had not MkII FPA assigned weights to different type of record accesses while there is no weight in COSMIC FFP’s measured data movements.

It would be interesting, then, to work both methods within a set of projects with a majority of data-intensive choices. This would compare the consistency of both methods on sizing data-intensive projects and essentially show the effectiveness of weights assigned by MkII FPA to elementary accesses.

Focusing back to our current results, COSMIC FFP is a major step ahead in measuring size for real-time embedded systems. We can extract a mathematical relationship between effort and COSMIC functional size. However if a rigorous cost estimation is required, using a formal method such as COCOMO II, the COSMIC FFP values might be substituted for SLOC, our results show that COSMIC functional size’s correlation with actual effort is higher than that of SLOC’s. COSMIC functional size-effort correlation curiously holds across different type of projects, while SLOC values drastically change as we move into the data-intensive domain where actual implementation practices also change drastically. This is not to say, though, that companies should use a single model for all projects they work on just because COSMIC FFP performs reasonably well across project domains. The most effective usage might be to analyze control-intensive projects, data-intensive projects, algorithm intensive projects, etc. separately. One might choose to use COSMIC FFP for embedded projects, MkII FPA for data-intensive projects, and perhaps a completely different method for hybrid projects.

COSMIC FFP definitely is promising in that it can be applied to both project domains, however an extension for algorithm intensive systems is still due. Perhaps it would be better to use the best of each world instead of searching for a universal method that can be applied to all projects.

REFERENCES


[37] The Netherlands Software Metrics Association (NESMA).: NESMA Definitions and Counting Guidelines for the Application of Function Point Analysis, v.2.0 (1997)