Exploration of a Method for COSMIC Size Estimation from S-BPM

Murat Salmanoğlu¹, Onur Demirörs¹, Oktay Türetken²

¹Informatics Institute, Middle East Technical University, Ankara, Turkey
²Industrial Engineering & Innovation Sciences, Technische Universiteit Eindhoven

Abstract. Effort is the main cost in software development projects, and software size is the main input for effort estimation. Project managers need to have sound size estimations to estimate required software effort. The inputs for size estimation models are generally software artifacts that are produced later in software development life cycle; however project managers need the estimations prior to the project start date. In this research we explore a method to estimate COSMIC functional size by using S-BPM diagrams that may be created before the project’s start date. We tested our method with a case study and observed a 5% estimation error with respect to COSMIC size measurements.

Keywords: Software size estimation, software size estimation, COSMIC, S-BPM, Software project management.

1 Introduction

Effort is the main cost item for software development projects and for good management practices effort estimation is one of the main activities of software project manager. To make realistic effort estimations, managers need to assess the size of the software, however measuring software size accurately at the beginning of the project, when the artifacts like requirements documents or use cases are not ready, is not possible. There are methods to estimate (instead of measure) the size of the software to be developed to give project manager the opportunity to use and make effort estimations, but they also require some project artifacts to be ready.

In this research we propose and explore a method to estimate functional software size at the beginning of the software development lifecycle by using Subject Behavior Diagrams (SBD) of Subject-Oriented Business Process Management (S-BPM), a recent approach in the business process management field.

Functional size measurement is one of the most used approaches for measuring software size. It is introduced to software world by Albrecht [1]. It aims to measure the functionality of the software using different artifacts produced in the software life cycle. There are several methods to measure functional size, the most commonly used
methods are IFPUG [2], MkII [3], COMIC [4], NESMA [5], FISMA [6]. These functional size measurement methods are accepted by ISO and are widely used throughout the world by software practitioners. In this study, we used COSMIC functional size measurement method to estimate the functional size from subject behavior diagrams. COSMIC is the most commonly used method after IFPUG [7] and better suited for automated measurement as it has been shown in previous works [8].

COSMIC measures the functionality of the software by counting data movements in and out of the software boundaries [9]. There are four types of data movements defined in COSMIC: Entry (E), eXit (X), Read(R), and Write (W) (Fig. 1). Entry is a data movement from the user to the functional process in the software, eXit is a data movement from the functional process to the user, Read is a data movement from persistent storage to the functional process, and Write is a data movement from the functional process to the persistent storage. Before starting measurement, measurer should first define Functional User Requirements (FUR), than Functional Processes (FP) of these requirements. Functional Processes includes Object of Interests (OOI) and Data Groups (DG), DG is a distinct, non-empty, non-ordered and non-redundant group of attributes related with one OOI and they represent the data that move with the Data Movements (DM).

COSMIC measurement can be conducted in different phases of development by using different types of artifacts like requirement specification, design documents, user manuals, and graphical user interface. There are some methods suggested for early size estimation in COSMIC [10] and in IFPUG [11], however these early estimation methods are usually criticized of being subjective. Buglione et. al. [12] introduced a method to estimate functional size from Project Size Unit (PSU) at the beginning of the project. PSU can be calculated at the beginning stages of the project and functional size can be estimated from PSU by using historical data of the organization. There are other successful methods suggested to measure functional size by using UML.
These methods use UML to calculate functional size. These available methods help to measure of estimate software size early in the lifecycle, however, we want to build a model which will help us to estimate software size accurately and objectively even before the development project starts. Process models are ideal to use for our purposes as they are constructed usually during organizational process improvement activities and are readily available before software development project begins.

There are studies focusing on size estimation from process models; Kaya suggested a model to estimate software functional size from eEPC diagrams [16], Monsalve et. al. [17], [18] used Qualigram and BPMN for functional size estimation and reached promising results. The research of Aysolmaz et. al. [19] suggested a method to make effort estimation directly from BPM measures with statistical methods skipping the size estimation phase. All these research show that functional size can be estimated from process models before starting a development projects. The improvement we want to introduce with our study is subject perspective. We want to focus on the users of the processes in our process models and estimate functional size from subjects’ view points. To reach this goal we will use S-BPM.

We want to build a model specifically for S-BPM, because we predict that S-BPM has significant potential for size estimation as it focuses on users of the processes as a modeling approach. Our hypothesis is that we may easily isolate the target software system and its users from other participants of the business processes by using subject behavior diagrams of S-BPM and this approach may result accurate size estimation. In this study we want to explore and test the possibility of this hypothesis.

We will give the details of our proposed method in section 2 and after that in section 3 we will explain and discuss our case study that we conducted to test the accuracy of suggested method. Finally, section 4 will summarize our findings and provide a roadmap for future studies.

2 Proposed Method

With our method we aim to estimate functional size of the software according to COSMIC by using SBDs of a business process. Usually process models are available as a part of analysis activities in organizations before software development projects start. In this study our first assumption is that the project team has Subject Behavior Diagrams of all subjects related with the software that they want to develop.

In COSMIC, software size may vary according to the definition of the boundaries of the software to be measured. While measuring software size by using S-BPM we should start with defining the boundaries of the software. In our method, we defined the boundary of the software as SBD of the software subject and messages interacting with this software. We make two main assumptions:

- All subjects interacting with software have available SBD’s
- All users of the software are represented as subjects in S-BPM diagrams
In S-BPM the perspectives of all subjects in the process should be considered and Subject Behavior Diagrams for each subjects participating in the process should be drawn. The definition of subjects in S-BPM does not include the computer systems, however; for our exploratory study we included the software that we will measure as a subject in the process and created SBD’s for it. As explained in the last section we plan to exclude software subject from our future work and improve our model accordingly. In this study we use behavior diagrams of software subject and other subjects who send message to or receive message from it without using the Business Objects. Even though S-BPM has the capability to represent OOI’s with the help of Business Objects, for simplicity purposes we didn’t represent them. In next phases we will integrate business objects into the model, and with the help of Business Objects we can identify Objects of Interests and Data Groups.

To estimate the functional size, we need to identify the number of data movements in the software. There are four types of movements defined in COSMIC, if we can estimate the numbers of each movement realistically, we can estimate total size of the software. For Enter data movements, we should count the number of data movements from the user of the software to the software. We can find these movements by counting the number of sent message states in SBD’s of the users of the software. Similarly, eXit movements are represented by the receive message states of the user subjects from software subject.

We also need to count Write and Read movements. These movements represent data read from and written into persistent storage by the software. As the movements are initiated by the software subject we should find some representatives for them in software SBD. The problem is; in software SBD send and receive states are dealing with message relations between software users and software, not between software and persistent storage. To solve this problem we define two new function states; read and write. We propose to define every functions of the software with one of these two keywords and then count reads and writes to find the number of Read and Write data movements.

![Fig. 2. Read and Write function states for software SBD](image_url)

After finding estimated sizes of four DM types, we need to sum them up to reach the estimated functional size of the software.

We structured the model based on counting data movements from SBD’s. The model needs to be tested to observe its effectiveness in real life situations. To validate our model we conducted a case study by using a real life case study. In the next section the detail of the case study and the discussions of the results are explained.
3 Case Study

3.1 Case study Design

After we defined the structure of the model to estimate functional size by using SBD’s of S-BPM, we designed a case study to test our initial idea. The purpose of this case study was to evaluate the potential of proposed methodology for real life application. We plan to improve our method using findings from this case study and conduct follow up case studies until reaching satisfactory results.

In this study we used a real business process from the “Ministry of Development”, former Development Agency, of Turkey that involves the activities to extend the auditor pool that the agency is using to assign auditors to different projects for independent evaluation (“extending independent auditor pool”). The process is simple enough to evaluate the improvement opportunities in our model and complex enough to test the model with different possible actions and movements. The same process is also used by Kaya [20] to test e-COSMIC method which aims to derive COSMIC size estimation from eEPC notation, and the details of the process can be found in [20]. Using same case study gave us the opportunity to compare effectiveness of these two approaches in future studies. Moreover, we already have the COSMIC size measurement of the software system proposed in this process model as a part of a project conducted with Ministry of Development, which helps us to compare the estimated size from the proposed method with the functional size of the software which is measured from use cases designed specifically for the software.

The steps we took during the case study were as follows:

• Derive SBD’s of the subjects in the process
• Apply proposed measurement method to find estimated COSMIC functional size
• Compare findings with the already measured size of the final product.
• Evaluate the results

The business process, expanding auditor pool, consists of 13 functional user requirements (FUR). The names of the FUR’s are given in Table 1 with their respective measurement results. We started by modeling the FUR’s using S-BPM subject behavior diagrams, because of the space limitations only the diagrams of the first FUR is given in this paper. The first FUR, “Create IA selection criteria” includes two users; Project Management Unit (PMU) Expert and software system. There are three functional processes in this FUR, which are; create selection criteria, delete selection criteria, and update selection criteria.

The diagrams for the first functional user requirement are as follows: for Program Management Unit (PMU) Expert; “Create Independent Auditor (IA) selection criteria” is given in Fig. 3, “Delete and update selection criteria” is presented in Fig. 4. The diagrams for software system subject are as follows: “Create Independent Auditor (IA) selection criteria” is given in Fig. 5 and “Delete and update selection criteria” is given in Fig. 6.
Fig. 3. PMU Expert SBD for Create Independent Auditor (IA) selection criteria

Fig. 4. PMU Expert SBD for Delete/update IA selection criteria
Fig. 5. Software SBD for Create Independent Auditor (IA) selection criteria
After we derived SBD’s for all 13 functional user requirements, we applied our proposed estimation method to all of them. For the first functional user requirement, given in detail in this paper, we first examined the diagram of the PMU expert to find Enter and eXit data movements by the messages sent to and received from the software. There are six messages sent and three messages received (create, update, and delete combined). This makes six Enter and three eXit data movements. Then we examined the SBD of the software for Read and Write data movements. As mentioned before, we used keywords of READ and WRITE during modeling in the functional states of the software, and in SBD we observed that there are three READ and six WRITE function states, that makes three Read, six Write data movements.

After we completed the examination of SBD’s for all functional user requirements and for all subjects, we compared the resulting COSMIC estimations with the measured functional sizes of the functional user requirements. (The detailed use cases and COSMIC measurements of all functional user requirements can be found in [20])

In Table 1, the results of the comparison of total data movements for all 13 functional user requirements are listed. As seen in the table, for the first functional user requirement, we estimated the same size with the COSMIC measurement, however for all of the other FUR’s there are differences (maximum of 11 and minimum of 1 in absolute values). In 5 FUR’s we over-estimated and in 7 FUR’s we under-estimated, the differences has the average of 0.54 with a median of -1. When we look at the total difference between estimated size of the software and measured size of the software, we can see that we are close to the measurement by 7 data movements, considering that the total size is 137 COSMIC Function Points (CFP) our error is 5%.

Fig. 6. Software SBD for Delete/update IA selection criteria
Table 1. Comparison of S-BPM estimation and COSMIC measurement for 13 FUR’s

<table>
<thead>
<tr>
<th>FUR#</th>
<th>(A) S-BPM Estimation</th>
<th>(B) COSMIC Measurement</th>
<th>B-A Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1- Create IA selection criteria</td>
<td>17</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>2- Create IA announcement</td>
<td>18</td>
<td>14</td>
<td>-4</td>
</tr>
<tr>
<td>3- Create IA application documents</td>
<td>17</td>
<td>12</td>
<td>-5</td>
</tr>
<tr>
<td>4- Plan IA announcement time, media, and duration</td>
<td>15</td>
<td>9</td>
<td>-6</td>
</tr>
<tr>
<td>5- Send outputs to approval</td>
<td>7</td>
<td>5</td>
<td>-2</td>
</tr>
<tr>
<td>6- Review the outputs</td>
<td>10</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>7- Update outputs</td>
<td>8</td>
<td>19</td>
<td>11</td>
</tr>
<tr>
<td>8- Preparations for announcement tools</td>
<td>5</td>
<td>3</td>
<td>-2</td>
</tr>
<tr>
<td>9- Publish announcement on webpage</td>
<td>5</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>10- Receive applications</td>
<td>5</td>
<td>4</td>
<td>-1</td>
</tr>
<tr>
<td>11- Decide selection commission</td>
<td>6</td>
<td>5</td>
<td>-1</td>
</tr>
<tr>
<td>12- Evaluate applications</td>
<td>9</td>
<td>13</td>
<td>4</td>
</tr>
<tr>
<td>13- Approve selected applications and save into database</td>
<td>8</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>130</strong></td>
<td><strong>137</strong></td>
<td><strong>7</strong></td>
</tr>
</tbody>
</table>

Average 0.54
Median -1

After comparing the total sizes of the FUR’s, we also compared the differences between numbers data movement types. The results are listed in Table 2. As seen in the table errors are distributed across all data movement types.
Table 2. Difference between data movements types for 13 FUR’s

<table>
<thead>
<tr>
<th>FUR#</th>
<th>E</th>
<th>X</th>
<th>R</th>
<th>W</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-</td>
<td>Create IA selection criteria</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>2-</td>
<td>Create IA announcement</td>
<td>-1</td>
<td>-1</td>
<td>-2</td>
<td>0</td>
</tr>
<tr>
<td>3-</td>
<td>Create IA application documents</td>
<td>-1</td>
<td>-2</td>
<td>-2</td>
<td>0</td>
</tr>
<tr>
<td>4-</td>
<td>Plan IA announcement time, media, and duration</td>
<td>-2</td>
<td>-2</td>
<td>-2</td>
<td>0</td>
</tr>
<tr>
<td>5-</td>
<td>Send outputs to approval</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>6-</td>
<td>Review the outputs</td>
<td>-3</td>
<td>4</td>
<td>4</td>
<td>-1</td>
</tr>
<tr>
<td>7-</td>
<td>Update outputs</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>8-</td>
<td>Preparations for announcement tools</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>9-</td>
<td>Publish announcement on web page</td>
<td>-1</td>
<td>4</td>
<td>4</td>
<td>-1</td>
</tr>
<tr>
<td>10-</td>
<td>Receive applications</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11-</td>
<td>Decide selection commission</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12-</td>
<td>Evaluate applications</td>
<td>3</td>
<td>-1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>13-</td>
<td>Approve selected applications and save into database</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>-3</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>7</td>
</tr>
</tbody>
</table>

3.2 Discussions, Results, and Threats

After comparing the results of the estimation model with the COSMIC measurements, we observed that even though the total difference in the size is 7 CFP, for individual FUR’s the estimations are not as accurate as we anticipated. The differences vary from 1 CFP to 11 CFP. To find out the reason of this variance we examined in detail the FUR’s with largest estimation differences.

Largest difference is in FUR 7, which is “Update outputs”. When we look at this FUR we realized that the reason of the difference is the concept of Data Group. In FUR 7 there is a message from the user to the software which requests retrieval of application documents, hire plan, announcement text, selection criteria, and approval status. For this message in the SBD of the software we draw one read function state to read all of these data and counted as one read data movement. Whereas in COSMIC we need to count 5 different reads for 5 different data groups. Particularly for FUR 7 this situation appears 4 times, resulting 14 under-estimated data movements.

Another problem is related with the reuse of functional processes. In the SBD’s we draw every functional processes in every FUR, independent from each other. If there a functional process, like listing all applicants, is included in several FUR’s, we should represent this functional process in all of SBD’s related with the FUR’s. Than in each SBD, we will count same data movements related with this functional process repeatedly. However, in COSMIC we should count the movements in a functional process only once regardless of its repetition in different functional user requirements. Because of this conflict we have over-estimated data movements in some FUR’s. For example there are 3 over-estimated data movements in FUR7 in the case study.

Our observations about the results of the comparison gave us two main reasons for the difference between estimated size and measured size:
• Data groups wasn’t considered in the model,
• Reused functional processes duplicated whenever used.

In the next phases of the research our goal will be to resolve these two issues to improve the model and then test it with another case study.

Even though we estimated the size of the software with 5% error rate, it is important to stress that this case study was for exploratory purposes and not mean to yield statistically significant results.

4 Conclusion and Future Work

In this study we proposed a method to make early COSMIC size estimations by using S-BPM diagrams and we test the estimation model in a case study. Results of the case study showed that the method has potential; however it needs to be improved to cover the excluded aspects.

In the case study, we estimated the size of the software with 5% error. 5% should be considered as a promising result for an exploratory study of a newly proposed method. During detailed examination of the case study results, we observed that the model has over-estimations and under-estimations for some functional processes. We identified two main reasons for these under- and over-estimations; first one is related with the lack of data group concept in the model. Data groups are the smallest data items in COSMIC that are moved by the data movements. In our method the diagrams weren’t designed to carry enough detail to let us count individual data groups. As listed as a future work, we plan to overcome this issue by using Business Objects to carry information about the Data Groups. The second point is related with the reuse of the functional processes; in COSMIC, regardless of its reuse, one functional process should be counted only once. Our method wasn’t designed according to this rule; in our diagrams, reused functional processes are represented in every diagram and as a result they are counted more than once.

The first issue results over-estimation of software size, and the second one results under-estimation. When combined they neutralized each other and give us an under-estimation with only 7 CFP in a 137 CFP functional user requirement.

To conclude, we think this method has potential to be used as an early size estimation method, we plan to improve the model to include precautions to cover these missing points and test with additional real life use cases to find statistically significant results. Also the future work related with this study can be listed as follows:

• Improving the model to use without the SBD for software system
• Carrying information about Data Groups by using Business Objects
• Implementing reuse of functional processes
• Designing a tool for automatic size estimation from S-BPM
• Conducting case studies with different contexts and level of detail
References
