A comparative study on measuring software functional size to support effort estimation in agile

Fabián Ugalde, Christian Quesada-López, Alexandra Martínez, and Marcelo Jenkins

University of Costa Rica, San Pedro, Costa Rica
{fabian.ugalde, cristian.quesadalopez, alexandra.martinez, marcelo.jenkins}@ucr.ac.cr

Abstract. Software effort estimation models based on functional size allow software organizations to plan their development projects. A large number of organizations have adopted agile processes, but there is little evidence on the adoption of functional sizing methods to support software effort estimation in agile contexts. In this study, we compare four functional size estimation methods as the basis for effort estimation in the context of a startup company that develops mobile applications using an agile methodology. Measurements of software size, expressed in User Story Points (USP), Use Case Points (UCP), IFPUG Function Points (UFP), and COSMIC Function Points (CFP), were taken for a set of requirements from one project in the company. Effort estimation models were then derived from these measurements, using regression, and their accuracy was determined by the Mean Magnitude of Relative Error (MMRE) and Mean Balanced Relative Error (MBRE). We obtained the following MMRE results for each functional sizing method: 0,86 for UCP, 0,36 for USP, 0,36 for UFP and 0,22 for CFP, and the following MBRE results: 0,98 for UCP, 0,45 for USP, 0,53 for UFP and 0,35 for CFP. The effort estimation model based on COSMIC function points turned out to be the most accurate in the context of the software organization under study. Additionally, convertibility models between sizing measurements were generated to allow the organization to convert its historical measurements into any other software size measure, without having to perform the counting process of the target method.

Keywords: Functional size, software effort estimation, function points, IFPUG FPA, COSMIC FFP, empirical study.

1 Introduction

The use of effort estimation models can help software companies plan, monitor, and control their efforts in terms of costs and development processes [8]. Therefore, having realistic estimates at an early stage of the project life cycle, allows managers to control resources effectively [10]. In agile projects, effort estimations based on functional size are being incorporated to provide more accurate estimates [18]. However, more empirical evidence is needed on the use of functional size measurements to support software effort estimation in agile contexts. According to Lavazza et al [8], the accuracy of effort estimation must be carefully evaluated before its use in a real life environment.

Hence, our study aims to provide empirical evidence on the accuracy of effort estimation based on functional size measurements in an agile context. It also provides enough details of the methods used –as adviced by Kitchenham et al. [6] – so that other researchers can replicate the study in different contexts, thus widening the body of evidence.

In this study, we investigated whether the choice of functional size measurements has an effect on the accuracy of effort estimation models in the context of an agile organization. This organizacion is a startup company that develops mobile applications aimed at promoting healthy lifestyles, through the application of the social network approach to nutritional concepts.

The purpose of our case study is to compare the accuracy of effort estimation models built upon four different functional size measurements: Use Case Points (UCP), COSMIC Function Points (CFP), IFPUG Function Points (UFP), and User Story Points (USP). The four functional size estimation methods were applied to the same set of requirements from a mobile application project within the organization. Regression methods were then used to derive effort estimation models from the four size measurements. The Mean Magnitude of Relative Error (MMRE) and Mean Balanced Relative Error (MBRE) were used to determine which model was more accurate for predicting effort. Criteria set out by [6, 8] were considered to mitigate some threats to validity. To guide this study, two research questions were defined:

RQ1: Which functional size-based effort estimation model yields the most accurate estimates in the context of the organization under study?

RQ2: Can convertibility models of functional size measurements yield accurate results in the context of the organization under study?

The rest of the paper is organized as follows. Section 2 explores the functional size estimation methods used in our study: Use Case Points, User Story Points, COSMIC Function Points and IFPUG Function Points. Section 3 presents the related work. Section 4 describes the case study performed in the startup company. Section 5 shows the results obtained from this case study. In Section 6 we discuss and summarize the results. And section 7 presents our conclusions and future work.

2 Background

Here we describe each of the functional size estimation methods used in this study: Use Case Points, User Story Points, IFPUG Function Points and COSMIC Function Points.

2.1 Use Case Points

A *use case* details a scenario of how the software system should interact with the user –or other systems– to carry out a specific action or activity [23]. Software estimation by *use case points* is a technique to calculate and predict the size of a system for software development projects [24]. To calculate the total points, it is necessary to count the total transactions in each use case.

Use cases and its actor are categorized into simple, average and complex. For each category, a weighted point is assigned. The unadjusted use case points are the sum of all weighted points assigned.

2.2 User Story Points

User stories are the way in which requirements are captured in agile software development methodology. A *user story* is independent, negotiable, valuable, estimated, small and verifiable [25]. One method to estimate the size of a user story is through *planning poker*, which uses a Fibonacci scale. In planning poker, each team member assigns user story points for a requirement, based on its expert criteria. Then the team discusses until a single size value is agreed on, and that will be the *user story points* assigned to that requirement [26].

2.3 IFPUG Function Points

Function Point Analysis (FPA) is a proven and accepted methodology to determine the size of a software development project [27]. FPA measures software size in terms of the following attributes [27]:

- Data entering a system: external inputs (EI) as logical transaction inputs.
- Data leaving the system: external outputs (EO) or external queries (EQ) such as online screens, reports, or inputs to other systems.
- Data created and stored within the system: internal logical files (ILF) such as user-defined logical groups of data.
- Data maintained within a different system but necessary to satisfy a certain process requirement: external interfaces (EIF) as interfaces to other systems.

There are set of counting rules that assign *IFPUG Function Points* (UFP) to each EI/EO/ILF/EIF, based on the number of Data Element Types (DET) and File Type Referenced (FTR) associated with the transactional function. Such rules can be found in the work of Gandomani et al. [28].

2.4 COSMIC Function Points

System functions are modeled through functional processes, which can be divided into sub-processes [29]. Sub-processes are classified in two types: data movement and data manipulation. The data movement sub-process is the main goal for the calculation of the function point, which is categorized into four categories:

- Input (I): the movement of a user's data to a functional process.
- Output (O): the movement of data from a functional process to a user.
- Read (R): the movement of data from persistent storage to a function process. Persistent storage must be part of the system.
- Write (W): the movement of data from a functional process to persistent storage. Persistent storage must be part of the system.

In a functional process, the data movement quantity for the four categories mentioned above are presented as *COSMIC function points* (CFP), in a unit called CFSU (COSMIC functional size unit).

3 Related work

We performed a literature review and found several studies on comparing software effort estimation models, but only a few studies on convertibility of size measurements. We will first present the studies related to estimation model comparison [1, 3, 5, 9, 10, 11, 13, 15, 17, 19, 20, 22], and then the studies about convertibility between functional size measurements [7, 18].

The work by Salmanoglu et al. [17] performed three case studies that compared the effectiveness of effort estimation based on two functional size methods (CFP and USP) in an agile context. They used regression analysis to generate statistical models from software size measurements and evaluated their performance using MMRE. They concluded that CFP produced the most accurate effort estimation model.

Similarly, Ungan et al. [20] compared the effectiveness of two approaches for effort estimation in an organization that uses SCRUM. They compared USP-based effort estimation (with Planning Poker) to CFP-based effort estimation. They used regression models and artificial neural networks to develop the estimation models. They showed that COSMIC measurements are a better basis for effort estimation than User Story Points, in their context.

Paz et al. [13] applied Increment COSMIC Function Points in an agile environment, and compared its accuracy against the Rational Unified Process (RUP) method, concluding that the COSMIC-based model was better. They introduced the concept of *Incremental COSMIC Function Points* as an approach for adapting the CFP measure to agile enviroments. They proposed that the sum of all Incremental COSMIC function points for each requirement or work size unit (user story, from SCRUM standpoint), will determine the total COSMIC function points for the whole system.

Desharnais et al. [3] proposed an approach to move from COCOMO (Constructive Cost Model) to User Stories using COSMIC as the size measurement method. They showed a model that allows estimating the effort for each user story based on CFP. Sholiq et al. [19] compared UCP against UFP in an effort estimation model, concluding that UCP was slightly better. Brian et al. [1] propose an effort estimation model using external databases based on UFP and CFP, reaching the conclusion that there was no significant difference between these two software size estimation methods.

Mendes et al. [10] studied three cost estimation models applied to web hypermedia applications, using regression models to support their estimations. They compared linear and stepwise regressions, and case-based reasoning (CBR), concluding that CBR technique gave the most accurate results.

Wilkie et al. [22] explored the utility of function point software sizing techniques when applied at two levels of software requirements documentation, in a commercial software development organization. They appraised the value (cost/benefit) that functional sizing techniques can bring to the planning and management of software projects,

concluding that "Estimated NESMA" is the most appropriate tool to use for size estimation in the context of the studied company.

Phannachitta [15] did a comparison of similarity measures in analogy-based software effort estimation, using a robust approach involving MMRE and some other comparison indicators. Mittas et al. [11] propose a graphical method to evaluate the performance of a cost estimation model by using an analysis of regression error characteristic. Later, Mittas et al. [12] created an algorithm to cluster and rank software cost estimation models through multiple comparisons.

Other studies [5, 9] have compared different effort estimation methods to expert-based estimates. Jørgensen and Boehm [5] debated over which effort estimation method produces better accuracy, comparing formal models against expert judgment. Elaborating on evidence that backup both approaches, they agreed there is no single *best* method, but different perspectives that benefit particular scenarios. Nonetheless, they highlight that more advanced estimation models will likely lead to significantly more accurate effort estimates. Similarly, the study by Lenarduzzi et al. [9] compared the effectiveness of an effort estimation model based on IFPUG Function Points to an expert-based estimation, in an agile environment. They showed that the accuracy of the effort estimate provided by the SCRUM team was better than the obtained through functional size measurements.

Finally, two studies [7, 18] addressed the issue of convertibility between functional size measurements. Lavazza et al. [7] presented a correlation study between IFPUG and COSMIC function points. They provide a set of models to convert counts from one software size method to the other. Santana et al. [18] examined whether function points are compatible with user story points on agile projects, and found a correlation between them. They stated that their results should respect the units of measurement and the reality of each organization.

All previously mentioned studies were performed in mid to large companies, and focused on comparing effort estimation models based on just *two* functional size measurements. Furthermore, their competing techniques consider only completed projects with a defined set of requierements and a clear scope. In contrast, our study considers effort estimation models based on *four* functional size measurements, and is performed in an agile startup company. This context poses some challenges such as unclear project scope and unstable requirements.

4 Our Case Study

We conducted a case study in a software startup company mainly dedicated to the development of mobile applications. It has four developers, who use the SCRUM methodology. The organization has been using an agile metric collection tool, hence both estimated and real effort data were available for every user requirement (use case). We obtained most of the project data from this repository. Although the requirements were developed using SCRUM, there were formal specification documents that we used to count COSMIC, IFPUG Function Points, and Use Case Points.

4.1 Goal and object selection

The goal of our case study was to accurately estimate the effort by generating models based on software functional size measurements. We scoped our work to four functional size estimation methods: Use Case Points, User Story Points, IFPUG Function Points, and COSMIC Function Points. A secondary and related goal of the study was to seek accurate convertibility models between functional size measurements. Both goals are tied to the context of the studied organization and projects, hence results of this case study are limited to this context only.

Since the organization where the study was performed used SCRUM, we selected nine epics from one project, for this case study. The epics were broken down into 32 requirements and use cases, which were in turn decomposed into 119 user stories.

4.2 Data collection procedure

We created a spreadsheet to record the functional size measurements as well as the actual effort time. To collect the functional size measurements expressed in UCP, UFP, and CFP, one of the researchers (previously trained on these size estimation methods) performed the counting corresponding to each sizing method. On the other hand, for each user story, we recorded both its actual effort, and its estimated effort obtained with planning poker.

We identified outliers by calculating the productivity index (**PI**) of each requirement under each sizing method as **PI= actual effort/size**, and then generating box-plots with this information. Outliers indicated in boxplots were then discarded.

4.3 Analysis procedure

The analysis procedure for generating the effort estimation models was based on a regression analysis. To select the regression analysis approach to follow (linear, multiple, polynomial, exponential), the dataset was analyzed with a scatterplot. Depending on the pattern revealed by the scatterplot, different regression approaches were chosen. By default, linear regression was used. In all models, the dependent variable was the estimated effort time (EFF) and the independent variables were a subset of the software functional size measurements (i.e., UCP, USP, UFP or CFP).

On the other hand, the procedure for calculating the MMRE was to apply Equation 1 to each generated model. Case study wise, the lowest MMRE would determine the best effort estimation model. Nevertheless, a model that exhibits an MMRE value less than 0.25 is considered accurate enough.

$$MMRE = \frac{100}{T} \times \sum_{i=0}^{T} \frac{|estimated\ eff\ (i) - actual\ eff\ (i)|}{actual\ effort\ (i)} \quad (1)$$

Additionally, the procedure for calculating the MBRE was as follows: first, we applied Equation 2 to obtain the Relatative Error (RY_i); then, we calculated the Balance Relative Error (R_i) by applying Equation 3; finally, we applied Equation 4 to obtain the MBRE value.

$$RY_i = \frac{estimated\ eff\ (i) - actual\ eff\ (i)}{actual\ effort\ (i)} \quad (2)$$

$$R_i = \begin{cases} \frac{estimated\ eff\ (i) - actual\ eff\ (i)}{actual\ effort\ (i)}, & estimated\ eff\ (i) - actual\ eff\ (i) \geq 0 \\ \frac{estimated\ eff\ (i) - actual\ eff\ (i)}{estimated\ eff\ (i)}, & estimated\ eff\ (i) - actual\ eff\ (i) < 0 \end{cases} \quad (3)$$

$$MBRE = \frac{1}{N} \times \sum_{i=0}^{N} R_i \quad (\mathbf{4})$$

In the context of the MBRE metric, a model is considered accurate enough when the MBRE value is less than 0.35. According to Miyazaki [30], 35% accurancy might be enough in the early phase of the software life cycle.

4.4 Threats to validity

We evaluated the validity of our study based on Runeson's guidelines [16], considering construct validity, internal validity, external validity, and reliability, as shown below. The threats to validity are briefly mentioned next.

Construct validity: in our case study, we created a set of effort estimation models based on functional size measurement for a specific organizational context, and evaluated their accuracy performance using MMRE and MBRE.

Internal validity: one validity threat is the small sample size used (having only 32 requirements to analyze). Additionally, a bias could have been introduced in the sample selection process. Also, some data were discarded from the analysis as they were identified as outliers. Another threat is that only one researcher performed the functional size measurement for each requirement, however, this researcher has proven experience with these counting methods. Finally, MMRE as a statistical measure to evaluate a model's performance has sometimes failed to show statistical evidence, as advised by Kitchenham et al. [6] and Lavazza et al. [8]. We tried to mitigate this threat by also comparing accurancy in combination with MBRE, which has demonstrated more effiency, according to Jørgensen [31]

External validity: the effort estimation models are applicable only to the organization described in this study. These models are not intended to be used in other contexts besides this one.

Reliability: since the organization's dataset grew with time, there is a validity threat that this case study presents non-repeatable results, as pointed out by Lavazza et al. [8]. As the organization scales up, results might vary.

5 Analysis of Results

In this section we present the results of our case study, addressing each of the research questions.

5.1 Accuracy of effort estimation models based on functional size (RQ1)

Our first research question aimed at finding the most accurate estimation model based on functional size measurements, in the context of the organization, and using MMRE and MBRE as the accuracy performance measurements.

Figure 1 describes the relationship between requirements, use cases, and user stories. For this case study, we gathered 9 Epic User Stories, which were broken down into 32 Requierements and 32 Use Cases. During the development phase, those Requirements and Use Cases were structured as 119 User Stories. These User Stories were delivered by the team, having the actual time tracked against each User Story. Requierements assets were used for UFP and CFP functional size measurements, Use Cases were used for UCP and User Story items for USP functional size measurements.

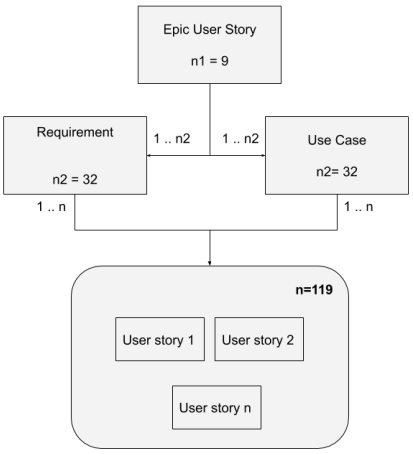

Fig. 1. Requirements detail structure.

Using regression, we developed a series of parametric statistical models to describe the relationship between size and effort. We applied different kinds of regression analyses to build the models. The model types were selected by using a visual pattern scan in the scatterplot for each functional measurement. Simple, multiple, polynomial, and exponential regression methods were applied to UCP, UFP, USP, CFP counts to render several effort estimation models. We generated scatterplots for each functional size method to decide on the type of regression to apply, based on the dataset pattern. Figure 2 shows scatterplots of actual effort vs. size for each sizing measurement: (1) UCP,

(2) UFP, (3) USP, and (4) CFP. The Y-axis represents actual effort, while the X-axis represents the functional size measurements, for each method.

In total, we developed 11 effort estimation models based on four different functional size measures. Table 1 summarizes these models, their input parameters, dependent parameters, analysis method, MMRE and MBRE

The accuracy of each effort estimation model was assessed by its Mean Magnitude of Relative Error (MMRE) and Mean Balanced Relative Error (MBRE), depicted in Equations 1 and 4. The accuracy of an estimation model increases as its MMRE gets closer to 0, thus a small MMRE value indicates that the model is a good predictor. Following [4][30], a model with an MMRE value less than 0.25 and an MBRE value less than 0.35 was deemed accurate enough to be used in the organization under study.

Model 11 from Table 1, which uses a exponential regression on COSMIC Function Points, is the most accurate effort estimation model, with an MMRE value of 0.22 and an MBRE value of 0.35. Revisiting chart (4) from Figure 2, we confirm that this dataset follows an exponential pattern, which explains why model 11 provided the lowest MMRE and MBRE of all models.

On the other hand, model 1 from Table 1 is the least accurate estimation model, and it uses a linear regression on UCP. Revisiting chart (1) from Figure 2, there is no clear pattern in this dataset, which might explain why model 1 showed a high MMRE and MBRE.

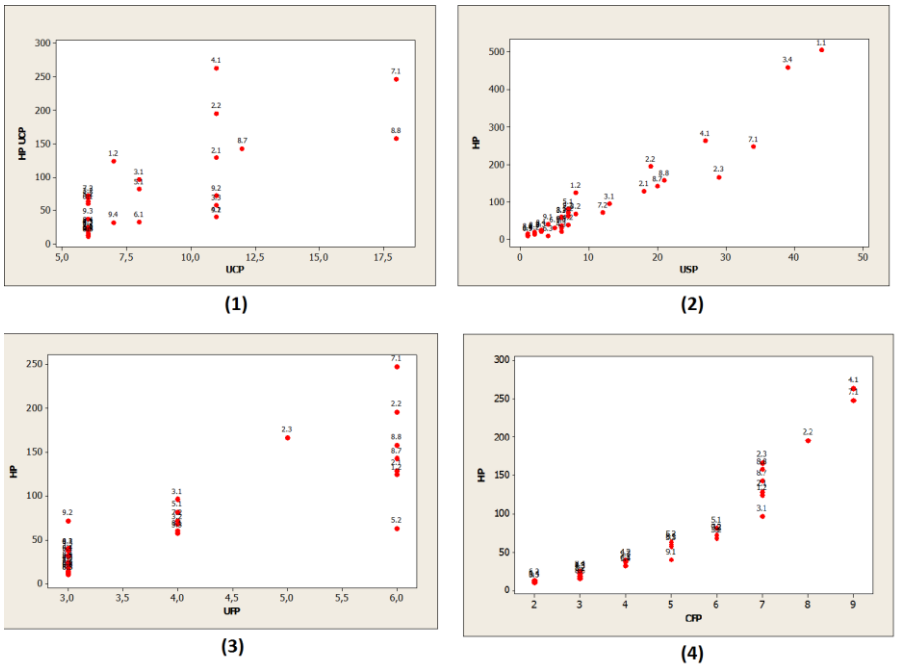

Fig. 2. Scatterplots of actual effort vs. size for (1) UCP, (2) UFP, (3) USP and (4) CFP.

Table 1. Effort estimation models based on functional size measurements.

#	Analysis Method	Parameters	Dependent Variable	Model	MMRE	**MBRE**
1	Linear	UCP	Effort	EFF = -51,0 + 14,7 UCP	0.86	0.98
2	Linear	USP	Effort	EFF = -13,1 + 9,86 USP	0.45	0.87
3	Polynomial	USP	Effort	EFF = 18,47 + 3,392 USP + 0,1650 USP^2	0.41	0.45
4	Polynomial	USP	Effort	EFF = -8,75 + 12,30 USP - 0,3977 USP^2 + 0,008921 USP^3	0.36	0.58
5	Linear	UFP	Effort	EFF = -100 + 42,9 UFP	0.48	0.53
6	Linear	DET, FTR	Effort	EFF = -46,2 + 4,51 DET + 44,0 FTR	0.36	0.54
7	Linear	DET, FTR	Effort	EFF = 1,7 + 1,47 DET + 13,9 FTR	0.31	0.45
8	Linear	DET, FTR	Effort	EFF = -86,6 + 5,93 DET + 53,3 FTR	0.23	0.36
9	Linear	CFP	Effort	EFF = -78,6 + 31,0 CFP	0.58	0.51
10	Linear	E,R,WX	Effort	EFF = -79,4 + 45,1 E + 31,4 R + 30,8 W + 15,1 X	0.47	0.79
11	Exponential	CFP	Effort	EFF = 0,789382 × $CFP^{2.63169}$	0.22	0.35

5.2 Accuracy of convertibility models for functional size measurements (RQ2)

Our second research question sought to find accurate convertibility models of functional size measurements, in the context of the organization, using MMRE and MBRE as the accuracy performance measurements.

Further analysis was done to provide a set of models that allow convertibility between different functional size measurements (e.g. from USP to CFP), without having to execute the formal counting protocol for each size estimation method. As previously stated, these convertibility models are applicable only in the context of the organization under study, and are therefore not intended to be widely used across industry.

Twelve parametric statistical models were built, one for each possible convertibility combination between funtional size measurements. We used linear regression analysis, and then we measured the accuracy by calculating the MMRE and MBRE for each model. These convertibility models are shown in Table 2, together with their analysis method, dependent variable, parameters (independent variables), MMRE and MBRE. Only models with MMRE less than 0.25 and MBRE less than 0.35 are accurate enough to be considered for common use in our company context.

Two interesting findings arise from Table 2:

1. Given the MMRE and MBRE results for models 4, 5 and 6, there is no linear model that offers an acceptable accuracy for USP convertibility from any other software size measure. Therefore, for this organization to get software size estimates in USP, they would need to perform SCRUM ceremonies (like Planning Poker) to provide them.
2. Using model 12 from Table 2, the organization would be able to convert their current USP estimates into CFP, and then use model 11 from Table 1 to estimate the effort. This approach could improve accuracy, rather than using the model 4 from Table 1.

However, this approach would lead the company to have less accuracy performance, given the probability addition rule. The MMRE must then be added to get a total MMRE of 0.45 for the whole effort estimation process. Similarly, it would lead to a total MBRE of 0.60 for the complete effort estimation process

Table 2. Convertibility models of software functional size measurements.

#	Analysis Method	Dependent Variable	Parameter	Model	MMRE	**MBRE**
1	Linear	UCP	USP	UCP = 5,57 + 0,324 USP	0.18	0.20
2	Linear	UCP	CFP	UCP = 3,09 + 1,09 CFP	0.21	0.24
3	Linear	UCP	UFP	UCP = 2,24 + 1,57 UFP	0.24	0.28
4	Linear	USP	CFP	USP = - 10,3 + 4,05 CFP	4.73	4.73
5	Linear	USP	UFP	USP = - 16,8 + 6,88 UFP	8.37	8.37
6	Linear	USP	UCP	USP = - 8,56 + 2,25 UCP	4.72	4.72
7	Linear	UFP	UCP	UFP = 2,37 + 0,195 UCP	0.21	0.24
8	Linear	UFP	USP	UFP = 2,96 + 0,113 USP	1.57	4.01
9	Linear	UFP	CFP	UFP = 1,52 + 0,506 CFP	0.14	0.15
10	Linear	CFP	UFP	CFP = - 0,435 + 1,36 UFP	0.38	0.38
11	Linear	CFP	UCP	CFP = 1,72 + 0,392 UCP	0.35	0.39
12	Linear	CFP	USP	CFP = 3,05 + 0,203 USP	0.23	0.25

6 Discussion

Regarding RQ1, we found that model 11 from Table 1 (based on COSMIC Function Points) yields the most accurate effort estimates. Comparing the accuracy of COSMIC models, we observe that the CFP-based estimation model (model 11 in Table 1) performs slightly better than the UFP-based model (model 8 in Table 1). However, model 8 is applicable only to requirements with six or more data elements (DETs). Hence, only the COSMIC-based model proved to be accurate enough and applicable to any requirement, regardless of its size or number of data elements.

These results support the findings of Salmanoglu et al. [17], who concluded that an effort estimation model based on COSMIC Function Points was more accurate than another model based on User Story Points. They also used MMRE as the performance indicator for the models, making our results quite relatable to theirs.

Our results also relate to existing evidence by Ungan et al. [20], who also found that the effort model based on COSMIC Function Points performed better than the model based on User Story Points. Besides regression analysis, they included an artificial neural networks approach to build their models, yet they obtained similar results to ours.

It is worth mentioning that our study relates to the work of Paz et al. [13] in two aspects. First, we obtained similar results, as we determined that the most accurate effort estimation model was based on COSMIC function points. Second, we followed a

similar approach to estimate the software size: we both used *Incremental COSMIC Function Points*, mostly because the application was developed using a SCRUM methodology, and this approach allows to apply CFP in agile environments.

Also for RQ1, we found that model 1 from Table 1 (based on Use Case Points) yields the least accurate effort estimates. This finding differs from the results obtained by Sholiq et al. [19], as they found a slightly better performance using Use Case Points. There are, however, two main differences among the studies that could have affected the results. First, we used a regression analysis to build the effort estimation models, while they used instead a productivy factor (calculated by them) for that specific scenario. Second, they measured the size of already completed projects, whereas we measured the size of an ongoing agile project (incremental pieces of software).

Regarding RQ2, we were able to obtain accurate convertibility models for some functional size measurements, but not for every pair. The convertibility models that can be safely used in the organization context are models 1, 2, 3, 7, 9 or 12 from Table 2, given that their estimation error measured as MMRE is less than 0.25 and MBRE is less than 0.35. These results relate to the findings of Lavazza et al. [7], as both presented a model that allow convertibility from CFP to UFP (in our study, it is model 9 from Table 2). However, in the case of the convertibility from UFP to CFP, Lavazza et al. [7] showed evidence that allows it, while we were not able to find an accurate model for it (model 10 from Table 2 was not accurate enough).

7 Conclusions

In this paper, we described a case study that compares the accuracy of effort estimation models derived through regression from four functional size measurements: Story Points, Use Case Points (UCP), IFPUG Function Points (UFP), and COSMIC Function Points (CFP). Additionally, we derived a set of convertibility models between functional size measurements, which can be used to easily convert the organization's historical measurements into different software size measures.

Our results confirm findings from previous studies such as [20], [13], and [17], which concluded that effort estimation models based on CFP show better accuracy than alternative methods compared in their studies.

However, compared to counting USP, none of the convertibility models are accurate enough to estimate software size. Thus, in the organization under study, current agile ceremonies must continue to be performed in order to estimate work size.

Our convertibility models reaffirm previous evidence by Lavazza et al. [7], as they presented a scenario where UFP and CFP were correlated, and a model is proposed to convert from one software sizing method to the other. Although expert judgment is still generally preferred as the estimation method in agile environments [21], our recommendation to the organization under study is to adopt CFP as their sizing method, especially as the number of projects grows and the company scales up. There is ample evidence in the literature that using CFP could improve an organization's estimates [3], [20], [13], and [17].

As future work, we plan to replicate this study using a larger dataset with an extended sample of functional requirements. Another possible extension to our work would be to perform a statistical analysis to compare the effort estimation models, including the statistical tests suggested by Lavazza et al. [8] such as paired *t*-tests of the MRE, or paired tests of the absolute residuals.

References

1. Briand, L., & Maxwell , K. An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques. *ICSE '99*, 1999, Pages: 313-323.
2. Chi-Jui , L., & Yeh, D.-M. A Software Maintenance Project Size Estimation Tool Based On Cosmic Full Function Point. *2016 International Computer Symposium (ICS)*, 2016, Pages: 555-560.
3. Desharnais, J., Buglione, L. & Kocaturk,B., Using the COSMIC Method to Estimate Agile User Stories. Proceedings of the *12th International Conference on Product Focused Software Development and Process Improvement*, 2011.
4. Foss , T., Stensrud, E., Kitchenham, B., & Myrtveit, I. A Simulation Study of the Model Evaluation Criterion MMRE. *IEEE Transactions on Software Engineering*, Volume: 29, Issue: 11, 2003, Pages: 985-995.
5. Jørgensen, M., Boehm, B., & Rifkin, S. (2009). Software development effort estimation: Formal models or expert judgment? *IEEE software*, 26(2), 14-19.
6. Barbara Kitchenham and Emilia Mendes. 2009. Why comparative effort prediction studies may be invalid. In *Proceedings of the 5th International Conference on Predictor Models in Software Engineering (PROMISE '09)*. ACM, New York, NY, USA, , Article 4 , 5 pages. DOI=http://dx.doi.org/10.1145/1540438.1540444
7. Lavazza, L., & Liu, G.. A Study of the Correlation between Functional Size Measures and Object-oriented Measures from UML Requirements Models. *IWSM Mensura 2018*, Pages:54-69
8. Lavazza, L., & Morasca, S. On the Evaluation of Effort Estimation Models. EASE'17. 2017.
9. Lenarduzzi, V, Lunesu, I., Matta, M & Taibi, D., Functional Size Measures and Effort Estimation in Agile Development: A Replicated Study. In Suomalainen, T. *Continuous Strategy Process in the context of Agile and Lean Software Development*. Springer, 2015.
10. Mendes, E., Watson, I., Triggs, C. et al. A Comparative Study of Cost Estimation Models for Web Hypermedia Applications. Empirical Software Engineering (2003) 8: 163. https://doi.org/10.1023/A:1023062629183
11. Mittas, N., & Angelis, L. Ranking and Clustering Software Cost Estimation Models through a Multiple Comparisons Algorithm. *IEEE Transactions on Software Engineering* Vol. 39, No. 4, 2013, Pages: 537- 551.
12. Nikolaos Mittas and Lefteris Angelis. 2010. Visual comparison of software cost estimation models by regression error characteristic analysis. J. Syst. Softw. 83, 4 (April 2010), 621-637. DOI=http://dx.doi.org/10.1016/j.jss.2009.10.044
13. Paz, F., Zapata, C., & Pow-Sang, J. A. An Approach for Effort Estimation in Incremental Software Development using COSMIC Function Points. *ESEM'14*, 2014.
14. Kai Petersen, Sairam Vakkalanka, Ludwik Kuzniarz, Guidelines for conducting systematic mapping studies in software engineering: An update, Information and Software Technology, Volume 64, 2015, Pages 1-18,

15. Phannachitta, P. Robust Comparison of Similarity Measures in Analogy-Based Software Effort Estimation. *11th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, 2017.
16. Runeson , P., & Höst, M. Guidelines for conducting and reporting case study research in software engineering. *Empiric Software Eng.* 2008. Pages: 131–164.
17. Salmanoglu, M., Hacaloglu, T., & Demirors, O. Effort Estimation for Agile Software Development: Comparative Case Studies Using COSMIC Functional Size Measurement and Story Points, *IWSM/Mensura '17*, 2017.
18. Santana C., Leoneo F., Vasconcelos A., Gusmão C. (2011) Using Function Points in Agile Projects. In: Sillitti A., Hazzan O., Bache E., Albaladejo X. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2011. Lecture Notes in Business Information Processing, vol 77. Springer, Berlin, Heidelberg
19. Sholiq, Renny, S., & Pribadi , A. A Comparative Study of Software Development Size Estimation Method: UCPabc vs Function Points. *4th Information Systems International Conference 2017*, Pages 470-477.
20. Ungan, E., Çizmeli, N., & Demirörs, O. Comparison of Functional Size Based Estimation and Story Points, Based on Effort Estimation Effectiveness in SCRUM Projects. *Euromicro Conference on Software Engineering and Advanced Applications*, 2014, Pages. 77-80.
21. Usman, M., Mendes, E., & Börstler, J. Effort Estimation in Agile Software Development: A Survey on the State of the Practice. *EASE'15*, 2015.
22. F.G. Wilkie, I.R. McChesney, P. Morrow, C. Tuxworth, N.G. Lester, The value of software sizing, *Information and Software Technology*, Volume 53, Issue 11, 2011, Pages 1236-1249
23. Bente , A., Hege , D., Sjøberg, D., & Magne, J. Estimating Software Development Effort Based on Use Cases - Experiences from Industry. The Unified Modeling Language. Modeling Languages, Concepts, and Tools, 2001, Pages: 487-502.
24. Karner, G. Metrics for Objectory. Diploma thesis. University of Linköping, No. LiTH-IDA-Ex-9344:21, 1993
25. Popli , R., & Chauahn , N. Managing Uncertainty of Story-Points in Agile Software. 2nd International Conference on Computing for Sustainable Global Development (INDIACom), 2015, Pages: 1357-1361.
26. Gandomani, T. J., & Mahsa Radnejad, H. F. Planning Poker in cost estimation in Agile methods: Averaging Vs. Consensus. 2019 IEEE 5th International Conference on Knowledge-Based Engineering and Innovation (KBEI), 2019, Pages: 66-71.
27. Garmus, D., & Herron, D. Function Poin Analysis: Measurement Practices for Successful Software Projects. Addison-Wesley, 2001
28. Gandomani, T. J., & Mahsa Radnejad, H. F. Planning Poker in cost estimation in Agile methods: Averaging Vs. Consensus. 2019 IEEE 5th International Conference on Knowledge-Based Engineering and Innovation (KBEI), 2019, Pages: 66-71.
29. Chi-Jui , L., & Yeh, D.-M. A Software Maintenance Project Size Estimation Tool Based On Cosmic Full Function Point. 2016 International Computer Symposium (ICS), 2016, Pages: 555-560.
30. Miyazaki, Y., et al., Robust regression for developing software estimation models. Journal of Systems and Software, 1994. 27(1): Pages. 3-16.
31. M. Jørgensen. A critique of how we measure and interpret the accuracy of software development effort estimation. Proceedings of 1st International Workshop on Software Productivity Analysis and Cost Estimation, 2007, Pages 15–22.