Comparison of Functional Size Based Estimation and Story Points, Based on Effort Estimation Effectiveness in SCRUM Projects

Erdir Ungan
Bilgi Grubu Software Research,
Training, Consultancy Ltd.
Ankara, Turkey
erdir@bg.com.tr

Numan Çizmeli, Onur Demirörs
Graduate School of Informatics
Middle East Technical University
Ankara, Turkey
numancizmeli@gmail.com, demirors@metu.edu.tr

***Abstract*— In this study, we compared the effectiveness of two approaches to effort estimation for organizations utilizing SCRUM. We compared SCRUM's native effort estimation method Story Points and poker planning, with effort estimation models based on COSMIC Function Points (CFP) for a selection of projects. We utilized different regression models and ANN methodology to develop estimation model from the backlog stories. Results indicated that, estimation model built with COSMIC measurement results prove to be a better method for effort estimation than SCRUM's story points.**

Keywords—SCRUM; COSMIC; effort estimation; Story Points

I. INTRODUCTION

Software effort estimation is one of the base activities of software project planning. Estimation results are used in various planning activities such as scheduling, budget planning, cost assessment, risk analysis, resource planning and leveling. Accurate estimations are essential for successful management. Studies [10][11] have shown that most of the unsuccessful software projects fail because of inaccurate estimations.

In last few decades, numerous effort estimation approaches were devised for software projects. These effort estimation methods can be broadly classified as [12]; Expert Judgment, Formal Estimation Methods, Composite Estimation Methods.

Various statistical techniques were used in formal estimation model such as: Single and multi-valued regression analyses, case-based reasoning, classification and regression trees, neural networks, Bayesian statistics, genetic programming, linear programming, fuzzy logic modeling.

These methods utilize size of the software to be developed size as either the sole or main input for calculating effort. Earlier methods mostly relied on software lines of code (SLOC) as the size input. However, functional size measurement methods such as Function Point Analysis (FPA), IFPUG and COSMIC has become more popular as input in later models years. Functional size measurement methods had certain advantages over SLOC measurements such as being able to be applied early on the software project lifecycle and being independent of the development technology and programming language.

In this study we used COSMIC functional size measurement method to measure the size of software. COSMIC is the one of the most recent and popular functional size measurement methods both in the academia and industry. Software is measured in a standardized level of functionality called Functional Process. Each Functional Process is further broken down to Data Movements and Data Manipulations. In COSMIC Data Movements are defined as Entry, Exit, Read and Write data movements. These constitute the Base Functional Components of the measurement method. The method is also accepted as an ISO/IEC International Standard - ISO/IEC 19761: Software Engineering – COSMIC-FFP – A functional size measurement method. Further information about the COSMIC method can be found in [2].

On the other hand, agile methodologies became increasingly popular in software development and most popular methodologies of agile movement, incorporate or suggest effort estimation activities which are essentially based on expert judgment.

SCRUM is one of the most common agile methodologies. It defines an iterative lifecycle and allows prioritizing large task items into manageable smaller work items. SCRUM promotes adaptive planning, evolutionary development and delivery, an iterative approach, and encourages very rapid response to change. SCRUM utilizes Story Point estimations, for effort estimation, which is essentially a collaborative effort estimation method for expert judgment. Further information about the SCRUM methodology and its activities can be found in [13] and won't be described within the scope of this paper.

In this study, we compare effort estimation accuracy of SCRUM's estimation method - Story Points and possible formal effort estimation models developed using COSMIC Function Points (CFP) through a case study.

The study was conducted in a large software company in Turkey which practice SCRUM methodology. We measured COSMIC size of the projects, recorded their Story Point estimations and collected actual effort data from the company's metric databases. We developed a number of mathematical effort estimation models based on COSMIC sizes and actual data. Then we compared the accuracies of these models with the story points estimations performed by domain experts and actual developers of the projects.

978-1-4799-5795-8/14 $31.00 © 2014 IEEE
DOI 10.1109/SEAA.2014.83

II. Previous Studies

There are lots of studies about software effort estimation models in literature. Some of the well-known studies in literature are briefly summarized below.

Maxwell, Wassenhove and Dutta have a research on effort estimation utilizing data from 108 projects conducted in European Space Agency (ESA) [3]. They developed an effort estimation model for ESA. Then, they studied whether a model based on a multi-company database can be successfully used in estimating effort in a specific company. They developed those models, including only productivity factors which are found to be significant for individual projects.

They concluded that, the most accurate generic effort estimation model was based on the size of the software. Main factors found to affect the productivity values in projects within the ESA dataset were: application category, programming language, required software reliability, main storage constraint, and the use of modern programming practices or software tools.

Oliveira used support vector regression, radial basis functional neural Networks and linear regression for estimation project effort [6]. He used datasets of NASA for estimation calculations. Results showed that support vector regression significantly outperforms radial basis function networks which is a single-layer type of ANN.

In his paper in 2004 [7], Jorgensen studied methodologies with top down and bottom up expert estimation of software development effort. He investigated when to apply either approach. In top down approach, total effort of the project may be based on project properties as whole and must be distributed over project activities. In bottom up approach, all project activities at the bottom will be estimated and will be summed to see the whole estimation. In his study, Jorgensen observed that, having experience and historical data from similar previous projects is a precondition to have an accurate top down estimate. He suggests that, companies should apply bottom up estimation strategy for a new project unless they have experience with similar projects before.

In her study, Tunalılar [8], proposes a methodology for organizations to manage and execute effort estimation processes (EFES). She evaluated some concepts such as Functional Similarity and usage of individual Base Functional Components (BFC) in effort estimation models. She reported that incorporating functional similarities and BFCs to the estimation model, resulted in consisted productivity value for teams which would otherwise fluctuate.

Studies mentioned in this section mainly focus on building an effort estimation model for traditional software development lifecycles having requirement specifications and defined project boundaries. In this study, we investigated whether it is possible to build an effort estimation model using COSMIC size measurement for agile/scrum projects to replace poker planning and story points.

III. Case Study

We designed and conducted a case study in a large software company which develops software solutions for the biggest governmental bank in Turkey. The organization has 500 developers and practices SCRUM methodology in their operations throughout the company. The organization has been developing software with SCRUM methodologies since 2011, that is, for 3 years at the time of the study.

We required the organization to possess accurate and accessible historical data for their projects. They have been using a metric collection tool specifically designed for agile metrics. We collected most of the project data from this repository.

As all projects were developed using SCRUM, there existed no formal requirement specification documents to measure COSMIC function points from. There were backlog stories that describe the behavior of the piece of software to be developed. Therefore, COSMIC measurements were conducted on these user stories and any other supporting documents. We performed the measurement with a measurement tool CUBIT which we had developed in our research group.

In order to minimize measurement errors, measurement results were verified before constructing any mathematical model. CUBIT was used for the first step of verification as it incorporates an experimental automated measurement verification tool developed within our research group. These results were also reviewed and verified by an independent COSMIC expert. After verification, data was analyzed based on other project data like effort/size rate, tools used, experience of team members, platforms used, team's experience on the technology etc. to identify any possible outliers. A sample of 10 software development projects were selected for the case study.

After these steps, it was assumed that it was possible to have a clean data set. Project data including measurement results are listed in TABLE I.

IV. Data and Analysis

After data collection we developed parametric mathematical models which describe the relationship between size and effort. We applied different kinds of regression analyses and ANN to build the models. Simple, multiple, polynomial, power, exponential and logarithmic regression formulas were applied to CFP results, SP values and effort data. A 9-Neuron artificial neural network was defined and CFP values and effort data were used to train this network. Resulting models' effort estimation accuracies were assessed by analyzing MMRE and PRED results. Then, we compared these models with models relating story points and effort.

For simple and multiple linear regression methods, effort values for different lifecycle phases (Analysis, Coding and Testing) are considered besides total project effort. For multiple regression, base Functional Components of COSMIC method – Entry, Read, Write and eXit data movements - were used as independent variables. However, it was not possible to use multiple regression for Story Point values as it cannot be broken down to multiple independent variables.

TABLE I. PROJECT DATA

#	Project	Application Type	E	X	R	W	Total CFP	Effort	Prod. Rate	Deviance from Avg.
1	HCZ	EMC Documentum and C# Windows Application Int.	45	60	47	25	177	364	0,4863	-0,1903
2	MEV	C# Windows Application	47	81	57	31	216	519	0,4162	-0,2604
3	IP	C# ASP.NET Web Site	34	34	18	16	102	117	0,8718	0,1952
4	IH	C# Windows Application	24	34	25	9	92	196	0,4694	-0,2072
5	IAM	C# Windows Application	39	47	43	20	149	332	0,4488	-0,2278
6	KK	C# Windows Application	40	44	29	26	139	286	0,4860	-0,1906
7	WP	C# ASP.NET Web Site	49	53	38	25	165	188	0,8777	0,2011
8	MT	C# Windows Application	47	68	52	22	189	373	0,5067	-0,1699
9	NF	C# Windows Application	31	30	17	17	95	189	0,5026	-0,1739
10	BN	C# ASP.NET Web Site	56	47	32	34	169	542	0,3118	-0,3648

In total we developed 13 effort estimation models for COSMIC size input and 8 effort estimation models for Story Point input. Input parameters, dependent parameters and analysis method used is summarized in TABLE II.

After developing models, we assessed and compared their estimation accuracies. Magnitude of Relative error (MRE), Mean Magnitude of Relative Error (MMRE) and PRED (N) values, two commonly used criteria used in software estimation models, are used to evaluate the accuracy of estimations. Calculation of these values are given below.

$$MRE = \frac{|estimated\ effort - actual\ effort|}{actual\ effort} \quad (1)$$

$$MMRE = \frac{100}{T} x \sum_{i=0}^{T} \frac{|estimated\ eff.(i) - actual\ eff.(i)|}{actual\ effort\ (i)} \quad (2)$$

$$PRED\ (N) = \frac{100}{T}\ x\ \begin{cases} 1, & MRE(i) < \frac{N}{100} \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

Accuracy of an estimation model gets higher as its MMRE gets closer to 0 and PRED (N) value gets closer to 100. A small MMRE indicates that the model is a good predictor and it is considered to be an acceptable estimation model if its MMRE is equal to 0.25 or less. PRED is used as an indicator of how many of the predicted values fall within a given range of their actual values. In the study, the estimation model is considered to be acceptable if PRED (30) >= %70. In other words, the model is said to be acceptable if 70% of the predicted values are within 30% of their actual values.

MMRE and PRED (30) values for each analysis method for both COSMIC size and Story Points are presented in TABLE II.

V. RESULTS

Among the models developed using COSMIC function points, best MMRE and PRED (30) values are achieved with multiple regression method having total project effort as the dependent value. Multiple inputs of this method were based functional components of COSMIC measurement method which are E, R, X, W data movements. With MMRE value of 6.29 and PRED value of 100, this method proves to be highly accurate and applicable.

Among the models developed using Story Points, most accurate results are obtained with power regression and polynomial regression methods. Power regression method has an MMRE value of 18.01 and PRED (30) value of 77.89 whereas polynomial regression method has an MMRE value of 18.93 and PRED (30) value of 88.89.

When we compare the accuracy indicators of COSMIC and SP based models, we observe that SP based estimation models perform slightly better than COSMIC based models in common analysis methods. However, significantly better results are achieved by multiple input analysis methods i.e. multiple regression and ANN. Only COSMIC results were eligible to be used in such methods.

VI. THREATS TO VALIDITY

Certain factors may affect generalizability of the results of the study.

Organization does not use standard templates for user stories recorded in the backlog. In addition to that, there exist no review or similar verification activities for backlog stories. Possible discrepancies in user story definitions and/or inconsistent abstraction levels for user stories may have introduced a limited level of error to measurements.

Metric collection tool used in the organization has no separated task classifications such as analysis, testing or coding. We retrospectively made this separation for the purposes of this study. This may have caused small errors in the effort values assigned to lifecycle phases. However, best effort estimation models turned out to be those having total project effort as the dependent variable, which minimize the effect of such errors on the study.

TABLE II. APPLIED ANALYSIS METHODS AND MMRE AND PRED FOR CFP BASED MODELS

#	Analysis Method	Parameters	Dependent Variable	MMRE (CFP)	PRED (30) (CFP)	MMRE (SP)	PRED (30) (SP)
1	Simple Regression	Total Size	Total Effort	22.48	77.78	20.15	77.78
2	Simple Regression	Total Size	Analyze Effort	70.95	44.44	61.76	66.67
3	Simple Regression	Total Size	Coding Effort	81.28	44.44	46.28	66.67
4	Simple Regression	Total Size	Testing Effort	27.24	66.67	48.34	44.44
5	Multiple Regression	#x, #e, #r, #w	Total Effort	6.29	100.00	N/A	N/A
6	Multiple Regression	#x, #e, #r, #w	Analyze Effort	28.22	66.67	N/A	N/A
7	Multiple Regression	#x, #e, #r, #w	Coding Effort	19.12	77.78	N/A	N/A
8	Multiple Regression	#x, #e, #r, #w	Testing Effort	9.25	100.00	N/A	N/A
9	Curve Fit. Logarithmic	Total Size	Total Effort	23.14	77.78	20.68	88.89
10	Curve Fit. Exponential	Total Size	Total Effort	21.52	77.78	20.94	66.67
11	Curve Fit. Power	Total Size	Total Effort	22.57	77.78	18.01	77.78
12	Curve Fit. Polynomial	Total Size	Total Effort	20.60	77.78	18.93	88.89
13	ANN	#x, #e, #r, #w	Total Effort	7.88	100.00	N/A	N/A

VII. CONCLUSION

In this study, we compared effort estimation accuracy of SCRUM's estimation method Story Points and several effort estimation models developed using COSMIC Function Points (CFP) through a case study.

We observed that models that utilize a single total size value as input fall short in accuracy. Such models developed with CFP and SP inputs resulted in very close MMRE and PRED values. These results did not explicitly favor COSMIC or SP based models. Single input estimation models based on COSMIC or SP sizes had either low or marginally acceptable accuracy indicators. However, we observed that estimation models incorporating multiple aspects of software size had much higher accuracy in estimations. Nonetheless, Story Point estimations did not incorporate any multiple aspects or dimension in measurement and therefore cannot be used as an input for multi input analyses. COSMIC measurements on the other hand, consist of 4 different BFCs by definition. Results indicate that these BFCs collectively characterize the size of software much better than Story Points for the purposes of effort estimation rendering COSMIC based estimation models to be more accurate than SP based estimation models.

Story Point estimation is a form of collective expert judgment for effort estimation. It takes relatively less time and effort to estimate a project through Story Points compared to the effort required to measure User Stories using COSMIC method. However, an estimation method that incorporates a multiple input statistical model, constructed with organization wide historical project data would be more accurate and using distinct data movement types of COSMIC can be used for such a model whereas Story Points cannot.

REFERENCES

[1] ISO/IEC, IS 14143-1: 2007 - - Information Technology - Software Measurement - Functional Size Measurement - Part 1 definition of concepts, International Organization on Standardization, February 2007

[2] COSMIC, Measurement Manual, Version 3.0.1, Common Software Measurement International, 2009.

[3] Katrina Maxwell, Luk Van Wassenhove, Soumitra Dutta, Performance Evaluation of General and Company Specific Models in Software Development Effort Estimation. 1999.

[4] Magne Jørgensen, Kjetil Moløkken-Østvold, Reasons for Software Effort Estimation Error: Impact of Respondent Role, Information Collection Approach, and Data Analysis Method, 2004

[5] Magne Jørgensen, Ulf Indahl, Dag Sjøberg, Software effort estimation by analogy and ''regression toward the mean'', December 2002

[6] Adriano L.I. Oliveira, Estimation of software project effort with support vector regression, December 2005

[7] Magne Jørgensen, Top-down and bottom-up expert estimation of software development effort, April 2003

[8] Seçkin Tunalılar, EFES : An effort estimation methodology, September 2011.

[9] Comparison Of Functional Size And Story Points For Effort Prediction Effectiveness On Scrum Projects

[10] Boehm, B.W., Horowitz, E., Madachy, R., Reifer, D., Bradford K.C., Steece, B., Brown, A.W., Chulani, S., Abts, C.: Software Cost Estimation with COCOMO II, Prentice Hall, New Jersey, (2000). ISBN-10: 0-13-026692-2

[11] Steve McConnell. Rapid development: taming wild software schedules. Microsoft Press, 1996. ISBN: 1-55615-900-5

[12] K. Kavoussanakis, Terry Sloan,UKHEC Report on Software Estimation,The University of Edinburgh December 2001

[13] Jeff Sutherland; Ken Schwaber (2013). "The Scrum Guide". www.Scrum.org. Retrieved July 2013