ANALYSIS OF SIZE ESTIMATION TO PREDICT THE SOFTWARE EFFORT USING USE CASE POINTS METHOD

M. Pramod Kumar
Research Scholar
Krishna University, Machilipatnam
Asst. Prof in SCNR Govt Degree College, Proddatur

Dr. M Babu Reddy
HOD, CS Department
Krishna University, Machilipatnam

Lavanya A L
CSE Dept
Asst. Prof in SVEC, TP Gudem

Abstract:-In any object oriented design the basic build block is use case diagram, which are prepared at early stages of SDLC .The use case diagrams are considered to be useful in the process of accurate estimates for software development project. This paper gives detailed analysis of size estimation methods based on use case points. We proposed a method called Size Optimization Technique (SOT) which takes use case points as input and is based on multiple least-squares regression. This approach consists the phases of (1) deals with use case point estimation, (2) obtained correction coefficient by applying multiple least-squares regression, (3) obtained new estimation by applying proposed estimation equation. We compare the results of Size Optimization Technique (SOT) with UCP and RE-UCP models. The comparisons of tested projects show that SOT has significantly outperformed the existing UCP and RE-UCP size estimation techniques.

Keywords: Software effort estimation, RE-UC Points, UC Points

1. Introduction

Effort estimation of software development process plays an important role in the software Engineering field. Effort estimation is the process of estimating the number of business activities of workers as well as how much time it takes to accomplish the software development process. the reliable and accurate software development effort estimation has always been encouraging for project managers [1].There are several approaches which can be put in to practice to estimate software effort. The function point analysis (FPA) is the one of the approaches, which has an official technical standard in ISO standardization [2]. It is descripted as set of functions and the analyst should understand the system in great detail. The FPA analysis called function point analysis is considered as a black box, the results of estimates only the effort of each individual system function. The estimation influenced by the personal opinion of analyst. It makes this method unsuitable for comparing productivity in software development process. Function Point Analysis considers external input and outputs. The processes that send data in to the boundary are called external inputs. The external outputs considered data from external interface files and internal logic files. An estimation technique called use case point method for early stage estimation developed by Gustav Karner [3].This method is based on use case diagrams which are commonly used as functional description of proposed software. There have been number of modification of original method, such as use case size points [4], extended use case points [6], adapted use case points[5], transaction path analysis[7]. In order to improve the efficiency effort estimation at the early stage of development, this research work concentrates on use cases for the purpose of estimation to identify refined model. The rest of the paper gives an insight view of methodology used in section 2. Section 3 shows experiment results and discussed the

effectiveness of each model. Section 4 show conclusion and future work.

Fi	Factors contributing to efficiency	Wi
1.	Familiar with Objectory	1.5
2.	Stable requirements	2.0
3.	Analyst capability	0.5
4.	Application experience	0.5
5.	Object oriented experience	1
6.	Motivation	1
7.	Difficult programming language	-1
8.	Part-time workers	-1

Table 1: Weighted Use Cases

2. Methodology

2.1 Description of UCP method

UCP method is an extension of function point analysis. In UCP method [2][3][6][8], actors and use cases are classified into simple, average and complex. An application programming interface or a non physical system user represented by a Simple actor. Average actor typically represents a system inter connected by network protocol and Complex actor who interacts to the system with the help of GUI. Table -1 represents an actor classification and weighing factor. The total unadjusted actor weights are calculated based on equation eq (1). Similarly the complexity of use case is based on number of transactions. The use case is defined as simple if the number of transaction is less than 4. If number of transaction is between 4 to 7 then the use case is defined as average. The use case defined as complex if the number of transaction is greater than 7.
The weights of unadjusted use cases are based on eq (2). The first factor is called the technical factors is applied to the unadjusted UCP. The technical complexity is already defined in the function point analysis (FPA). Table 3 represents technical factors. The second factor called environmental factors is applied to UCP. This factor defines some requirements which are non functional and these are shown in Table 4. Factors T1 to T13 and E1 to E8 have fixed weights. There is a significant factor for each value from 0 to 5. Where 0 indicates no impact, 3 indicate an average impact, 5 indicated as strong impact. Technical complexity factor (TCF) can be clarified by equation eq (3). The second factor called the environmental complexity factor (ECF) can be calculated by using equation eq (4). Final result called adjusted UCP can be obtained by applying the UCP equation eq (2) [2][3][6][8].The final UCP value represents software size but person-hour value is used to measure the function effort. To get final effort, each UCP is typically indicated as 20 man hours [3] or more generally by 15 to 30 man hours.

$$UUCP=UAW+UUCW \quad (1)$$

$$UCP=TCF*ECF*UUCP*PF \quad (2)$$

$$TCF=0.6+(0.01*TOTALFACTOR) \quad (3)$$

$$ECF=1.4+(-0.03*TOTALFACTOR) \quad (4)$$

Actor	Complexity	Description Weight
Simple	Through an API	1
Average	Through a text-based user interface	2
Complex	Through a GUI	3

Table 2: Weighted Actors

Fi Factors	Factors contributing to complexity	Wi
1.	Distributed systems	2
2.	Application performance objectives	1
3.	End user efficiency	1
4.	Complex internal processing	1
5.	Reusability	1
6.	Easy installation	0.5
7.	Usability	0.5
8.	Portability	2
9.	Changeability	1
10.	Concurrency	1
11.	Special security features	1
12.	Direct access for third parties	1
13.	Special user training facilities	1

Table 3: Technical Factors

Use Case Complexity	Number of Transactions	Weight
Simple	3	5
Average	4 to 7	10
Complex	More than 7	15

Table 4: Weighted Use Cases

2.2 Description of Re-UCP method

The extension of UCP is Revised Use Case Point (RE-UCP). In Re-UCP the functionality of the system is measured to evaluate the all use case points in the system. Actors use cases, environmental and other technical factors are further generalized to associate a particular impact factor on the specific use case activity. The Re-UCP uses different categorization of actor and use cases. In Re-UCP there are four categories (1) simple (2) average(3) complex(4) critical, along with the weighted parameters of 1, 2, 3, 4 respectively. There are four types of use cases (1) Simple (2) Average (3) Complex (4) Critical as shown in Table 5 and Table 6. If an actor interacts with systems API then actor is defined as simple, if an actor interacts with the system with the help of protocol driven interface then it is called average, if an actor interacts through graphical user interface then it is defined as complex, if an actor interacts with modules then the actor is defined as critical .The weighted parameters for simple is 1, average is 2, complex is 3 and critical is 4. Similarly if the number of transaction is less than or equal to 4 then it is simple use case, if the number of transaction is between 5 and 8 then it is average, if the number of transaction is between 9 to 15 then it is complex, if the number of transaction is greater than 15 then it is critical. To estimate the complete use case points of the system development, environmental and technical parameters need to be considered in development process and are called technical complexity factors (TCF) and environmental complexity factors (ECF). In Re-UCP the number of parameters in TCF increased from 13 to 14, the 14th parameter is scalability. It has weight 2 as shown in Table 7. To obtain the value of technical complexity factor (TCF) the equation (5) is used. The environmental complexity factors are E1 to

Fi	Factors contributing to efficiency	Wi
1.	Familiar with Objectors	1.5
2.	Stable requirements	2.0
3.	Analyst capability	0.5
4.	Experience in application environment	0.5
5.	Experience in Object oriented	1
6.	Motivation	1
7.	Difficult programming language	-1
8.	Part-time workers	-1

Table 5: Environmental Factors

E9 in UCP method, in Re-UCP project methodology is added as E9th factor and their corresponding weight factor is described in Table 8. The ECF value can be obtained from the equation (7). Thus the total number of use case points is calculated by multiplying UUCP, TCF, and ECF according to equation (8). The effort can be calculated by using 20 man-hours per UCP as suggested by Karner[3]. The equation (9) is used to convert the no of Re-UCP in to man-hours.

$TCF=0.6+ (0.1*\sum_{i=1}^{14}(TF_i))$ (6)

$ECF=1.4+ (-0.03*\sum_{i=1}^{9}(EF_i))$ (7)

RE-UCP=UUCP*TCF*ECF (8)

Effort= UCP*PERSON PER HOUR (9)

Use Case Complexity	Number of Transactions	Weight
Simple	<=4	5
Average	5 to 8	10
Complex	9 to 15	15
Critical	>15	20

Table 6: Weighted Use Cases [1]

Actor	Complexity	Description Weight
Simple	Through an API	1
Average	Through a text-based user interface	2
Complex	Through a GUI	3
critical	Through modules	

Table 7: Weighted Actors [1]

Fi Factors	Factors contributing to complexity	Wi
1.	Distributed systems	2
2.	Application performance objectives	1
3.	End user efficiency	1
4.	Complex internal processing	1
5.	Reusability	1
6.	Easy installation	0.5
7.	Usability	0.5
8.	Portability	2
9.	Changeability	1
10.	Concurrency	1
11.	Special security features	1
12.	Direct access for third parties	1
13.	Special user training facilities	1
14.	Scalability	2

Table 8: Technical Factors [1]

Fi	Factors contributing to efficiency	Wi
1.	Familiar with Objectory	1.5
2.	Stable requirements	2.0
3.	Analyst capability	0.5
4.	Application experience	0.5
5.	Object oriented experience	1
6.	Motivation	1
7.	Difficult programming language	-1
8.	Part-time workers	-1
9.	Project Methodology	1.0

Table 9 Environmental Factors

2.3 Description of Size Optimization Technique (SOT)

The proposed method Size Optimization Technique (SOT)[1] which is on the basis of commonly used approach of UCP technique is implemented by karner. The SOT consists series of steps as summarized in fig x which shows three phases.

Phase-I : The preparation phase which is used to calculate the correction coefficients β0, β1, β2. The least square regression is used for obtaining these coefficients. The following formula is used for calculations $\beta_{i1}=tuaw_{i1}+tuucw_{i1}$
the values of β1, β2 can be different for each individual dataset. T

phase-II : Use case point estimation in which the UCP parameters are calculated. This technique will be used only when there is old data points exist. The fundamental use case parameters can be calculated from equation (1-4) are used to obtain tuaw , uucw , tcf ,ecf parameters.

Phase-III : This phase called tuning phase in which the final UCP according to SOT will be estimated, which could be used to plan the project and later on for effort estimation. The calculation of UCP according to SOT is obtained from eq (10).

$$UCP_{SOT}=\beta0+\beta1*(UAW*TCF*ECF)+ \beta2*(UUCW*TCF*ECF) \quad (10)$$

Phase-1: Step 1:- Obtain UAW, UUCW ,TCF, ECF from completed projects
Step 2:- Compute X_1= UAW*TCF*ECF
X_2 = UUCW*TCF*ECF
Y= Real_P20
Step 3:- Apply multi linear regression to obtain β0, β1, β2.

Phase-2: Step 4:- Obtain parameters (UAW, UUCW ,TCF,ECF) from new projects data by using UCP

Phase-3: Step 5:- Compute new UCP using size optimization technique method using the following equation:
UCP_{SOT} =β0+β1*(UAW*TCF*ECF)+ β2*(UUCW*TCF*ECF)
Step 6:- Compute final project size estimation

Fig 1: Algorithm for size optimization technique (SOT)

3. Experiment planning

In this study the experiment deals with calculating SOT can be effectively increase the size estimation approach on the basis of UCP.To evaluate efficiency of SOT, an empirical validation approach was performed in which the first model called use case points(UCP) developed by Karner had been used as fundamental approach to set minimum performance level. The next approach is the revised UCP(Re-UCP) has used to compute size. The third model which may be called SOT was performed on same data set. Finally the efficiency of prediction of three implemented

approaches was compared to make a decision which approach is more efficient to predict software size. A statistical hypothesis was tested

H0: UCpoints = Re-UCP = SOT

Where no variation of efficiency in the process of prediction among use case points (UCP),Re-UCP and SOT approaches. There is no variation of errors in the process of estimation, the alternate hypothesis as

H1: UCP! = Re-UCP! = SOT

We have identified the variation in the process of prediction among UCponts, Re-UCP, SOT estimation models. There is a variation of errors in the estimation. Our approach makes the comparisons of SOT accuracy with the UCP and Re-UCP approaches with the help of

$$MRE_i=\sum_{i=1}^{N} \frac{abs(actualeffort_i-estimatedeffort_i)}{actualeffort_i} \quad (11)$$

$$MMRE=\frac{1}{N}\sum_{i=1}^{N} MER_i \quad (12)$$

$$PRED(x)=(1-(\frac{\sum_{i=1}^{n} abs(a.effort_i-p.effort_i)}{N}))*100 \quad (13)$$

$$TSS=\sum_{i=0}^{n}(real\_p20 \quad - predicted\ size)^2 \quad (14)$$

Predicted size stands for values obtained from ucp ,re-ucp or SOT. The main goal of results is to keep MMRE and TSS minimized and Pred (0.15) is maximized

4. Experimental outcomes and discussions:

In this study ucp method, RE-UCP method and our proposed method SOT were compared. The dataset [9] contains uaw, uucw, tcf and ecf are derived from use case points method .Real_p20 is a value which represents real project size in the form of use case points size when the projects were finished ucp ,re-ucp and t-test. T-test is an approach which can be used to check the null hypothesis which means that two populations which are distributed normally are the same. The t-test can be used to asses the magnitude of relative error (MRE).

3.1 Performance Evaluation

Magnitude of Relative Error (MRE), Mean Magnitude of Relative Error (MMRE) and percentage of prediction PRED (0.25)) are the performance measure accepted as the standard evaluation measures to predict the size and effort estimation. We also used total sum of squares which is the metric to evaluate prediction models and median of errors. The following eq (11) to (14) have been used for obtaining performance metrics. SOT models used same dataset for the evaluation and the evolution is at α=0.05 significance level. The dataset is divided in to two parts one is raw data and another one is test data, the raw projects datasets have taken role as historical data set. For the evaluation purpose we have used rest of data sets called test data the Table-11 shows statistical characteristics of test data.Table-12 shows that comparisons of estimation method performance ,from the table it can be observed that SOT produces significantly better results than UCP and RE-UCP. According to pred (0.15) SOT is more than twice better than UCP and RE-UCP .MRE was tested at significant level 0.05 and null hypothesis can be rejected because performance of SOT ,UCP and RE-UCP is not equal.

MIN	MAX	MEAN	s.d
288.75	294	291.5154	2.1267

Table 10: Statistical characteristics of test data

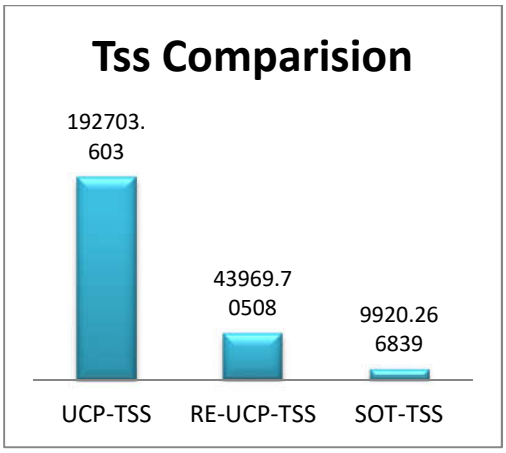

Fig 2: Tss Comparisons

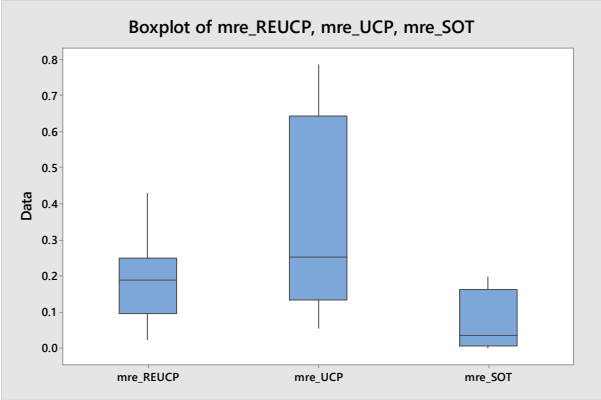

Fig 3: Box-plot of MMRE of UCP,RE-UCP,SOT

	Mean	Standard deviation	No of samples
UCP	0.376	0.263	11
RE-UCP	0.188	0.113	11
SOT	0.07	0.077	11

Table 11 Performance comparisons of estimation models for sample data

MODEL	T-VALUE	P-VALUE	DF	NO OF SAMPLES
UCP-SOT	3.74	0.03	11	11
UCP-RE_UCP	2.21	0.045	13	11
RE_UCP-SOT	-2.81	0.012	17	11

Table 12: T-test Comparisons

	MMRE	PRED(0.15)	TSS
UCP	0.378766	0.27	1,92,703.6
RE-UCP	0.187636	0.36	43,969.71
SOT	0.071575	0.727273	9,920.267

Table 13: Performance comparisons of estimation models

5. CONCLUSION

We proposed a new size estimation method called SOT ,in this method we calculated UAW, UUCW ,ECF,TCF parameters based on ucp for each historical project we have obtained three coefficients β0,β1, β2 by using multilinear regression .we used these coefficient to refine the size estimation, we compared SOT with UCP and RE-UCP estimation models the SOT significantly better than UCP and RE-UCP estimation methods, with the mean MRE i.e 30% better than ucp and 11% better than RE-UCP SOT results the size estimation work flow and describes an approach how a size of software project could be estimated therefore in our future work we will use clustering concepts to improve the accuracy of the estimation.

REFERENCES

[1] Lynch J., and Chaos M., October 2009. The Standish Group. Boston.Available online: http://www.standishgroup.com/ newsroom/chaos_2009.

[2] Robiolo G, Orosco R. Employing use cases to early estimate effort with simpler metrics. Innovations in Systems and Software Engineering. 2008; 4(1):31–43. doi: 10.1007/s11334-007-0043-y

[3] Karner G. Metrics for objectory', December 1993, Diploma, University of Linkoping, Sweden, No. LiTHIDA-Ex-9344. 21.

[4] Braz MR, Vergilio SR, editors. Software effort estimation based on use cases. Computer Software and Applications Conference, 2006 COMPSAC'06 30th Annual International; 2006: IEEE.

[5] Mohagheghi P, Anda B, Conradi R. Effort estimation of use cases for incremental large-scale software development. 2005:303–11. doi: 10.1109/icse.2005.1553573

[6] Wang F, Yang X, Zhu X, Chen L. Extended Use Case Points Method for Software Cost Estimation. 2009:1–5. doi: 10.1109/cise.2009.5364706

[7] Robiolo G, Badano C, Orosco R. Transactions and paths: Two use case based metrics which improve the early effort estimation. 2009:422–5. doi: 10.1109/esem.2009.5316021

[8] Ochodek M, Nawrocki J, Kwarciak K. Simplifying effort estimation based on Use Case Points. Information and Software Technology. 2011; 53(3):200–13. doi: 10.1016/j.infsof.2010.10.005

[9] Silhavy20171, title = "Analysis and selection of a regression model for the Use Case Points method using a stepwise approach ", journal = Journal of Systems and Software ", volume = "125", number = "", pages = "1 - 14", year = "2017", note = "", issn = "0164-1212", doi = http://dx.doi.org/10.1016/j.jss.2016.11.029"