Efforts Estimation by Use Case Point using Experience Data

Chetan Nagar[1], Anurag Dixit, PhD.[2]

[1]Ph.D Student, Mewar University (Gangrar) Chittodgarh Rajasthan India

[2] Dean-Professor (CS/IT) BRCM CET,Bahal Bhiwani

ABSTRACT

Many software efforts estimation models and methods are invented to make efforts estimation accurate. Unfortunately no model or method is suitable for all kind of project and situations. it is frequently suggested that using experience data, estimation models and checklists can increase software effort estimation accuracy. However, there has been limited empirical research on the subject. It was found that in projects where experience data was utilized in the estimation process, they experienced a lesser magnitude of effort overruns. The use of a checklist also appeared to increase estimation accuracy. The utilization of an estimation model did not appear to have a substantial impact [08].This paper is suggesting that use of estimation model can also produce good estimation results, but historical data is always necessary to assist the estimation. We can use historical data to improve the result of Use Case Point and COCOMO model .In our research we have gain 10% improvement in Use case Point model with use of historical data. This paper is also suggesting that a strong monitoring policy is always required to make your estimation as a success.

Keywords*:* Software estimation, experience data, estimation models, checklists, COCOMO, Use Case Point.

1. INTRODUCTION

We should accept that Estimation by Analogy and Expert Estimation are good estimation technique, it means not that estimation model cannot produce good result. Historical data provide a strong base to our prediction. It makes our prediction better. Use of historical data can also improve the performance of estimation model that we have shown in this paper.

Estimation is an important part of software engineering process, and the ability to produce accurate effort estimates has an impact on key economic processes, including budgeting and bid proposals. Projects estimated optimistically might be selected instead of a project that has been estimated pessimistically [2]. If organizations want to improve the accuracy of their employees‟effort estimates, employees must be trained to make better estimates. It has already been established that estimation ability does not increase with experience [10].

In [4], Jørgensen raises the issue of why estimation models are not applied by project managers more frequently. He argues that the lack of evidence for their efficacy may be the most significant reason.

Most of the estimation done today is expert-based. Research has shown that the average effort overrun in software development projects is about 30%-40% [1].

Today we are referring old data only in Analogy Based Estimation and Expert Estimation. We can also use the historical data to improve the accuracy of an estimation model.

2. METHODS OF EFFORTS ESTIMATION

It is necessary understanding the principals of each estimation method to choose the best. Because performance of each estimation method depends on several parameters such as complexity of the project , duration of the project, expertise of the staff, development method and so on [14].

COCOMO and Use Case Point is two most popular models used for estimation. In the COCOMO we have to predict two things first is how much KLOC will required to build that project and second 22 Efforts Adjustment Factors .Historical data can guide us to predict how much KLOC will required to build the project and it can also help in the prediction of EAF. For the effective utilization it is necessary that database must be well managed. We must record the values of the parameters and reason why we had chooses that value .A database of previous successful project must be maintain for future reference.
Use Case Point is another popular in which we need to predict Actors, Use Case, TCF and EF.

A. COCOMO[9]

One after one three models of COCOMO given by Barry Boehm:

- i. Simple COCOMO.
- ii. Intermediate COCOMO.
- iii. Advance COCOMO

i.Simple COCOMO:- It was the first model suggested by Barry Boehm, which Follows following formula:

Efforts= a*(KLOC) ^{b}

Here a and b are complexity factor.

TABLE I
Complexity Factors

Model	A	B
Organic (simple in terms of size and complexity	3.2	1.05
Semi-ditched (average in terms of size and complexity	3.0	1.12
Embedded (Complex)	2.8	1.20

ii *Intermediate COCOMO:*-Previous model does not include the factors which can affect the efforts. Intermediate COCOMO includes 17 factors that can affect the efforts estimation.

Efforts= a*(KLOC) ^{b} *EAF

Here a and b are complexity factor.

TABLE II
Complexity Factors

Model	A	B
Organic (simple in terms of size and complexity	3.2	1.05
Semi-ditched (average in terms of size and complexity	3.0	1.12
Embedded (Complex)	2.8	1.20

Following are Efforts Adjustment Factors used in Intermediate COCOMO. Typical values for EAF range from 0.9 to 1.4.

TABLE III
Cost Drivers

S NO	Cost Driver	Value	Description
1	DATA		Database size.
2	CPLX		Product complexity.
3	TIME		Execution time constraint.
4	STOR		Main storage constraint.
5	**RUSE**		Required reusability.
6	**DOCU**		Documentation match to life-cycle needs.
7	**PVOL**		Platform volatility.
8	SCED		Scheduling factor.
9	RELY		Required reliability.
10	TOOL		Use of software tools.
11	APEX		Application experience.
12	ACAP		Analyst capability.
13	PCAP		Programmer capability.
14	PLEX		Platform experience.
15	LTEX		Language and tools experience.
16	**PCON**		Personnel continuity.
17	**SITE**		Multisite development.

Scale factors are new in COCOMO II. The effect of scale factor is in 1.01 to 1.26 ranges

TABLE IV
New Cost Drivers

S NO	Cost Driver	Value	Description
18	PREC		Precedence.
19	PMAT		Process maturity.
20	TEAM		Team cohesion.
21	FLEX		Development flexibility.
22	RESL		Architecture and risk resolution.

What we have to predict in the COCOMO, first we have to predict KLOC, second parameters specified in Table-III and Third Parameters specified in Table-IV. Experience data can help us in prediction .Now suppose we have a rich database for such kind of project so which projects can be taken as reference, Answer is that we must keep two parameters in mind first we have to take latest project and second we have to take successful project.

B. Use Case Point [3] [13].

The Use Case Points (UCP) method provides the ability to estimate the man hours a software project requires from its use cases. Based on work by Gustav Karner [1], the UCP method analyzes the use case actors, scenarios, and various technical and environmental factors and abstracts them into an equation.

The UCP equation is composed of three variables:
1. Unadjusted Use Case Points (UUCP).
2. The Technical Complexity Factor (TCF).
3. The Environment Complexity Factor (ECF).

A. Calculate no of Actors:-We use following table to calculate no of Actors used in project

TABLE V
Actor Calculation

Actor Type	Description	Quantity	Weight Factor	Subtotal
Simple	Defined API		1	
Average	Interactive or protocol driven interface		2	
Complex	Graphical user interface		3	
Total Actor Points				

B. Calculate no of Use Cases:-We use following table to calculate no of Use Cases used in project

TABLE VI
Use Case Calculation

Use Case Type	Description	Quantity	Weight Factor	Subtotal
Simple	Up to 3 transactions		5	
Average	4 to 7 transactions		10	
Complex	More than 7 transactions		15	
Total Use Cases				

UUCP =Weighted Actors + Weighted Use Cases
UCP=UUCP*TCF*EF
Calculate TCF (Technical Complexity Factor)

List of Technical factors where weight factor rate from 0-2 and project rating rate from 0-5

TABLE VII
Technical Complexity Factors

Technical Factor	Factor Description	Wight Factor	Project Rating	Sub Total
T1	Must have a distributed solution	2		
T2	Must Respond to specific performance objective	1		
T3	Must meet end user efficiency desired	1		
T4	Complex internal processing	1		
T5	Code must reusable	1		
T6	Must be easy to install	0.5		
T7	Must be easy to use	0.5		
T8	Must be portable	2		
T9	Must be easy to change	1		
T10	Include special security feature	1		
T11	Must provide direct access to third parties	1		
T12	Requires special user training facilities	1		
T13	Must allow concurrent user	1		
TOTAL				

TCF= (0.01 * TC factor) + 0.6
Calculate EF (EXPERIENCE FACTOR)

TABLE III
Experience Factors

Experience factor	Factor Description	Wight Factor	Project Rating	Sub Total
E1	Familiar with FTP software Process	1		
E2	Application Experience	0.5		
E3	Paradigm Experience	1		
E4	Lead analyst capability	0.5		
E5	Motivation	0		
E6	Stable Requirements	2		
E7	Part time workers	-1		
E8	Difficulty of programming Language	-1		
TOTAL				

EF= (-0.03 *E factor) + 1.4

In the Use Case Point approach we have to predict no of Actor (Table-V), no of Use Cases (Table-VI), TCF (Table-VII) and EF (Table-VIII).Record of latest and successful project can help us in prediction of these values.

An early project estimate helps managers, developers, and testers plan for the resources a project requires. As the case studies indicate, the UCP method can produce an early estimate within 20 percent of the actual effort, and often, closer to the actual effort than experts and other estimation methodologies [13].

3. USE OF HISTORICAL DATA IN MODEL

As we know that in COCOMO we need to predict the KLOC and other 22 parameter which is called Efforts Adjustment Factors. In the Use Case Point approach we have to predict the 13 Technical Complexity Factor and 08 Experience Factor.

Historical data provide us guidelines to predict these parameters. Historical improve our prediction .The idea of recording and utilizing data from experience when estimating software development effort is not new [5]. One of the strengths of this approach is that estimates are based on actual experience [6]. The problem is the often very unique nature of software development projects, which makes it difficult to assess how similar a new project is to a previous one. Estimation by analogy is, or at least has been, widely utilized in the software industry [7**].**

We have taken some projects of a small software development company and estimate the efforts for these project using Use Case Point approach without taking reference of historical data .Again we have estimated the efforts for the same project using historical data and we have found that on an average we got average 10% of improvement. This improvement definitely decreases the MRE.

TABLE IX
Comparison of Results

Name of Project	Effort Estimated by Use Case Point (in Man-Hours)	Effort Estimated by Use Case Point with experience data (in Man-Hours)
A	1042	1138
B	917	1015
C	822	910

Fist time we had predict the values of parameters on the basis of our experience and project requirements. But in second time we use historical data to predict the value of parameters and we have got such change.

During the study we have found that if we have incorrectly predict even a single parameter and make of difference of value one then we get difference of 4 UCP (On an average) on per 1000 UUCP. That mean we are losing 80 man hours (4 UCP*20 Man-hours/UCP) on a single value of TCF or EF.

As we know that we have 13 TCF and 08 EF. So 21 times we need to predict the correct value .On a single miss prediction we will got a major difference.

We had work on a project has around 1000 UUCP and try to illustrate the above study.

TABLE X
Estimation Results for UUCP=1000

∑ TCF	TCF	∑ EF	EF	UCP
52	1.12	32	0.44	493
51	1.11	32	0.44	488
50	1.10	32	0.44	484

TABLE XI
Estimation Results for UUCP=2000

∑ TCF	TCF	∑ EF	EF	UCP
52	1.12	32	0.44	985
51	1.11	32	0.44	977
50	1.10	32	0.44	968

In the above study we have assume that we had miss predicted only one parameter by value one. We can lose more UCP if we incorrectly predict the many parameters with more value.

4. CONCLUSION

In the study we have seen that prediction is important in efforts estimation .Your estimation will be better if you can predict better. Historical data play vital role in prediction, it recommend us what we have to do. Use of model with historical data can produce good result that we have seen. This paper shows that not only the check list, analogy based estimation, or expert estimation can perform better, estimation model can also perform better but assistance of historical data is must.

Neither estimation strategy has been shown to be superior in all cases [11]. All the models could not predict the actual against either the calibration data or validation data to any level of accuracy or consistency. No model is best for all situations and environment. [12]

A lot of estimation models and methods are suggested by the researchers ,but no one is best suitable for all projects and all software companies .A good monitoring policy is always required to make your estimation as a success .Every time we have to check the gap between actual and estimated and take the actions to bridge the gap.

Every model and method required a little bit of modification according to your local environment .so modify the method according to your requirement and use it , it will produce better results.

5. REFERENCES

[1] K. Moløkken-Østvold and M. Jørgensen, "AReview of Surveys on Software Effort Estimation," in *2003 ACM-IEEE International Symposium on Empirical Software Engineering (ISESE 2003)*, Frascati Monte Porzio Catone (RM), ITALY, 2003,pp. 220-230.

[2] M. Jørgensen and S. Grimstad, "Overoptimism in Software Development Projects:“The winner’s curse”," in *IEEE CONIELECOMP*, Puebla, Mexico, 2005, pp. 280–285.

[3] Karner Gautav “Resource Estimation for objector project “Objective system SF AB 1993.

[4] M. Jørgensen, "Forecasting of Software Development Work Effort: Evidence on Expert Judgment and Formal Models," Accepted for International Journal of *Forecasting*, 2007

[5] E. A. Nelson and A. Force, *Management Handbook for the Estimation of Computer Programming Costs*: System Development Corp.; Distributed by Clearinghouse for Federal Scientific and Technical Information, 1967.

[6] F. Walkerden and R. Jefferey, "An empirical study of analogy-based software effort estimation," *Empirical Software Engineering*, vol. 4, pp. 135-158, 1999

[7] Kristian Marius Furulund1 and Kjetil Moløkken-Østvold “**Increasing Software Effort Estimation Accuracy - Using Experience Data, Estimation Models and Checklists** ” Seventh International Conference on Quality Software (QSIC 2007) pp 342-347

[8] Bohem,” Software Engineering Economics”, Prentice Hall, 1981.

[9] M. Jørgensen and D. I. K. Sjøbert, "Impact of experience on maintenance skills," *Journal of Software Maintenance and Evolution: Research and Practise*, vol. 14, pp. 123-146, 2002.

[10] M. Jørgensen, "A Review of Studies on Expert Estimation of Software Development Effort," *The Journal of Systems and Software*, vol. 70, pp. 37-60, 2004.

[11] Saleem Basha , Dhavachelvan P “Analysis of Empirical Software Effort Estimation Models” (IJCSIS) International Journal of Computer Science and Information Security, Vol. 7, No. 3, 2010

[12] Carroll, Edward R. “Estimating Software Based on Use Case Points.” 2005 Object-Oriented, Programming, Systems, Languages, and Applications (OOPSLA) Conference, San Diego, CA, 2005.

[13] [1]Vahid Khatibi, [2]Dayang N. A. Jawawi “Software Cost Estimation Methods: Review”, Journal of Emerging Trends in Computing and Information Sciences Volume 2 No. 1 January 2011.