

A New Estimation Model for Small Organic Software Project

Wan-Jiang Han, Tian-Bo Lu, and Xiao-Yan Zhang
School Of Software Engineering, Beijing University of Posts and Telecommunication, Beijing, China
Email: {hanwanjiang, lutb, xiaoyan}@bupt.edu.cn

Li-Xin Jiang
Department of Emergency Response, China Earthquake Networks Center, Beijing, China
Email: jlx@seis.ac.cn

***Abstract*—It is very hard to estimate software development effort accurately. So far, no model has proved to be successful at effectively and consistently estimating software development effort or cost. So it is useful to research a particular model for a particular type of project. A new model for small organic project is proposed for software effort estimation. This model is based on actual project data and well-established theories, using Gauss-Newton model to calibrate the parameters of the COCOMO model, using Fuzzy logic models to maintaining the merits of the COCOMO model. In particular, this model has been successfully used in some small project, and has demonstrated great potential to predict software cost more accurately.**

***Index Terms*—software cost estimation, software effort estimation, small project, organic project, Fuzzy, Constructive Cost Mode**

I. Introduction

As there are a great variety of software development project in many areas, software estimation is becoming more and more important in effective software project management, especially in cost estimation. Accurate software estimation can provide powerful assistance when software management decisions are being made; for instance, accurate cost estimation can help an organization to better analyze the feasibility of a project and to effectively manage the software development process, therefore, greatly reducing the risk.

Lots of attempts [1], [2], [3], [4], [5], [6] have been made to solve the problem in the last few decades, no approach has proven to be successful in effectively and consistently predicting software effort.

So it is useful to research a model for particular type of project. This paper offers a new model to estimate the software effort for small organic project based on the data of actual projects and it is the improvement of COCOMO.

Manuscript received October 2, 2012; revised March 7, 2013.

We have taken into consideration the features of the effort estimation problem and some techniques and have proposed a new model. We imply Gauss-Newton model and Fuzzy model to the COCOMO, and have validated our approach with later project data.

Gauss-Newton model are used in our model to automatically calibrate the parameters of the COCOMO model.

II. Background

Our new model is based on the standard COCOMO model, the gauss–newton algorithm and the fuzzy logic model, we briefly review these techniques.

A. COCOMO Model

The COCOMO model originally published by Boehm is one of most popular parametric cost estimation models of the1980s[1] , [11]. At present, the model is still the most important in the software field. In the middle of 1990's, Boehm proposed COCOMO II[1], [2] based on COCOMO81 . Nowadays, it is considered as one of the most extensively used and approved software estimating model in academia and industrial area. The basic principle of COCOMO model is to express effort with software size and a series of cost factor, as the following equation:

$$PM = A \times (\sum Size)^{\sum_B} \times \prod (EM)$$

B. Gauss–Newton Algorithm

Gauss–Newton algorithm is a method used to solve non-linear least squares problems. The method is named after the mathematicians Carl Friedrich Gauss and Isaac Newton [7], [8].

Non-linear least squares problems arise for instance in non-linear regression, where parameters in a model are sought such that the model is in good agreement with available observations.

Given m functions $r = (r_1,\ldots,r_m)$ of n variables $\beta = (\beta_1,\ldots,\beta_n)$, with $m \geq n$, the Gauss–Newton algorithm finds the minimum of the sum of squares, as in (1).

$$S(\beta) = \sum_{i=1}^{m} r_i^2(\beta) \quad (1)$$

doi:10.4304/jsw.8.9.2218-2222

Starting with an initial guess $\beta^{(0)}$ for the minimum, the method proceeds by the iterations, as in (2).

$$\beta^{(s+1)} = \beta^{(s)} + \Delta, \quad (2)$$

where

$$S(\beta^{(S)} + \Delta) = S(\beta^{(S)}) + \left[\frac{\partial S}{\partial \beta_i}\right]\Delta + \frac{1}{2}\Delta^{\mathrm{T}}\left[\frac{\partial^2 S(\beta)}{\partial \beta_i \partial \beta_j}\right]\Delta$$

Δ is a small step. We then have.

If we define the Jacobian matrix as in (3).

$$J_r(\beta) = \frac{\partial r_i}{\partial \beta_j}\Big|_{\beta}, \quad (3)$$

we can replace

$$\left[\frac{\partial S}{\partial \beta_i}\right] \text{ with } J_r{}^{\mathrm{T}} r$$

and the Hessian matrix can be approximated by (4).

$$S(\beta^{(S)} + \Delta) \approx S(\beta^{(S)}) + J_r{}^{\mathrm{T}} r\Delta + \frac{1}{2}\Delta^{\mathrm{T}} J_r^{\mathrm{T}} J_r \Delta \quad (4)$$

(assuming small residual), giving: $J_r^{\mathrm{T}} J_r$ .We then take the derivative with respect to Δ and set it equal to zero to find a solution as in (5).

$$S'(\beta^{(S)} + \Delta) \approx J_r{}^{\mathrm{T}} r + J_r^{\mathrm{T}} J_r \Delta = 0 \quad (5)$$

This can be rearranged to give the normal equations which can be solved for Δ as in (6).

$$(J_r^{\mathrm{T}} J_r)\Delta = -J_r^{\mathrm{T}} J_r \quad (6)$$

In data fitting, where the goal is to find the parameters β such that a given model function $y=f(x, \beta)$ fits best some data points (x_i, y_i), the functions ri are the residuals.

$$r_i(\beta) = y_i - f(x_i, \beta)$$

Then, the increment Δ can be expressed in terms of the Jacobian of the function f, as in (7).

$$(J_f{}^{\mathrm{T}} J_f)\Delta = J_f{}^{\mathrm{T}} r \quad (7)$$

C. Fuzzy Logic Models

A fuzzy system [9] is a mapping between linguistic terms, such as ‘‘high complexity’’ and ‘‘low cost’’ that are attached to variables. Thus an input into a fuzzy system can be either numerical or linguistic with the same applying to the output. A typical fuzzy system is made up of three major components: fuzzifier, fuzzy inference engine (fuzzy rules) and defuzzifier. The fuzzifier transforms the input into linguistic terms using membership functions that represent how much a given numerical value of a particular variable fits the linguistic term being considered. The fuzzy inference engine performs the mapping between the input membership functions and the output membership functions using fuzzy rules that can be obtained from expert knowledge. The greater the input membership degree, the stronger the rule fires, thus the stronger the pull towards the output membership function.

Triangular fuzzy numbers are a subset of fuzzy sets with properties that make them well suited for modeling and design-type activities. Specifically, it has a triangular shape represented by the triple <a, b, c>,like Fig.1.

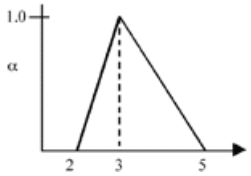

Figure 1. Triangular fuzzy numbers

III. The Estimation Model for Small Organic Project

The estimation model for small organic project is a process that takes Line of Codes (LOC) as inputs in order to estimate the workloads. So our goal is to fit a curve to data from actual software projects.

A. The Model Function Form

The results of research and practice show that the relationship between Code Line and effort is nonlinear [10][11]. And from the trend of the data fitting curve, we also got that non-linear regression analysis is fit for our model. Hence, we applied nonlinear regression fitting method to form our model in this paper. We tried to gain an equation as a relation function between LOC and effort, shown as in (8).

$$y = f(x) \quad (8)$$

On the other hand, basing on COCOMO 81 model, we adjusted the form of the model, shown as in (9).

$$y = a \times x^b \quad (9)$$

In the Equation (9), y is represented for the human resource needed (person hour), x is represented for Code Lines, “a” and “b” are both parameters.

B. The Fitting Procedure for the Model

Now let’s look at the project data, as in Table Ⅰ, which are for the type of small organic project.

TABLE I.

PROJECT DATA

No.	1	2	3	4	5	6	7	8	9
LOC	41	132	144	176	194	255	291	378	591
Person-Hour	6	10	11	16	22	30	32	35	42

It is desired to find a model function of the form of (9), that is to say, the model function is $PH = a \times LOC^b$,which fits best the data in the least squares sense, with the parameters a and b to be determined.

Denote by x_i and y_i the value of LOC and the Person-Hour from the Table Ⅰ, i=1,…,9. We will find a and b such that the sum of squares of the residuals,

$$r_i = y_i - a \times x_i^{\ b}, (i=1,\ldots,9) \text{ is minimized.}$$

after seven iterations of the Gauss–Newton algorithm the optimal values a=0.3709 and b=0.7547 are obtained. The plot in Fig. 2 shows the curve determined by the model for the optimal parameters versus the observed data.

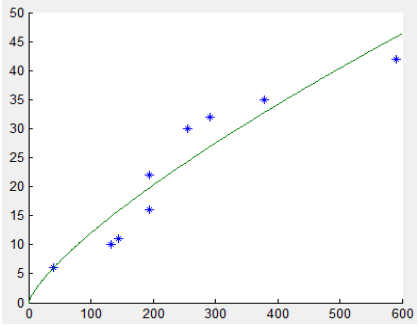

Figure 2. LOC-Person Hour

So, we have achieved the expression of our estimation model as follows:

$$PH = 0.3709 \times LOC^{0.7547} \quad (10)$$

C. The Adaptation of the Model

In order to improve estimating accuracy of our model, we also consider the factors that impact the effort. Therefore, the equation of our model can be adjusted into (11).

$$PH = 0.3709 \times LOC^{0.7547} \times F \quad (11)$$

In Equation (11), F is a multiplier which is a correction factor to our model. And F is expressed as in (12).

$$F = \sum_{i=1}^{5} (w_i \times CD_i) \times \prod_{i=1}^{5} CD_i \quad (12)$$

In Equation (12), CD means Cost Driver, w is the Weight of Cost Driver. According the character of small organic project, we choose five cost drivers, which are PREX (experience), PERS (skill or capability), RCPX (reliability and complexity), PDIF (platform difficulty), SCED (Required development schedule), referred to cost drivers of CocoMo II. Each cost driver represents one factor that contributes to the development effort. We use "CD_1", "CD_2", "CD_3", "CD_4", "CD_5" to represent "PREX", "PERS", "RCPX", "PDIF", "SCED'', respectively.

Since the importance of every cost driver is different, we maintain the merits of correction factor of COCOMO model.

Rating of every cost driver is linguistic terms such as "very low", "low", "nominal", "high", "very high", "extra high", and the value of every rating can get by referring COCOMO II model.

w is the ratio o*f* these cost driver, also a key of fuzzy evaluation. When we compare with each of the two cost driver, it is difficult to describe their importance by number, so we use the Triangular Fuzzy Number, by the sequence from high to low to ensure the weights of each cost driver[12].

1: For these cost drivers, considering the importance for them, there should be PERS (CD_2) > PREX (CD_1)> RCPX (CD_3)> PDIF (CD_4)> SCED (CD_5). Here comes to the Triangular fuzzy judgment matrix, like Table Ⅱ.

TABLE II.

TRIANGULAR FUZZY JUDGMENT MATRIX OF THESE COST DRIVER PROJECT DATA

CD	CD1	CD2	CD3	CD4	CD5
CD1	(1,1,1)	(1/3,1/2, 1)	(2,3,4)	(3,4,5)	(4,5,6)
CD2	(1,2,3)	(1,1,1)	(3,4,5)	(4,5,6)	(5,6,7)
CD3	(1/4,1/3, 1/2)	(1/5,1/4. 1/3)	(1,1,1)	(2,3,4)	(3,4,5)
CD4	(1/5,1/4, 1/3)	(1/6,1/5, 1/4)	(1/4,1/3,1 /2)	(1,1,1)	(2,3,4)
CD5	(1/6,1/5, 1/4)	(1/7,1/6, 1/5)	(1/5,1/4,1 /3)	(1/4,1/3,1 /2)	(1,1,1)

2: Computing the weights of cost driver.

(1) From Table Ⅱ, we can get the triangular fuzzy weight vector:

w1= (0.29, 0.29, 0.30),
w2= (0.43, 0.44, 0.42),
w3= (0.15, 0.15, 0.15),
w4= (0.082, 0.081, 0.080),
w5= (0.049, 0.045, 0.045),

(2) For every row, Adding each element , we can get :

s1= (2.61, 3.78, 5.08),
s2= (1.84, 1.95, 2.78),
s3= (6.45,8.58,10.83),
s4= (10.25, 13.33, 16.5),
s5= (15, 19, 23)

(3) Calculating the maximum Eigen value vector λmax, we can get the Max Eigen value $E(\lambda max)$= 5.27.

(4) Consistency checking

CI=0.067, CR=0.054< 0. 1, And $E(\lambda max)$ < order critical maximum eigenvalue. Hence, this Triangular fuzzy matrix satisfy the consistency check.

3: Calculating the Triangular fuzzy weight vectors expectations value

$$E(w_1) = 0.30,$$
$$E(w_2) = 0.43,$$
$$E(w_3) = 0.15,$$
$$E(w_4) = 0.08,$$
$$E(w_5) = 0.04,$$

From these expectations value, weights of each cost driver can be shown as in Table Ⅲ.

TABLE III.
EACH WEIGHTS OF COST DIVER.

Cost driver	PREX	PERS	RCPX	PDIF	SCED
w	0.30	0.43	0.15	0.08	0.04

Above all, our estimation model for small organic software project is shown as the following equation:

$$PH = 0.3709 \times LOC^{0.7547} \times \sum_{i=1}^{5}(w_i \times CD_i) \times \prod_{i=1}^{5} CD_i$$

We can get w_i from Table Ⅲ, CD_i from COCOMO Ⅱ model.

D. The Evaluation of the Model

We can use MRE (Magnitude of Relative Error) to evaluate the accuracy of estimation results. MRE can compare actual value and estimated value, which can be described as follows :
MRE= (ActualValue-EstimaedValue)/ActualValue. In fact, MMRE is a more useful evaluation tool, $MMRE = \frac{1}{n} \sum_{i=1}^{i=n} MREi$ .To evaluate our model, we calculated out that MMRE is equal to 8.8% from the follow-up 8 projects.

IV. IN THE CASE OF THE APPLICATION

The small project estimation model has been adapted to several projects and it has received really good effects. For instance, a case of small organic project, the input is 500 LOC, we estimate the effort will be 40.3872 PH before adaptation, shows in Fig. 3. Then we take correction to the value with cost drivers. The values of each factor are shown below: *PREX* =1 (nominal), *PERS* =0.86 (high), *RCPX* =1 (nominal), *PDIF* =1 (nominal), *SCED* =1 (nominal), that is to say, *F*=0.8083. So, the final result of estimation is *Effort*=32.645 Person-Hours. In this case, the project actually took 36 Person-Hours, *MRE*=9.3%, this value basically fits the evaluation value from the model.

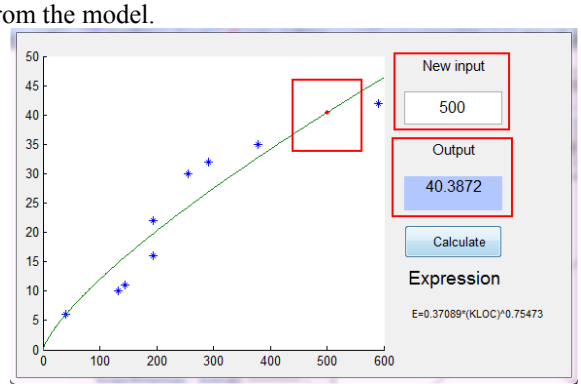

Figure 3. A case of software estimation

On the other hand, we also applied other methods to estimate the effort of this case, but the *MRE* > 20%. So our model is more suitable to this type of project.

V. CONCLUSIONS

Software development is a complex process, so is it very difficult to predict software development effort accurately. This paper put forward a software effort estimation model for small organic project. The model is based on actual project data and well-established theories which provides a promising tool to deal with many difficulties of small software estimation; thus, it is useful for project management. In particular, this model has been successfully used in some small project, and has demonstrated great potential to predict software cost more accurately.

At the same time, other type of software estimating model can also be formed in terms of methods of this model. It can provide assistance in project planning.

The advantage of this model is simplicity, practicability and automated administration. Next improvement thought of the model is 1) model improvement in connection with large scale project. 2) gradual optimization of correction factor computation.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61170273).

REFERENCES

[1] B. W. Boehm, *Software Engineering Economics*, Prentice Hall PTR, 1981.
[2] B. W. Boehm, *et al*. *Software Cost Estimation with COCOMOII*, Prentice Hall, 2000.
[3] S. Chulani, "Bayesian analysis of software cost and quality models," Ph.D. Dissertation, University of Southern California, LosAngeles, 1999.
[4] M. Shepperd and G. Kadoda, "Comparing software prediction techniques using simulation," *IEEE Trans. Software Eng,* vol. 27, no. 11, pp. 1014–1022, November 1999.
[5] A. B. Nassif, L. F. Capretz, and D. Ho, "Software estimation in the early stages of the software life cycle," in *International Conference on Emerging Trends in Computer Science, Communication and Information Technology*, 2010, pp. 5-13.
[6] R. J. Madachy, "Heuristic risk assessment using cost factors," *IEEE Software*, vol. 14, no. 3, pp. 51–59, May/June 1997.
[7] P. K. Subramanian, "Gauss-Newton methods for the complementarity problem," *Journal of Optimization Theory and Applications*, vol. 77. no. 3, pp. 467-482, June 1993.
[8] M. Y. Yerina and A. F. Izmailov, "The Gauss-Newton method for finding singular solutions to systems of nonlinear equations," *Computational Mathematics and Mathematical Physics*, vol. 47, no. 5, pp. 748-759, May 2007.
[9] R. Fuller, *Introduction to Neuro-Fuzzy Systems*, Physica-Verlag, 2000.

[10] E. Castillo, A. S. Hadi, and R. Minguez, "Diagnostics for non-linear regression, Department of Applied Mathematics and Computational Sciences, University of Cantabria, Santander; Spain, University of Castilla-La Mancha, Ciudad Real, Hadi; Minguez," *Journal of Statistical Computation and Simulation*, September 2009.
[11] Li-Xin Jiang and Wan-Jiang Han, "Research on Size Estimation Model for Software system Test based on testing steps and Its Application", *Computer Science and Information Processing (CSIP), 2012 International Conference* , pp. 1245-1248.
[12] Han Wan-jiang and Lu Tian-bo, "Study On Quality Evaluation Model Of Communication System", *System Science, Engineering Design and Manufacturing Informatization (ICSEM), 2012 3rd International Conference,* pp. 1-4.

Wan-Jiang Han was born in HeiLongJiang province, China, 1967. She received her Bachelor Degree in Computer Science from Hei Long Jiang University in 1989 and her Master Degree in Automation from Harbin Institute of Technology in 1992.

She is an assistant professor in School Of Software Engineering, Beijing University of Posts and Telecommunication, China. Her technical interests include software project management and software process improvement.

Tian-Bo Lu was born in Guizhou Province, China, 1977. He received his Master Degree in computer science from Wuhan University in 2003 and his PH.D Degree in computer science from the Institute of Computing Technology of the Chinese Academy of Sciences in 2006.

He is an Associate professor in School of Software Engineering, Beijing University of Posts and Telecommunications, China. His technical interests include information and network security, trusted software and P2P computing.

Xiao-Yan Zhang was born in Shandong Province, China, 1973. She received her Master Degree in Computer Application in 1997 and her PH.D Degree in Communication and information system from Beijing University of Posts and Telecommunication, China, in 2011.

She is an Associate professor in School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing, China. Her technical interests include software cost estimation and software process improvement.

Li-Xin Jiang was born in HeiLongJiang province, China, 1966. He received his Bachelor Degree and Master Degree in physical geography from Beijing University in 1989 and 1992.

He is a professor in the department of Emergency Response of China Earthquake Networks Center. His technical interests include software cost estimation and Emergency Response software development.