Journal of Information & Communication Technology
Vol. 2, No. 1, (Spring 2008) 10-18

Influence of Process Maturity Level on Software Development Effort of Standard Size Projects

Ghulam Qadir Memon*
Professor and Associate Dean, Dadabhoy Institute of Higher Education, Karachi, Pakistan.

ABSTRACT

Software development effort can be greatly influenced by the process maturity level of the development organization. This paper investigates the influence of different process maturity levels on the development effort of standard size software projects. The software development effort is computed using COnstructive COst MOdel (COCOMO). The percent increase/decrease (change) in software development effort, productivity of development team and diseconomy of scale are used as the primary measures of effectiveness for this study. The results indicate that the influence of process improvement based on CMM maturity levels increases with the size of the software development projects.

INSPEC Classification : C6150G; C6110B; C6110S;

Keywords: Process Maturity, COCOMO, Scale Factors, Effort, Software Development, Standard Size Projects, Diseconomy of Scale, Productivity, Percent Change in Effort

1) INTRODUCTION

The software development effort is a direct indicator of development cost. This development effort is influenced by the software size, in addition to, various cost drivers and scale factors. The most common measure of software size is source lines of code (SLOC). In the original version of COnstructive COst MOdel (COCOMO) known as COMOCO 81, Boehm recommended the use of 15 cost drivers based on the product, personnel, platform, and project attributes (B. Boehm, 1981). This number was increased to 17 in COCOMO II for considering the impact of attributes such as software reuse and personnel continuity (B. Boehm, 2000). COCOMO 81 uses 3 modes of software development; namely, organic, semi-detached and embedded; to account for varying development conditions and degree of innovation required to develop the given product (B. Boehm, 1981, K. Branley, 2004, A. Jayaraman, 2007, B. Boehm, 1984).

* The material presented by the authors does not necessarily portray the viewpoint of the editors and the management of the Institute of Business and Technology (BIZTEK) or Dadabhoy Institute of Higher Education, Karachi, Pakistan.

*Ghulam Qadir Memon : gqmemon@yahoo.com

On the contrary, COCOMO II uses 5 scale factors to account for precedence of the development environment and project conditions, software development flexibility, architecture/risk resolution, cohesion of development team and process maturity based on the capability maturity model (CMM). The last scale factor is the focus of this study as it is directly based on the development process maturity level of an organization (B. Boehm 2000). Further details about this are provided in the sections to follow.

In this study, in order to estimate software development effort, COCOMO II is used. In doing so, all cost drivers/effort multipliers are assumed to be nominal. In addition, all scale factors except the one related to the process maturity (PMAT) are also assumed to be nominal. The scale factor PMAT is used to investigate the influence of different process maturity levels on software development effort for standard size projects. First of all, the development effort related to different process maturity levels is estimated in man-months (MM) or person-months (PM) using the standard project size and ratings of scale factors. This effort is then used to compute the productivity, diseconomy of scale and percent change (increase/decrease) in the effort, which are used as the primary measures of effectiveness (MOE) for investigating the desired influence.

2) ROCESS MATURITY LEVELS

Software process improvement (SPI) can be defined as an approach for designing and defining a new and improved software process to achieve basic business goals and objectives. During early stages of its inception, SPI was designed for improving quality and reliability of a process's product. Recently, it has been geared towards creating a new and improved software process in order to obtain some benefits. These benefits often include increased revenues or profits, decreased costs, and/or significant cost savings (D.F. RICO, 2004).

SPI is based on the Capability Maturity Model (CMM) of the Software Engineering Institute (SEI) at the Carnegie Mellon University (CMU), Pittsburgh, USA. CMM defines five levels of maturity for a given process, where each level signifies the level of performance that can be expected from an organization. For example, Maturity Level 1 organizations have ad hoc processes, whereas Maturity Level 2 organizations have a basic project management system in place, and so on (MARGARET K. KULPA, 2008). COCOMO II uses five scale factors as shown in Table 1.

Table 1.
COCOMO II Scale Factors

Scale Factor	**Abbreviation**
Precedentedness	PREC
Development Flexibility	FLEX
Architecture/Risk Resolution	RESL
Team Cohesion	TEAM
Process Maturity	PMAT

Out of these five scale factors four will be assumed nominal, while the remaining fifth (PMAT) rating will based on the development process maturity level, and hence, will vary according to the rating of process maturity level. In order to investigate the influence of process maturity level on the development effort, we will have to isolate the influence of PMAT from other factors because when many improvements are made concurrently, project managers have no way of determining how much improvement is due to process maturity versus other factors (Bradford K. Clark, 2000). CMM defines five levels of process maturity, whereas COCOMO II uses six ratings of PMAT varying from very low to extra high. In doing so, level 1 of CMM is divided into two halves – lower and upper as per details in Table 2 (B. Boehm, 2000).

Table 2.
PMAT Rating Based on Estimated Equivalence to CMM

COCOMzO II Rating	**CMM Level**	**Value**
Very Low	1 (Lower Half)	7.80
Low	1 (Upper Half)	6.24
Nominal	2	4.68
High	3	3.12
Very High	4	1.56
Extra High	5	0.00

In many surveys, majority of organizations evaluated according to SEI's SW-CMM Levels have been found at level 2. For example, in a survey of 161 organizations in 1998, 73 (45%) were evaluated at level 2. Table 3 gives the complete results (Bradford K. Clark, 2000). Since COCOMO II considers CMM level 2 as the nominal rating for PMAT rating, hence, percent increase/decrease in productivity, effort, etc., will be compared against the nominal ratings.

Table 3.
Survey Results of 161 Organizations

COCOMO II Rating	**CMM Level**	**No. of Organizations**	**%**
Very Low	1 (Lower Half)	6	4
Low	1 (Upper Half)	36	22
Nominal	2	73	45
High	3	23	14
Very High	4	17	11
Extra High	5	6	4
SUM		**161**	**100**

3) SIZE CLASSIFICATION

In order to investigate the influence of process maturity level on different software product sizes, it is desirable to classify the sizes into a reasonable scheme. According to (B. Boehm, 1981), the software products can be classified as small, intermediate, medium, large, and very large based on their sizes as per details given in Table 4. KLOC stands for kilo (thousand) lines of code.

Table 4.
Classification of Software Products Based on Size (B. Boehm, 1981).

Classification	**Size (KSLOC)**
Small (S)	2
Intermediate (I)	8
Medium (M)	32
Large (L)	128
Very Large (VL)	512

4) SOFTWARE EFFORT ESTIMATION

Software cost estimation refers to predicting the resources required for a software development process. It includes an approximate judgment of the costs for a project involving many variables. Often software cost estimation is measured in terms of effort. According to Johnson, the actual effort for individual tasks should be compared with estimated and planned values, enabling project managers to reallocate resources when necessary (K. Johnson, 1998). For estimating software development effort, the important issues include assumptions made, cost driver ratings, scale factor ratings, etc.

4.1) Assumptions

Generally, assumptions are made to conduct a research or study. For this study, the following assumptions have been made:

1. COCOMO II will be used for effort estimation.
2. Software projects of standard sizes from small to very large are used herein.
3. The ratings for all cost drivers are assumed to be nominal.
4. The ratings for all scale factors except PMAT are assumed to be nominal.
5. The percent increase/decrease in effort, productivity and diseconomy of scale is used as the primary measure of effectiveness.

4.2) Software Development Effort Estimation

There are many techniques and models for software effort estimation. However, due to openness and in-depth coverage, COCOMO/COCOMO II are most widely used. In this study, COCOMO II will be used for effort estimation (B. Boehm, 2000). The equation for nominal effort estimation is given by:

$$\mathbf{E_n = a\ S^e} \qquad \textbf{(1)}$$

Where $\mathbf{E_n}$ is the nominal effort and **a** is a constant that can be calibrated according to the local conditions. In the absence of the calibrated value, Boehm et al. suggest a value of 2.94 (B. Boehm, 2000), which is used herein. **S** stands for the size of the software product to be developed in kilo source lines of code (KSLOC). In Eq. 1, **e** represents the exponential constant to be determined from the ratings of a set of 5 scale factors (SF) that determine the economies/diseconomies of scale for the software product under development (B. Boehm, 2000, S. Chulani, 1999, Softstar Systems, 2005) These scale factors replace the development modes in original COCOMO, which always give diseconomies of scale. Mathematically, **e** is determined by the formula:

$$e = b + 0.01\sum_{j=1}^{5} SF_j \qquad \textbf{(2)}$$

In Eq. 2, **b** is a constant and a value of 0.91 is recommended by Boehm et al. (B. Boehm, 2000). The term ESF_j represents the summation of scale factors. Since nominal ratings are assumed for four of the scale factors in this study, the summation for different ratings of PMAT is given in Table 5 (B. Boehm, 20000).

Table 5.
Computation of Exponent e based on PMAT Rating

Scale Factor	**Rating of PMAT**					
	Very Low	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
PREC	3.72	3.72	3.72	3.72	3.72	3.72
FLEX	3.04	3.04	3.04	3.04	3.04	3.04
RESL	4.24	4.24	4.24	4.24	4.24	4.24
TEAM	3.29	3.29	3.29	3.29	3.29	3.29
PMAT	7.80	6.24	4.68	3.12	1.56	0.00
ESFj	**22.09**	**20.53**	**18.97**	**17.41**	**15.85**	**14.29**
e	**1.131**	**1.115**	**1.100**	**1.084**	**1.069**	**1.053**

For nominal rating of PMAT, substituting values in Eq. 2, we get:

$$\mathbf{e} = 0.91 + 0.01\ (18.97) = 1.1$$

For example, for a medium size project (32 KSLOC), putting the values in Eq. 1, we get:

$$\mathbf{E_n} = 2.94\ (32)^{1.1} = \mathbf{132.91\ MM}$$

Table 6 shows effort computation for standard size projects. This table can be used to compute productivity by using Eq. 3. In this equation, size is taken in lines of code (LOC), effort in MM and productivity in LOC/MM. Table 7 shows computed values of productivity for the standard size projects and Figure 1 graphically illustrates these values.

$$\text{Productivity} = \frac{\text{Size}}{\text{Effort}} \qquad \mathbf{(3)}$$

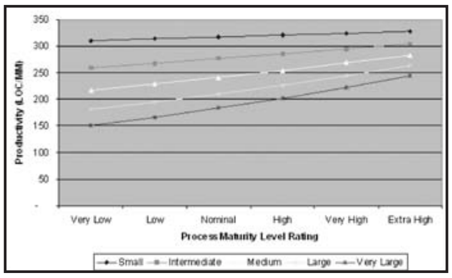
Table 6.
Effort Computation for Standard Size Projects

Project		**Effort (MM) based on PMAT Rating and Project Size**					
Classification	**Size (KLOC)**	**Very Low**	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
Small	2	6.44	6.37	6.30	6.23	6.17	6.10
Intermediate	8	30.88	29.89	28.94	28.01	27.12	26.25
Medium	32	148.09	140.29	132.91	125.92	119.29	113.01
Large	128	710.22	658.44	610.44	565.94	524.69	486.44
Very Large	512	3,406.13	3,090.27	2,803.70	2,543.71	2,307.82	2,093.81

Table 7.
Productivity Computation for Standard Size Projects

Project		**Productivity (LOC/MM) based on PMAT Rating and Project Size**					
Classification	**Size (KLOC)**	**Very Low**	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
Small	2	311	314	317	321	324	328
Intermediate	8	259	268	276	286	295	305
Medium	32	216	228	241	254	268	283
Large	128	180	194	210	226	244	263
Very Large	512	150	166	183	201	222	245

Figure 1.
Illustration of Productivity Rates for Standard Size Projects



Diseconomy of scale refers to relatively more increase in effort as compared to the increase in size of a software product. That is, if the size is doubled, the effort required will be more than double. To illustrate this, the standard sizes intermediate, medium, large and very large are divided by the small size (2 KLOC) and are called from small to intermediate (From S to I), from small to medium (From S to M), etc. Similarly, their corresponding efforts are also divided by the effort of small size to visualize the effect of diseconomy of scale as shown in Table 8 and Figure 2. In Figure 2, the size ratio has been plotted to visualize the impact of diseconomy of scale.

Table 8.
Diseconomy of Scale Computation for Standard Size Projects

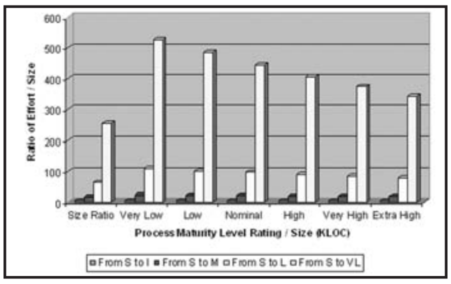
Project		**Productivity (LOC/MM) based on PMAT Rating and Project Size**					
Classification	**Size Ratio**	**Very Low**	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
From S to I	4	4.80	4.69	4.59	4.49	4.40	4.30
From S to M	16	23.00	22.03	21.09	20.20	19.35	18.53
From S to L	64	110.31	103.38	96.88	90.80	85.09	79.75
From S to VL	256	529.03	485.19	444.98	408.11	374.29	343.27

5) MEASURES OF EFFECTIVENESS (MOE)

To evaluate and compare results, certain criteria or measures of effectiveness are required. In this case, the percent increase/decrease (change) in software development effort, productivity rate and diseconomy of scale are used as the primary MOE. The objective is to investigate the influence of different process maturity levels on the software development effort for standard size projects. These percentages are computed by assuming the nominal rating of PMAT as the base case.. Mathematically, the percent increase/decrease (change) in any of three parameters (PC) is computed by using Eq. 4.

$$PC = \frac{\text{Value - Value}_{\text{Nominal}}}{\text{Value}_{\text{Nominal}}} \qquad \textbf{(4)}$$

Figure 2.
Illustration of Diseconomy of Scale for Standard Size Projects



In Eq. 4, Value represents the computed value of the concerned parameter at a given PMAT rating. Whereas ValueNominal stands for the value of the same parameter at nominal rating of PMAT. Negative value shows percent reduction and the positive one means percent increase in the parameter value. The percent change in effort, productivity and diseconomy of scale for standard size projects against different ratings of PMAT are shown in Figures 3 to 5.

Figure 3
Percent Change Computation of Effort for Standard Size Projects

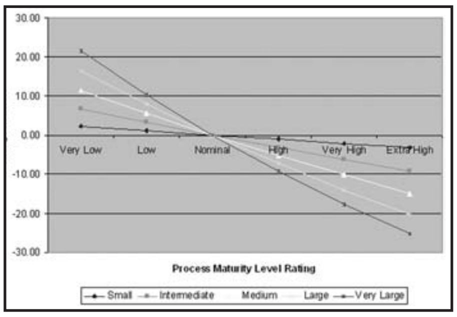


Figure 4
Percent Change Computation of Productivity for Standard Size Projects

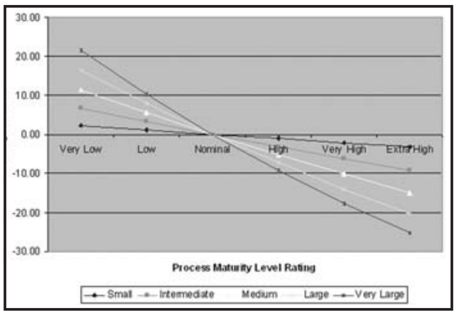
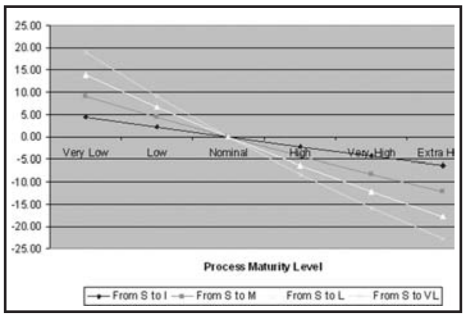


Figure 5
Percent Change Computation of Diseconomy of Scale for Standard Size Projects



6) EVALUATION AND INTERPRETATION OF RESULTS

The results are evaluated on the basis of the percent increase/decrease (change) in the software development effort, productivity and diseconomy of scale.. To make the process of evaluation and comparison of results more comprehensive, the average change in these parameters per one level change in PMAT rating is given in Tables 9 and 10.

Table 9.
Average % Change in Effort and Productivity Per Level Change in PMAT Rating

Project		Average % Change in	
Classification	Size (KLOC)	Effort	Productivity
Small	2	1.08	1.09
Intermediate	8	3.20	3.30
Medium	32	5.28	5.57
Large	128	7.33	7.91
Very Large	512	9.36	10.32

Table 10.
Average % Change in Diseconomy of Scale Per Level Change in PMAT Rating

Project		Average % Change
Classification	Size (KLOC)	
From S to I	4	2.14
From S to M	16	4.24
From S to L	64	6.31
From S to VL	256	8.35

The results indicate that as the PMAT rating improves, there is decrease in effort required, increase in productivity and reduction in diseconomy of scale. These changes are more significant for larger size projects as compared to smaller ones. For example, Figure 1 clearly indicates that the productivity changes more rapidly for larger projects as compared to smaller ones. Figure 2 shows that diseconomy of scale improves considerably with improvement in PMAT rating, which is quite desirable.

The percent change in required effort varies from increase of 2.19% for very low rating of PMAT to a decrease of 3.19% for a rating of extra high (a total change of 5.38%) for small projects. The corresponding figures for very large projects become 21.49%, -25.32% and 46.81%, respectively. Tables 9 and 10 show that the average change per level change in PMAT rating for three MOE varies from 1.08% for small projects to 10.32% for very large ones.

7) CONCLUSION

Results of this study indicate that the PMAT rating considerably influences the software development effort in terms of productivity and diseconomy of scale. This impact is more significant for larger projects as compared to smaller ones. In other words, the process improvement can lead to considerable reduction in resources required for software development efforts, especially, the projects of bigger size are affected more by the rating of PMAT. This signifies the need of process improvement for organizations handling large size projects making it a beneficial and worthwhile exercise for them.

REFERENCES

B. BOEHM (1981). Software Engineering Economics. Prentice-Hall, Englewood Cliffs, New Jersey, USA, October 1981.

B. BOEHM (2000), et al. Software Cost Estimation with COCOMO II. Prentice-Hall, Upper Saddle River, New Jersey, USA, 2000.

K. BRANLEY (2004). Software Estimation. March 2004.

http://home.iprimus.com.au/kbranley/Papers/Cost_Estimation.pdf

A. JAYARAMAN (2007). Characteristics that Make One Estimation Technique Better than Others. Institute of Software Research, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, May 2007.

B. BOEHM (1984). Software Engineering Economics. IEEE transactions on software engineering, 10(1): 4-21, January 1984.

D.F. RICO (2004). ROI of Software Process Improvement - Metrics for Project Managers and Software Engineers. J. Ross Publishing, Inc., USA, 2004.

MARGARET K. KULPA, KENT A. JOHNSON (2008). Interpreting the CMMI - A Process Improvement Approach. Auerbach Publications, New Jersey, USA, 2008.

Bradford K. Clark (2000). Quantifying the Effects of Process Improvement on Effort. IEEE Software, Vol. 17, No. 6, pp. 65-70, Nov/Dec, 2000.

K. JOHNSON (1998). Software cost estimation: metrics and models. Department of Computer Science, University of Calgary, Alberta, Canada, February 1998.

S. CHULANI (1999). Bayesian analysis of software cost and quality models. PhD dissertation, Faculty of the Graduate School, University of Southern California, USA, May 1999.

SOFTSTAR SYSTEMS (2005). Overview of COCOMO. Revised April 18, 2005. http://www.softstarsystems.com/overview.htl