Empirical validation of Software development effort multipliers of Intermediate COCOMO Model

Basavaraj M.J
Perot Systems, EPIP Phase II,
Whitefield Industrial Area, Bangalore-560 066, India
Email : basavaraj.m@ps.net; basavarajmj@hotmail.com

Dr. K.C Shet
Professor, Computer Department, National Institute of Technology Karnataka, Surathkal, India
Email : kcshet@nitk.ac.in; kcshet@yahoo.co.uk

Abstract—**Intermediate COCOMO Model computes effort as a function of program size and a set of cost drivers. Effort adjustment factor (EAF) is calculated using 15 cost drivers. EAF is an important significant factor in computing software development effort. We have taken one delivered development project of size of 479 function points and planned for 917 Person days of SEI CMM Level 5 "Excellent" Company as a case study to analyze the EAF. We have empirically validated the cost driver model for Intermediate COCOMMO using this projects data. Validation has been done by using other two development projects data of Excellent Company. From our analysis, we have found that cost drivers defined ratings need to be revisited for the projects of size less than 10 Person months. We have come out with ratings for some cost drivers where earlier it was not defined. This approach helps the project managers to anticipate and estimate the efforts for development projects preferably less than 10 Person months. We have achieved approximately 30% improvement in effort variance by following this approach.**

Keywords—**Software Estimation, Intermediate COCOMO Model**

I. INTRODUCTION

In the intention of maintaining security and confidentiality of data, authors are constrained not to disclose the company or client name or project name or exact named data in their research. In this context Company name "Excellent", Project name "A", Project name :"B", Project name "C" and client name "Super" refer some dummy names. Authors intended to use the past data of SEI CMM Level5 matured company "Excellent" of Project "A", Project "B" and Project "C" to empirically validate the Intermediate COCOMO Model by predicting cost drivers.

A. Literature Review

We have undertaken literature review to study work done till now by others in this context. Improving software effort estimation does not necessarily require adopting sophisticated formal estimation models or expensive project experience databases[2]. Jorgensen argues that estimation using expert judgements are better than models[2]. Fran Niessink and Hans van Vliet[3] clams that existence of a consistently applied process is an important and a prerequisite for a successful measurement program in case of different environments.
There are number of ways to determine the effort needed in software development projects. In traditional software cost models, costs are derived from effort. Empirical estimation models provide computational formulae for calculating the effort based on statistical approach by referring the past data of more or less similar projects executed[9][10]
COCOMO (Boehm, 1981) is the one of the best of these models. Boehm states that COCOMO's intermediate model gives estimates which varies from the actual needed effort about 20% in average. COCOMO-II (Boehm et al., 2000) is a new updated version of the model, with a more modern project database[9][10].
The Intermediate COCOMO model computes effort as a function of program size and a set of cost drivers[4]. Usage of cost driver is significant from the point of view of Project Managers while estimating projects which are of less size in person months. Besides, such studies are sparsely available in literature.

B. Scope of this Work

This paper explains the Empirical validation for software development effort multipliers of Intermediate COCOMO model and analysis has been done to define the ratings for some cost drivers of EAF.
Advantages are listed below.

(i) This approach can be used to estimate development projects which were having the projects of size less than 10 Person Month(PM).
(ii) Anticipating appropriate EAF contributes in achieving minimal effort variance
(iii) Productivity of the project can also be improved by predicting cost drivers properly by following this approach.

Software community can be benefited by adopting this methodology in their development project for achieving minimal effort variance by predicting cost drivers for computing EAF.

II. METHODLOGY OF THE WORK

A COCOMO Model

The Basic COCOMO model computes effort as a function of program size[4]. The Basic COCOMO equation is:

$$E = aKLOC \wedge b$$

Where E is the nominal efforts in Person Months, a and b are the constants.

Values of a and b for the Basic COCOMO model are shown in below Table 1[5].

Table 1
Effort for three modes of Basic COCOMO

Mode	a	b
Organic	2.4	1.5
Semi-detached	3.0	1.12
Embedded	3.6	1.20

The Intermediate COCOMO model computes effort as a function of program size and a set of cost drivers [4]. The Intermediate COCOMO equation is given by:

$$E = aKLOC \wedge b * EAF$$

The Values of "a" and "b" for the Intermediate COCOMO model are shown in below Table 2[5].

Table 2
Effort parameters for three modes of Intermediate COCOMO Model

Mode	a	b
Organic	3.2	1.05
Semi-detached	3.0	1.12
Embedded	2.8	1.20

B. Effort Adjustment factor

The effort adjustment factor has been calculated using 15 cost drivers. Cost drivers are grouped into four categories:

(i) Product
(ii) Computer
(iii) Personnel
(iv) Project

Each cost driver has been rated on a six-point ordinal scale ranging from low to high importance. Based on the rating, an effort multiplier is determined using Table 3[5]. Product of all effort multipliers leads to EAF.

Table 3
Software Development Effort Multipliers

Cost Driver	Description	Rating					
		Very Low	Low	Nominal	High	Very High	Extra High
Product							
RELY	Required software reliability	0.75	0.88	1	1.15	1.4	-
DATA	Database size	-	0.94	1	1.08	1.16	-
CPLX	Product complexity	0.7	0.85	1	1.15	1.3	1.65
Computer							
TIME	Execution time constraint	-	-	1	1.11	1.3	1.66
STOR	Main storage constraint	-	-	1	1.06	1.21	1.56
VIRT	Virtual machine volatility	-	0.87	1	1.15	1.3	-
TURN	Computer turnaround time	-	0.87	1	1.07	1.15	-
Personnel							
ACAP	Analyst capability	1.46	1.19	1	0.86	0.71	-
AEXP	Applications experience	1.29	1.13	1	0.91	0.82	-
PCAP	Programmer capability	1.42	1.17	1	0.86	0.7	-
VEXP	Virtual machine experience	1.21	1.1	1	0.9	-	-
LEXP	Language experience	1.14	1.07	1	0.95	-	-
Project							
MODP	Modern programming practices	1.24	1.1	1	0.91	0.82	-
TOOL	Software Tools	1.24	1.1	1	0.91	0.83	-
SCED	Development Schedule	1.23	1.08	1	1.04	1.1	-

III. WORK DONE

A. Case Study – Project 'A'

We have taken data from development "A" for function point analysis and empirical validation purpose. Project "A" is development project which follows the Software development Life Cycle Methodology(SDLC) for Delivery Execution Model. Project "B" and Project "C" taken for Validation purpose in next subsequent sections also belong to same nature of model.

Table 4
Function Point Count Sheet

Client : Project Name : Name of the Unit Process : FP	Module Type (E/N) : E (E: Existing, to be enhanced; N: New)

Sl. No.	Function Type	Class			Function Point
		Simple	Average	Complex	
1	External Inputs	19	2	5	95
2	External Outputs	5	0	0	20
3	Logical Internal Files	20	5	0	190
4	External Interface Files	14	0	0	70
5	External Inquiries	24	10	2	124
6	Function Count	Total Degree of Influence			499

General Information Processing Function

C1	Data Communications	3	C8	Online Update	2
C2	Distributed Functions	0	C9	Complex Processing	2
C3	Performance	4	C10	Reusability	1
C4	Heavily Used Configuration	3	C11	Installation Ease	2
C5	Transaction Rate	2	C12	Operational Ease	3
C6	Online Data Entry	4	C13	Multiple Site	0
C7	End User Efficiency	3	C14	Facilitate Change	2
GC	General Characteristics	31			

Degree of Influence Values :

Not present, or no influence	0	GCA General Characteristics Adjustment :	0.96
Insignificant influence	1	FP Function Points Measure :	479.04
Moderate Influence	2		
Average Influence	3		
Siginificant Influence	4		
Strong influence, throughout	5		

Table 4 shows how the function point count has been arrived for Project "A". Readers are assumed to be aware of how to calculate the function point count for the development project. Otherwise readers are suggested to get acquaint with the same by reading the paper[6] with same authors. In brief Function Point(FP) counting procedure has been explained below[6].

B. Function Point Counting Procedure – Project 'A'

FP Counting is identification of boundary, in this case complete Project "A". Counting boundary includes Data Function Types count, Transactional Function Types Count which gives Unadjusted Function FP Count[UAF][6][8] Also Counting Boundary includes Value Adjustment factor[VAF] which is determined by General Systems characteristics(GSC)[6][8].
Data Function type count is identification of Internal logical files(ILF) and External Interface files(ILF). Transactional Function Type count is identification of External Inputs(EI), External Outputs(EO), and External Inquiries(EQ).
Internal Logical File(ILF or Logical Internal File) is a user identifiable group of logically related data or control information maintained within the application boundary.
External Interface File(EIF) is a user identifiable group of logically related data or control information referenced by the application, but maintained within the boundary of another application.
External Input(EI) is an elementary process that processes data or control information that comes from outside the application boundary.
External Output (EO) is an elementary process that sends data or control information outside the application boundary.
External Inquiry (EQ) is an elementary process that sends data or control information outside the application boundary.
There are 14 points considered to come out with VAF (Value Added factor).

1. Data communications
2. Distributed data processing:
3. Performance
4. Heavily used configuration
5. Transaction rate
6. On-Line data entry
7. End-user efficiency
8. On-Line update
9. Complex processing
10. Reusability
11. Installation ease
12. Operational ease
13. Multiple sites
14. Facilitate change

All the GSC has ratings from 0 to 5[6][8].
Degrees of Influence are defined as below.

0 Not present, or no influence
1 Incidental or insignificant influence
2 Moderate influence
3 Average influence
4 Significant influence
5 Strong influence throughout

Variable adjustment factor[VAF] is calculated by below formulae[6][8].

VAF = (TDI * 0.01) + 0.65
Final FP count is given by using below formulae[8].

Final FP = UFP * VAF

479 is the Function Point that we have calculated for this project using the procedure explained above.
Below Table 5 shows activity-wise project effort distribution for project "A". Efforts have been distributed for the activities or Tasks by referring the past history data of the similar projects explained. Planned Efforts is given in terms of % and Person Days(PD).

Table 5
Activity-wise Project Efforts distribution

Client :			
Project Name :			
Estimated Size :	479 FP (98 Pgms)		
Estimated Productivity :	0.52		
Total Planned Efforts :	917 (Person days)		
S. No.	Tasks	Planned Efforts	
		% of total	Person-Days
1	Analysis Phase*	3%	25.676
2	Design Phase*	9%	82.53
3	Construction Phase*	39%	353.045
4	Testing* (ST,UAT & Warranty)	27%	243.922
5	Project Planning	3%	30.261
6	Project Tracking	3%	33.427
7	Software Quality Assurance	1%	9.17
8	Configuration Management	3%	27.51
9	Project Documentation	3%	27.51
10	Reviews	8%	70.609
11	Training	1%	9.17
12	Inter-group coordination	1%	4.585
	TOTAL	100%	917.415
* The phases can vary depending on the type of project.			
Remarks :	Construction includes coding and unit testing.Training is only for the project(application and VSS).The contingency is distributed across the phases		

Fig 1 shows graphical representation using Pie chart for project efforts distribution.

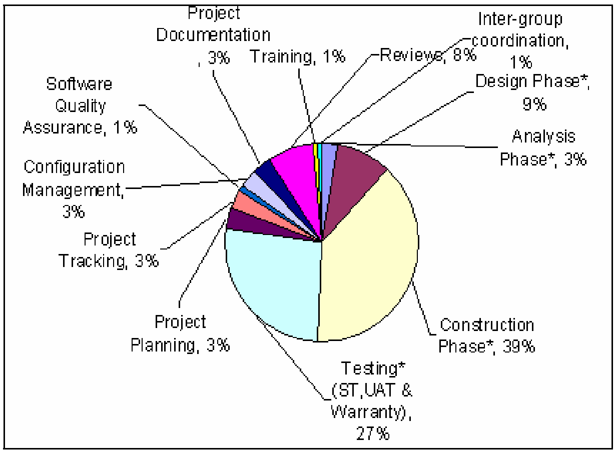

Figure 1. Pie Chart-Project Efforts Distribution

Below Table 6 shows effort variance with respect to each task of delivered project "A" for planned efforts Vs Actual Efforts.

Table 6
Project Estimate – Planned & Actual

Sr. No	Tasks	% Total	Planned(PD)	Actual(PD)
1	Analysis Phase*	3%	25.676	23
2	Design Phase*	9%	82.53	60
3	Construction Phase*	39%	353.045	392
4	Testing* (ST,UAT & Warranty)	27%	243.922	244
5	Project Planning	3%	30.261	29
6	Project Tracking	3%	33.427	32
7	Software Quality Assurance	1%	9.17	13
8	Configuration Management	3%	27.51	23
9	Project Documentation	3%	27.51	19
10	Reviews	8%	70.609	72
11	Training	1%	9.17	4
12	Inter-group coordination	1%	4.585	
	TOTAL	100%	917.415	911

C. Software Estimation and Analysis Tool

We have used Software Estimation and Analysis tool for validating purpose[7][8]. The goal is to use the tool for calculating efforts in PM by applying Intermediate COCOMO Model. Also, it is to find out the limitation of the Model in estimating development projects of size greater than 10 PM.
Below Fig 2 shows how the FP has been calculated for the Project "A".

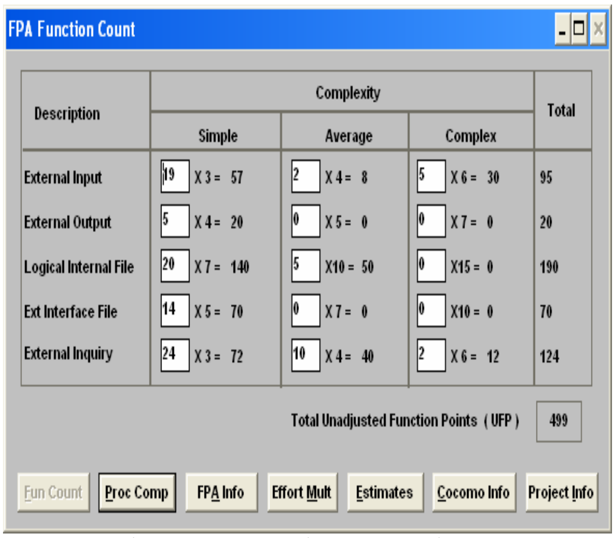

Figure 2. FPA Function Count –Project "A"

Values for EI, EO, ILF,EIF, EQ have been calculated for Project "A" using the procedure given in above section B. Multipliers for EI is given by 3 for Simple, 4 Average and 6 for Complex. Multiplier for EO is given by 4 for Simple, 5 for Average and 7 for Complex. Multiplier for ILF is given by 7 for Simple, 10 for Average and 15 for Complex. Multiplier for EIF is given by 5 for Simple, 7 for Average and 10 for Complex. Multiplier for EQ is given by 3 for Simple, 4 for Average and 6 for Complex[6][8].
Total Unadjusted Function Points(UFP) count is 499 is summation of count of EI,EO,ILF,EIF and EQ as shown in Fig 2.

Total Degree of Influence is nothing but count of General Characteristics is shown in below Fig 3. This is calculated by using the procedure explained in Section B above.

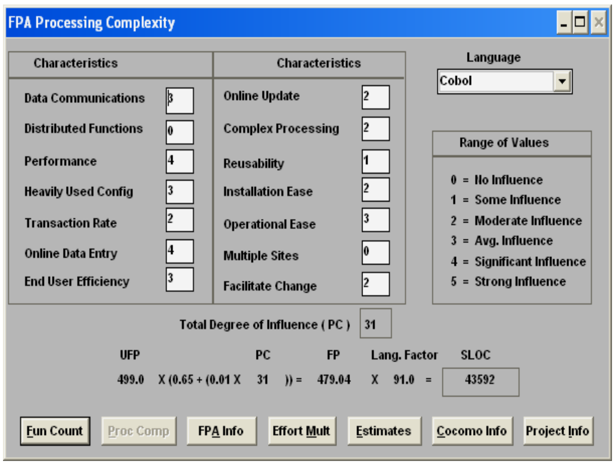

Figure 3. FPA Processing Complexities-Project "A"

Project "A' used Cobol language, so we have taken multiplication language factor as 91 by referring the data of similar past executed projects from Organization Software Process Database (SPDB)
Selecting Nominal option for each rating by referring Table 3, we have got EAF(Effort Adjustment Factor or Effort Fcator) as "1".
Nominal PM(person Month) from the tool we have got as 205.70
Effort Exponent is 1.12 from Table 2, since Project belongs to Semi-detached Mode[5].
KSLOC(Kilo Source Lines of Code) from Fig 3 we got 43.59

Nominal PM = Effort Factor * KSLOC ^ Effort Exponent

Nominal PM = 1 * 43.59 ^ 1.12 = 205.70 PM

By referring Project "A" data from Table 5, it is Total Planned efforts 917 Person days, in terms of PM, it is 5.73 PM which is less than 10 PM.

D. Analysis

Using the available ratings options from each cost driver, we cannot estimate the projects of these types of sizes from Intermediate COCOMO Model.
By analyzing each cost driver it can be inferred that increasing of product & computer attributes increases Total effort Multiplier, which in turn increases the development efforts. Decreasing of Personnel & Project Attributes decreases the development effort.
By selecting the minimal ratings for product & computer attributes and maximum ratings for Personnel & Project Attributes, we have got Effort Multiplier as 0.10 as shown in below Fig 4..

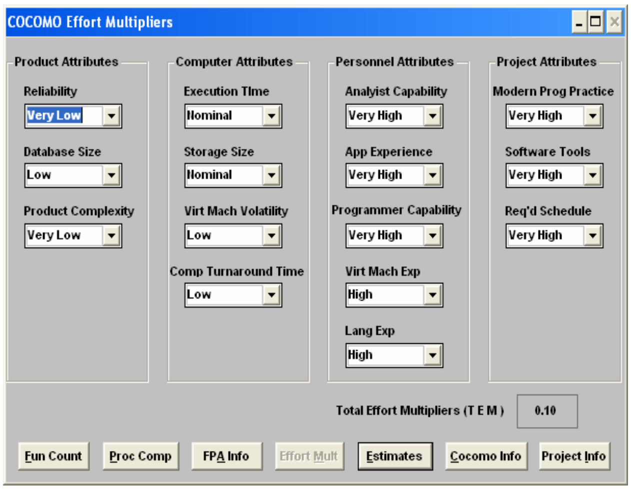

Figure 4. Ratings Combination-Maximum extent

As shown in Fig 4, by selecting Cost Driver we got rating by referring Table 3 as below.

- Reliability -(Very Low)-“0.75”
- Database Size-(Low)-“-“
- Product Complexity-(Very Low)-“0.7”
- Execution Time-(Nominal)-“1”
- Storage Size-(Nominal)-“1”
- Virtual(Virt) Machine volatility-(Low)-0.87“
- Comp Turn around Time-(Low)-“0.87”
- Analyst capability-(Very High)-“0.71”
- Applications experience-(Very High)-“0.82”
- Programmer capability-(Very High)-“0.7”
- Virtual machine experience-(High)-“0.9“
- Language experience-(High)-“0.95”
- Modern programming practices-(Very High)-“0.82”
- Software Tools-(Very High)-“0.83”
- Development Schedule-(Very High)-“1.1”

We got Total Effort Multiplier as 0.10 as shown in Fig 4 by multiplying all ratings for selected Cost Driver which is mentioned above.

In the same project we find that the PM is 20.57 which is very much higher in comparison to 5.73 PM as shown in Fig 5. Thus the improvement is 30% approximately

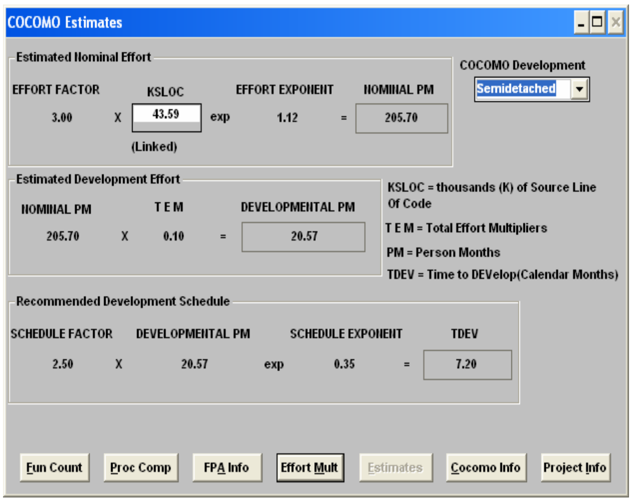

Figure 5. Minimum Person Months

We have obtained Developmental PM for Project A as Nominal PM * TEM

Developmental Person Month = 205.70 PM*0.10= 20 PM.

IV. IMPROVEMENTS ACHIEVED

We have introduced the minimal rating “0.7” to the drivers DATA, TIME, STOR VIRT and TURN as shown in below Table 9. The rating “0.7” is a minimum among all existing cost driver ratings of Intermediate COCOMO Model. By adding the new rating we have not altered the existing characteristic behavior of Intermediate COCOMO Model, but we have tailored to represent estimation for development projects of Size less than 20 PM approximately. We have achieved 6.81 PM which is very much nearer to 5.73 PM compared to 20.57 Person Month. This leads in achieving of 30% approximately improvement in effort variance by implementing this new approach.

Table 7
Introduced Ratings for Cost drivers for Project “A”

Project A								
Cost Driver Category	Cost Driver	Description	Rating					
			Very Low	Low	Nominal	High	Very High	Extra High
Product	RELY	Required software reliability	-	0.88	-	-	-	-
	DATA	Database size	0.7	-	-	-	-	-
	CPLX	Product complexity	-	0.85	-	-	-	-
Computer	TIME	Execution time constraint	0.7	-	-	-	-	-
	STOR	Main storage constraint	0.7	-	-	-	-	-
	VIRT	Virtual machine volatility	0.7	-	-	-	-	-
	TURN	Computer turnaround time	0.7	-	-	-	-	-
Personnel	ACAP	Analyst capability	-	-	-	-	0.71	-
	AEXP	Applications experience	-	-	-	-	0.82	-
	PCAP	Programmer capability	-	-	-	-	0.7	-
	VEXP	Virtual machine experience	-	-	1	-	-	-
	LEXP	Language experience	-	-	-	0.95	-	-
Project	MODP	Modern programming practices	-	-	-	-	0.82	-
	TOOL	Software Tools	-	-	-	-	0.83	-
	SCED	Development Schedule	-	-	1	-	-	-

Total Effort Multiplier[TEM] is the product of ratings of all Cost Driver Categories. From Table 7 we have arrived 0.033118 for TEM. Nominal Person month we get 205.70 as discussed in earlier section.

Developmental PM = Nominal PM * TEM

Developmental PM = 205.70 PM * 0.03118 = 6.81 PM

V. INTERMEDIATE COCOMO – EAFS VALIDATION

Proposed Methodology has been implemented for two projects namely Project “B” of Estimated Size 405 FP and Planned Efforts 698 PDs and for Project “C” of Estimated Size 503 FP and Planned Efforts 824 PDs. Both these projects size is less than 10 PM. We have introduced the same new ratings for these two projects also. 171.2.PM is a Nominal PM reflected as per the procedure explained.

From below Table 8 we have obtained 0.034861 TEM.

Table 8
Ratings for Cost drivers for Project "B"

Project B								
Cost Driver Category	Cost Driver	Description	Rating					
			Very Low	Low	Nominal	High	Very High	Extra High
Product	RELY	Required software reliability	-	0.88	-	-	-	-
	DATA	Database size	0.7	-	-	-	-	-
	CPLX	Product complexity	-	0.85	-	-	-	-
Computer	TIME	Execution time constraint	0.7	-	-	-	-	-
	STOR	Main storage constraint	0.7	-	-	-	-	-
	VIRT	Virtual machine volatility	0.7	-	-	-	-	-
	TURN	Computer turnaround time	0.7	-	-	-	-	-
Personnel	ACAP	Analyst capability	-	-	-	-	0.71	-
	AEXP	Applications experience	-	-	-	-	0.82	-
	PCAP	Programmer capability	-	-	-	-	0.7	-
	VEXP	Virtual machine experience	-	-	1	-	-	-
	LEXP	Language experience	-	-	1	-	-	-
Project	MODP	Modern programming practices	-	-	-	-	0.82	-
	TOOL	Software Tools	-	-	-	-	0.83	-
	SCED	Development Schedule	-	-	1	-	-	-

We have obtained Developmental PM for Project B as
171.2 PM * 0.034861 = 5.96 PM
We have obtained Developmental PM for Project "C" as 214.1 PM as Nominal PM, in the similar way.

From below Table 9 we have obtained 0.036311 as TEM.

Table 9
Ratings for Cost drivers for Project "B"

Project C								
Cost Driver Category	Cost Driver	Description	Rating					
			Very Low	Low	Nominal	High	Very High	Extra High
Product	RELY	Required software reliability	-	0.88	-	-	-	-
	DATA	Database size	0.7	-	-	-	-	-
	CPLX	Product complexity	-	0.85	-	-	-	-
Computer	TIME	Execution time constraint	0.7	-	-	-	-	-
	STOR	Main storage constraint	0.7	-	-	-	-	-
	VIRT	Virtual machine volatility	0.7	-	-	-	-	-
	TURN	Computer turnaround time	0.7	-	-	-	-	-
Personnel	ACAP	Analyst capability	-	-	-	-	0.71	-
	AEXP	Applications experience	-	-	-	-	0.82	-
	PCAP	Programmer capability	-	-	-	-	0.7	-
	VEXP	Virtual machine experience	-	-	1	-	-	-
	LEXP	Language experience	-	-	-	0.95	-	-
Project	MODP	Modern programming practices	-	-	-	-	0.82	-
	TOOL	Software Tools	-	-	-	0.91	-	-
	SCED	Development Schedule	-	-	1	-	-	-

Further we have obtained Developmental PM for Project C as
214.1 PM * 0.036311 = 7.71 PM
Below Table 10 shows Intermediate COCOMO – EAFs validation.

Table 10
Intermediate COCOMO – EAFs validation

Project A - Planned Efforts(PM) - 5.73 PM						
Intermediate COCOMMO Model representing 5.73 PM for Project A is having limitation with existing EAFs(Effort Adjustment Factors)						
Validated the same using Software and Estimation and Analysis tool found that maximum 20.57 PM can be represented using Intermediate COCOMO Model						
Proposed New ratings for some EAFs						
With proposed ratings Achieved to represent 6.81 PM						
	Estimated Size	Total Planned Efforts(PD)	Estimated Productivity	Total Planned Efforts(PM	With Proposed	Without Proposed
Project A	479 FP	917 PD	0.52	5.73 PM	6.81 PM	20.57 PM
Implementation the proposed Methodology Project B & Project C						
	Estimated Size	Total Planned Efforts(PD)	Estimated Productivity	Total Planned Efforts(PM	With Proposed	Without Proposed
Project B	405 FP	698 PD	0.58	4.36 PM	5.96 PM	20.57 PM
Project C	503 FP	824 PD	0.61	5.15 PM	7.71 PM	20.57 PM
Improvement is 30% Effort variance						

From Table 10, we infer that, we can use Intermediate COCOMO Model by implementing the proposed approach which results in measuring development projects of size less than 20 PM. Table 10 also shows that we have achieved 30% Effort variance while measuring using the proposed methodology.

VI. Conclusion

Predicting cost drivers for computing EAF is a significant factor. This helps in Project Managers to anticipate appropriate action to achieve minimal effort variance. This approach is useful in estimating development projects which were having the projects of size less than 10 person months. We have achieved approximately 30% improvement in effort variance by following this approach. Productivity of the project can also be improved by predicting cost drivers properly by following this approach.

REFERENCES

[1] Capers Jones, "Estimating Software Costs", Tata McGraw Hill, Edition 2005

[2] Magne Jorgensen, "Practical Guidelines for Expert-Judgment-Based Software Effort Estimation", IEEE Software, Published by the IEEE Computer Society,2005

[3] Frank Niessink and Hans van Vliet, " Two Case Studies in Measuring Software Maintenance Effort", Published in the proceedings of the International Conference on Software Maintenance, Bethesda, Maryland, USA, November-16-20, 1998,pp.76-85.

[4] Pressman, R.S. (1997). Software Engineering A Practitioner's Approach. McGraw-Hill.

[5] Boehm, B.W. (1981). Software Engineering Economics. Prentice Hall.

[6] Basavaraj M.J and Dr. K.C Shet," Software Estimation using Function Point Analysis Difficulties and Research Challenges", Proc. on International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering(CISSE-06)., Dec 2006

[7] Software Analysis and Estimation Tool (SEAT) version 2.4.2 Manual, East Tennessee State university (http://www.laatuk.com/tools/effort_estimation_tools.html, http://www.cs.umt.edu/RTSL/dsstud/dsstudio.htm)

[8] Function Points Counting Practices Manual, Release 4.1.1, International Function Point User Group(IFPUG),Oct 2004

[9] Boehm, B., Horowitz, E., Madachy, R., Reifer, D., Clark, B., Steece, B., Brown,A.W., Chulani, S. & Abts, C, "Software Cost Estimation with COCOMO II". Prentice Hall, 502 p., 2000.

[10] Boehm, B. & Papaccio, P. 1988. "Understanding and controlling software costs".IEEE Transactions on Software Engineering 14 (10), 1462-1477.,1988

Basavaraj M.J, Project Manager in Perot Systems, Bangalore, has an industry experience of more than 14 Years in Project Management, Software Estimation and Software Quality. He has done his B.E in Computer Science and Engineering from Bapuji Institute of Engineering & Technology, Karnataka and M.Tech in Computer Science and Engineering from Karnataka Regional Engineering college, Surathkal, Karnataka. He is currently undergoing his Part-time PhD in the area of Software Estimation from National Institute of Technology, Surathkal, Karnataka. He is associated with MISTE and is a certified Software Test Manager and certified Software Quality Analyst. He has published several papers under International Conference Proceedings/International Journals.

Dr. K.C Shet, Professor in Computer Engineering Department, National Institute of Technology, Karnataka has more than 35 years of experience in Teaching, Research and Development. He has done his B.E in Electronics & Communication from University of Mysore, M.SC Engineering from University of Sambalpur and his PhD from IIT Bombay. He is as Member of Computer Society of India, Mumbai and Indian Society of Technical Education, New Delhi. His Research work specialized areas are software testing, Security Solution for Web Services, Cyber Laws, Anti spam solutions, Wireless Networks, Mobile Computing, Ad hoc Networks. He has published more than 185 papers.