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: kcsheet@nitk.ac.in; kcsheet@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 COCOMO 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.
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. METHODOLOGY OF THE WORK

A. COCOMO Model

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

\[ E = aKLOC^b \]

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>
<thead>
<tr>
<th>Mode</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organic</td>
<td>2.4</td>
<td>1.5</td>
</tr>
<tr>
<td>Semi-detached</td>
<td>3.0</td>
<td>1.12</td>
</tr>
<tr>
<td>Embedded</td>
<td>3.6</td>
<td>1.20</td>
</tr>
</tbody>
</table>

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:

\[ E = aKLOC^b \times EAF \]

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

<table>
<thead>
<tr>
<th>Mode</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organic</td>
<td>3.2</td>
<td>1.05</td>
</tr>
<tr>
<td>Semi-detached</td>
<td>3.0</td>
<td>1.12</td>
</tr>
<tr>
<td>Embedded</td>
<td>2.8</td>
<td>1.20</td>
</tr>
</tbody>
</table>

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.

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.

© 2008 ACADEMY PUBLISHER
velopment project. Otherwise readers are suggested to get acquainted 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(EIF). 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 formula[6][8].

\[ VAF = (TDI * 0.01) + 0.65 \]

Final FP count is given by using below formulae[8].

\[ \text{Final FP} = \text{UFP} \times \text{VAF} \]

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>
<thead>
<tr>
<th>No.</th>
<th>Tasks</th>
<th>Estimated Size</th>
<th>Estimated Productivity</th>
<th>Total/Planned Efforts</th>
<th>% of total</th>
<th>Person Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Analysis Phase</td>
<td>479 FP</td>
<td>0.52</td>
<td>817</td>
<td>100%</td>
<td>256.415</td>
</tr>
</tbody>
</table>

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].

\[ \text{Final FP} = \text{UFP} \times \text{VAF} \]

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>
<thead>
<tr>
<th>% No.</th>
<th>Tasks</th>
<th>% of total</th>
<th>Person Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Analysis Phase</td>
<td>3%</td>
<td>256.415</td>
</tr>
<tr>
<td>2</td>
<td>Design Phase</td>
<td>9%</td>
<td>28.75</td>
</tr>
<tr>
<td>3</td>
<td>Construction Phase</td>
<td>39%</td>
<td>259.405</td>
</tr>
<tr>
<td>4</td>
<td>Testing Phase (UT &amp; WT)</td>
<td>27%</td>
<td>244.922</td>
</tr>
<tr>
<td>5</td>
<td>Project Planning</td>
<td>3%</td>
<td>25.621</td>
</tr>
<tr>
<td>6</td>
<td>Project Tracking</td>
<td>3%</td>
<td>25.621</td>
</tr>
<tr>
<td>7</td>
<td>Software Quality</td>
<td>1%</td>
<td>8.17</td>
</tr>
<tr>
<td>8</td>
<td>Configuration Management</td>
<td>3%</td>
<td>27.51</td>
</tr>
<tr>
<td>9</td>
<td>Project Documentation</td>
<td>3%</td>
<td>27.51</td>
</tr>
<tr>
<td>10</td>
<td>Reviews</td>
<td>6%</td>
<td>70.600</td>
</tr>
<tr>
<td>11</td>
<td>Training</td>
<td>1%</td>
<td>1.17</td>
</tr>
<tr>
<td>12</td>
<td>Inter group coordination</td>
<td>1%</td>
<td>4.596</td>
</tr>
</tbody>
</table>

TOTAL 100% 917.415

* The phases may 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.

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

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.

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.
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”
- Programmer capability-(Very High)-“0.7”
- Applications experience-(Very High)-“0.82”
- 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
From below Table 8 we have obtained 0.034861 TEM.

<table>
<thead>
<tr>
<th>Cost Driver Category</th>
<th>Cost Driver Description</th>
<th>Project B</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>Required software reliability</td>
<td>Very Low</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Database size</td>
<td>Low</td>
<td>0.06</td>
</tr>
<tr>
<td></td>
<td>Development size</td>
<td>Normal High</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>Experience</td>
<td>High</td>
<td>0.90</td>
</tr>
<tr>
<td></td>
<td>Product complexity</td>
<td>Extra High</td>
<td>0.95</td>
</tr>
<tr>
<td>Computer</td>
<td>Execution time constraint</td>
<td>Very Low</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Memory management</td>
<td>Low</td>
<td>0.06</td>
</tr>
<tr>
<td></td>
<td>Virtual machine</td>
<td>Normal High</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>Computer turnaround time</td>
<td>High</td>
<td>0.90</td>
</tr>
<tr>
<td>Personnel</td>
<td>Analyst capability</td>
<td>Very Low</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Application experience</td>
<td>Low</td>
<td>0.06</td>
</tr>
<tr>
<td></td>
<td>Virtual machine</td>
<td>Normal High</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>Language experience</td>
<td>High</td>
<td>0.90</td>
</tr>
</tbody>
</table>

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>
<thead>
<tr>
<th>Cost Driver Category</th>
<th>Cost Driver Description</th>
<th>Project B</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>Required software reliability</td>
<td>Very Low</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Database size</td>
<td>Low</td>
<td>0.06</td>
</tr>
<tr>
<td></td>
<td>Product complexity</td>
<td>Normal High</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>Experience</td>
<td>High</td>
<td>0.90</td>
</tr>
<tr>
<td>Computer</td>
<td>Execution time constraint</td>
<td>Very Low</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Memory management</td>
<td>Low</td>
<td>0.06</td>
</tr>
<tr>
<td></td>
<td>Virtual machine</td>
<td>Normal High</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>Computer turnaround time</td>
<td>High</td>
<td>0.90</td>
</tr>
<tr>
<td>Personnel</td>
<td>Analyst capability</td>
<td>Very Low</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Application experience</td>
<td>Low</td>
<td>0.06</td>
</tr>
<tr>
<td></td>
<td>Virtual machine</td>
<td>Normal High</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>Language experience</td>
<td>High</td>
<td>0.90</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Intermediate COCOMO – EAFs validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project A</td>
</tr>
<tr>
<td>Project B</td>
</tr>
<tr>
<td>Project C</td>
</tr>
</tbody>
</table>

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


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.

© 2008 ACADEMY PUBLISHER