ISSN No. 0976-5697

Volume 3, No. 5, Sept-Oct 2012

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Estimating Software Maintenance Cost by Varying Maintainability Metric

C.V.S.R Syavasya
Department of Computer Science and Engineering,
Gitam University, Visakhapatnam,
Andhra Pradesh 530045, India
syavasyacvsr1@gmail.com

Abstract: The main aim of software maintenance is keeping the availability of test systems. In this paper, we estimate software maintenance costs by applying release level model to obtain values from previous projects and find out software maintenance costs of current project. We come across three parameters namely: understandability, modifiability, testability. Maintainability metric is varied under a specified range to find maintenance costs. After deriving the final metric values, we derive the final costs of maintainability. Finally the results are formulated into three cases, where in each case provides how maintenance cost is varied by varying maintainability metric.

Keywords: Understandability, Modifiability, Testability, Annual Change Traffic, SLOC

I. INTRODUCTION

Software maintenance is the process of modifying, for update or repair, existing operational software, but leaving its primary functions intact. This definition excludes major enhancements and hence differs from Swanson's typology. Over the years, several software maintenance models have been proposed, to emphasize particular aspects of software maintenance. Boehm's maintenance model consists of three major phases: understanding, modifying and validating the software (testing). Lewis and Henry (1989) have addressed the need for metrics during development to help make the resulting product more maintainable. Productivity in the maintenance stage is directly related to the maintainability of the product. Current maintenance distribution models can be categorized into two general schools of thought. One regards software maintenance as bearing unique features, while a second treats maintenance as essentially the same as testing.

[1] Frequently, software companies look for maximum productivity while developing their products, but leave the maintenance stage in second place. This is an error, because, as much experience has revealed, this is the very stage which consumes the greater portion of resources, more than 60%. Apart from the obvious economic implications, if productivity in the maintenance stage is low, the persons employed to develop the project may need to invest much of their time in later maintenance. Such is the experience which the authors have gleaned from previous projects and from the application of their estimating methods to real projects.

Consequently, if a company wishes to take on further projects, it must include an entirely new team among its staff. This means at least a partial loss from the experience gathered by the first team, becoming unavailable for further projects. Furthermore, the new team would need to be trained in methods and tools used by the software company. This has implications for validation efforts. Maintainability is the quality factor which includes all the features of the software to make it easier to maintain or which make the maintenance stage more productive. The question here is to ascertain the efficiency of the maintenance process by using the body of data collected in previous projects. These objective and validated data, when applied via the proposed techniques, conserve the maintainability characteristics while the product is being developed by providing for the needs forecast for the maintenance stage, and so avoid unnecessary future costs.

[1] This paper proposes a model for estimating maintenance cost, based on the experience gained from previous projects. We take the COCOMO (Boehm, 1981) as the basis of our model, in which we incorporate indices measuring the maintainability of the product.

We summarize a report in the form of cases near the end of this paper.

II. BACKGROUND

A. *Release Level Model:* [2]:

Instead of estimating the cost of the maintenance phase as a whole, another group of models focuses on the maintenance cost at a finer-grained level, estimating the effort of a planned set of maintenance tasks or a planned release. This approach usually involves using data from the past releases and analyzing the changes to estimate the cost for the next release.

These are the five processes used to calculate final maintenance costs.

a. *Maintainability index:*

[3] Maintainability is, beyond doubt, the software quality factor with the most influence in the maintenance stage. A study made by W. Itzfeld in Germany and compiled by Wallmu¨ller (1994) presents quality metrics ranking in which maintainability metrics were reported in first position by 67% of those asked. Using the definition of maintenance we summarized earlier, Boehm (1979) recognized the importance of maintainability.

One of his studies indicated that maintenance costs for software with low maintainability had a relation of 40 to 1 with respect to new development. It is obvious that an interdependent relationship exists between maintainability characteristics of developed software and maintenance cost.

So, in order to calculate the estimated maintenance cost we must consider a factor that indicates a measurement of the maintainability (facile in respect to maintenance) of the product. That is to say, we must ascertain the relationship between the estimated maintenance cost and those characteristics which make a program more or less maintainable. There are two steps to formulate this model:

a) Establish maintainability measurements.
b) Obtain the maintainability functions which relate the established metrics and the maintainability index.

Before taking up these two points we should bear in mind the three main activities which occur in maintenance. To reflect them, the maintainability index is divided into three component indices: an understandability index, a modifiability index and a testability index.

To calculate the maintenance cost, we consider a factor that indicates a measurement of the maintainability.

Taking as a starting point for estimating maintenance cost, Maintenance Index factor is included as follows:

$$MM_{main} = ACT \cdot MM_{dev} \cdot I_{main}$$

$$I_{main} = f(X_1, X_2, .., X_n)$$

I_{main} shows inverse degree of maintainability.

High values indicate low maintainability, low values indicate high maintainability.

I_{main} determines relationship between the estimated maintenance cost and characteristics.

This makes a program more or less maintainable.

There are two steps to formulate this model:

a) Establish maintainability measurements.
b) Obtain the maintainability functions which relate the established metrics and the maintainability index.

b. *Maintainability components:*

As just suggested, maintenance action can be dividing into three parts:

Understanding the changes to be made,
Modifying or making the change, and
Testing, or verifying the changes made.

These are clearly differentiated and performed one after the other, so the maintenance cost could be considered to be the sum of three costs expressed in man-months: understanding, modifying and testing:

$$MM_{MAIN} = MM_U + MM_M + MM_T$$

We will have three maintainability indexes, I_U,I_M and I_T, which relate the two parameters of a software project, ACT and MM_{DEV}, to the three components of maintenance cost expressed in man-months:

$$MM_U = ACT \cdot MM_{DEV} \cdot I_U$$

$$MM_M = ACT \cdot MM_{DEV} \cdot I_M$$

$$MM_T = ACT \cdot MM_{DEV} \cdot I_T$$

Consequently and in a parallel manner the maintainability index, I_{MAIN}, will be given as the sum of these three equally weighted indices which may have very differing values,

$$I_{MAIN} = I_U + I_M + I_T$$

Where:
I_{MAIN} = maintainability index,I_U = understandability index,I_M = modifiability index, and I_T = testability index.

c. *Maintainability metrics:*

The model proposed here, which has been used in case studies, considers only three software characteristics. Each one directly affects one maintainability component.

X_U: understandability metric

The number of comment lines for every 100 lines of code. We observe that there is a close relationship between the internal documentation of the code and understanding cost.

As expected, as the value of the understandability metric increases (number of comment lines), the understandability index (directly proportional to understandability cost) decreases.

X_M: modifiability metric

The number of lines without constant data for every 100 lines of code. We observe that the more numerous the constant data in the code, the bigger the modification cost.

X_T: testability metric

The number of error testing lines for every 100 lines of code. We observe that testing the code will be simpler if there are error detection and treatment procedures built into the code. These three characteristics have been chosen because we observe that they are easily measured, and they have a great influence on maintainability. Nevertheless, the model can be applied whatever the metrics chosen, provided the interdependence between each metric and its maintainability component can be demonstrated.

d. *Maintainability functions:*

[5]: To incorporate the maintainability metrics, we introduce the maintainability function F. This is a statistically determined relationship between the metrics X_U, X_M and X_T just described, and the indices I_U, I_M and I_T, and can be summarized as follows:

$$I_U = F_U(X_U)$$

$$I_M = F_M(X_M)$$

$$I_T = F_T(X_T)$$

To obtain these F functions, it is necessary to use something fundamental to all estimation processes, historical data. The experience acquired in former projects is of great value in estimating new projects. Thus, the management procedures of the software project must include mechanisms which allow us to take these measurements:

a) of the developed product:
   X_U: understandability metric,
   X_M: modifiability metric, and
   X_T: testability metric;
b) of the maintenance process:

MM_U: understanding cost expressed in man-months,
MM_M: modifying cost expressed in man-months, and
MM_T: testing cost expressed in man-months.

[4]The measurements of the maintenance process must be made annually. Every year, the annual change traffic (ACT) experienced must be determined, and with that, the cost expended in each task (understanding MM_U, modifying MM_M and testing MM_T) must be measured in man-months for the entire ACT. Maintainability indexes can then be obtained from the measured maintenance efforts using a simple formula.

For example and consistent with expression, for the understandability index, the formula that implements expression using historical data is:

$$I_U = MM_U / (ACT \cdot MM_{DEV})$$

Where:
I_U = understandability index,
MM_U = maintenance understanding cost expressed in man-months,
ACT = annual change traffic, and

MM_{DEV} = development cost expressed in man-months.

In a parallel manner, we obtain numeric values for I_M and I_T by using the modifying and testing efforts respectively, project by project.

Company's experts could assign relative weights to the ACT values on a project independent basis. Values of 1 carry the unweighted ACT values forward from the HT into the estimates of the future ACT.

T: Matrix of n ´ 1 elements indicating the average annual change traffic in each project **T** = (ACT_1 ACT_2 .. ACT_n)T. ACT_i = annual change traffic for project I where the superscript index 'T' expresses the operation transposed matrix.

C: Matrix of 1 ´ m elements indicating the characteristics of the current project of concern. The provision of these data requires the intervention of expert personnel.

This is when the group of characteristics will be revised. Modification of this group requires updating the HT, revising the characteristics of all the projects in the HT.

C = (c1, c2,..c_m)

Cj = Characteristic j for the current project

Two possible values:

1: The project has the characteristic

0: Otherwise

B: Matrix of n ´ 1 elements indicating the rate of coincidence the current project has in relation to each project of the HT—that is, the sum of characteristics they have in common (each characteristic contributes to the sum according to its historical ACT values).

$$\mathbf{B = A * C^T}$$

Where the symbol * indicates the product of matrixes.

Using these matrix definitions, the future ACT is then estimated by the following formula:

$$\mathbf{ACT = ((B * T^T)/(B^T * B))}$$

In this way each project is involved in calculating the estimate as far as its characteristics match those of the project under study.

e. Implementation method:

In this method, we calculate the values of understandability metric in man-months, modifiability metric in man-months, testability metric in man-months by fetching values from the history table.

To calculate the value of the parameter $MM_{u,}$ the formula is,

$$\mathbf{MM_u=ACT*MM_{DEV}*I_u}$$

Where, ACT values are derived with the above mentioned formula in maintainability functions and MM_{dev} is a constant value to be taken as 57.

To calculate the value of the parameter $MM_{M,}$ the formula is,

$$\mathbf{MM_M=ACT*MM_{DEV}*I_M}$$

To calculate the value of the parameter $MM_{T,}$ the formula is,

$$\mathbf{MM_T=ACT*MM_{DEV}*I_T}$$

The reason why MM_{dev} values is taken as constant is that if the value of MM_{dev} is varied then there will be no possibility of comparing the maintenance cost. Hence by setting the MM_{dev} as a constant, we vary the parameter Xu which will arrive in the formula of I_u.

The formula to calculate I_U is,

$$\mathbf{I_U=a*e^{b*Xu}}$$

The formula to calculate I_M is,

$$\mathbf{I_M=a*e^{b*X_M}}$$

The formula to calculate I_T is,

$$\mathbf{I_T=a*e^{b*X_T}}$$

Where the values of a and b are obtained by taking two equations and by solving those two equations based on history table.

We derive a and b values from the following equations.

$$\mathbf{\sum I'u_i=a'N+b\sum Xu_i}$$

$$\mathbf{\sum Xu_i I'u_i=a'\sum Xu_i+b\sum X^2u_i}$$

Here Xu value is varied based on the given range numerical value as 17 in the case study. By taking the range of starting value X_u, we can I_u can be calculated for different cases.

Finally, after calculating I_u, it is multiplied by the other two parameters ACT and MM_{dev} to get MM_u.

In the same way we calculate modifiability in man-months (MM_m), testability in man-months (MM_t).

Hence, Finally after calculating the values of MM_u,MM_m,MM_t, the values of all three parameters are added to obtain total Maintenance costs.

Hence the formula to calculate total maintenance cost is:

$$\mathbf{MM_{MAIN} = MM_U + MM_M + MM_T = ACT * MM_{DEV} * (I_U + I_M + I_T)}$$

The above mentioned formula is used to calculate software maintenance costs of any project which gives correct idea about the method and process to calculate software maintenance cost.

III. RESULTS AND DISCUSSION

Below are the results obtained by taking three cases, where in each case, development cost in man-months is taken as constant. For each case, we take maintainability metric ranges between 5 to 10 and find out software maintenance cost.

The results are as Follows:

CASE-1: Table-1 showing the order of results of maintenance cost under range of maintainability metric for case-1

$X_u/X_m/X_t$	MM_{MAIN}
5	25.84
6	24.08
7	22.792
8	21.912
9	21.42
10	21.25

In this case, we observe that as the maintainability metric value is increasing, software maintenance cost is decreasing. Software maintenance cost is mainly influenced by b value while calculating maintainability metric values. When b value is nearer to positive then final maintenance costs will increase as maintainability metric value increase.

History Table For Case-1:

PROJECT	X_T	I_T	I'_T=Ln I_T	X^2_T	$X_T I'_T$
P1	15	0.65	-0.430	225	-6.45
P2	11	0.75	-0.28	121	-3.08
P3	10	0.65	-0.430	100	-4.3
SUM	36	2.05	-1.14	446	-13.83

Hence, we conclude that as maintainability index value increases, maintenance cost decreases.

This can be applied for any project.

We calculate maintenance cost using history table values. History table values for each case is obtained by taking values from the range of history table given in case study

CASE-2: Table-2 showing the order of results.

$X_u/X_m/X_t$	MM_{MAIN}
5	36.058
6	31.56
7	27.89
8	24.97
9	22.63
10	20.76

Table -2 showing the order of results of maintenance cost under range of maintainability metric for case-2

This is the table which is taken to calculate a and b values for understandability index. In the same way we take other two tables to calculate modifiability index and testability index.

History Table For Case-2:

PROJECT	X_U	I_U	I'_U=Ln I_U	X^2_U	$X_U I'_U$
P1	7	0.30	-1.20	49	-8.4
P2	5	0.35	-1.04	25	-5.2
P3	8	0.23	-1.46	64	-11.68
SUM	20	0.88	-3.7	138	-25.28

In this case, at maintainability metric 5 the software maintenance cost is 36.058, at metric value 6, maintenance cost is 31.56. In the same way we calculate software maintenance cost at maintainability metric 10.

Hence we conclude that, as maintainability index value increases, maintenance cost decreases.

i. ***This can be applied for any project:***

We calculate maintenance cost using history table values. History table values for each case are obtained by taking values from the range of history table given in case study.

This is the table which is taken to calculate a and b values for modifiability index.

CASE-3: Table -3 showing the order of results of maintenance cost under range of maintainability metric for case-3

$X_u/X_m/X_t$	MM_{MAIN}
5	64.561
6	62.45
7	60.39
8	58.48
9	56.69
10	55.02

ii. In this case, the software maintenance cost is less at maintainability metric value 10. By taking 10 for all the three parameters that is understandabilty, modifiability, testability indexes, it is obtained that at 10 the maintenance cost is less that is 55.02.

iii. Hence we conclude that, as maintainability index value increases, maintenance cost decreases.

iv. This can be applied for any project.

We calculate maintenance cost using history table values. History table values for each case are obtained by taking values from the rage of history table given in case study.

This is the table which is taken to calculate a and b values for modifiability index. In the same way we take other table to calculate index and testability index.

History Table For Case-3

PROJECT	X_M	I_M	I'_M=Ln I_M	X^2_M	$X_M I'_M$
P1	23	1.08	0.07	529	16.1
P2	35	0.99	-0.01	1225	-0.35
P3	19	0.85	-0.16	361	-3.04
SUM	77	2.92	-0.1	2115	12.71

This is the history table which is taken for testability metric to calculate x_t,mm_t.

This can be applied for any project. We calculate maintenance cost using history table values. History table values for each case are obtained by taking values from the range of history table given in case study.

This is the table which is taken to calculate a and b values for testability index. In the same way we take other table to calculate modifiability index and modifiability index.

IV. CONCLUSION

Maintainability is a quality factor to be taken into consideration when estimating the cost of the maintenance stage in a software project. For this reason a factor for indicating the maintainability of a software product must be a part of the calculation of this estimation. This factor is called "maintainability index". The interdependence between this index and a set of software metrics, which represent maintainability characteristics, is of great interest.

The main element of this research is historical data from previous projects, an indispensable element for all activities including making estimations. In this paper the problems of estimating the cost of the maintenance process have been solved with our model, using data collected from previous projects.

Here the main concentration is to calculate software maintenance costs by varying metric values of understandability, modifiability and testability.

V. REFERENCES

[1]. B. W. Boehm, Software engineering economics, IEEE Transactions on Software Engineering ,vol. SE-10 no. 1, pp. 4–21, 1984.

[2]. http://csse.usc.edu/csse/TECHRPTS/PhD_Dissertations/files/Nguyen_Dissertation.pdf

[3]. "An Approach To Find Out At Which Maintainability Metric, Software Maintenance Cost Is Less" vol 3(4), 2012 pp.4599-4602 in IJCSIT

[4]. F. Niessink and H. van Vliet, Predicting maintenance effort with function points,‖ in Software Maintenance , Published in the Proceedings of ICSM'97, September 28 – October 2, 1997 . IEEE, 1997, p. 32.

[5]. Software Maintenance: Research And Practice, John Wiley & Sons, Ltd. VOL. 9, 161–175 (1997)