Journal of Computer Sciences 1 (4): 538-542, 2005
ISSN 1549-3636

Measurement of Software Maintainability Using a Fuzzy Model

[1]K.K. Aggarwal, [1]Yogesh Singh, [1]Pravin Chandra and [2]Manimala Puri
[1]GGS Indraprastha University, Delhi, India, [1]SIT, GGS Indraprastha University, Delhi, India
[1]SIT, GGS Indraprastha University, Delhi, India, [2]IT Department, D.Y.Patil, COE, Pune, India

Abstract: Software maintenance is a task that every development group has to face when the software is delivered to the customers' site, installed and is operational. The time spent and effort required for keeping software operational consumes about 40-70% of cost of entire life cycle. This study proposes a four parameter integrated measure of software maintainability using a fuzzy model. The study also includes empirical data of maintenance time of projects which has been used to validate the proposed model.

Key words: Software maintainability, average live variable, average. life span, cyclomatic complexity, comment ratio, fuzzy model

INTRODUCTION

Software maintenance is defined as process of modifying existing operational software while leaving its primary functions intact. Every software needs to be modified to meet customer's requirement in its life cycle. Software maintenance encompasses a broad range of activities including error corrections, enhancement of capabilities, deletion of obsolete capabilities and optimization[1] The value of Software can be enhanced by meeting additional requirements, making it easier to use, more efficient and employing newer technologies. Maintenance may span for fifteen years whereas development may be 1-2 year[2]. Even though it is an important task, it is poorly managed. The fact that you cannot control ,what you cannot measure, makes measurement of maintainability very important. In literature, some metrics have been proposed for measuring/predicting maintainability. In 1984 a tool was proposed[3] which operated at syntactic level. Then another model[4] was proposed which considered design attributes. Another model was proposed[5] which used quality metrics. Software Maturity index [SMI][6] considered only modules being added or removed.

Another model[7]was proposed which considers only the design aspect. A fuzzy model[8] has been proposed where, Maintainability is a measure of characteristics of software e.g. source code readability, documentation quality and cohesiveness among source code and documents. It is also seen that maintainability very much depends on the average number of live variables in a program and average life span of variables. Presently there is no model that considers the effect of these two factors. Therefore, a model which integrates the four factors namely average number of Live Variables $\overline{LV}$,average Life Span ($\overline{LS}$) of variables, the average Cyclomatic Complexity*(ACC*) and the Comments Ratio (*CR*) and provides a measure of maintainability, is proposed.

FACTORS AFFECTING MAINTAINABILITY

Average number of live variables: A live variable is live at a particular statement only if it is referenced a certain number of statements before or after that statement. The average number of live variables ($\overline{LV}$) is the sum of the count of Live Variables divided by the count of executable statements (n)

$\overline{LV}$ = LV/n

For a program having Modules $\overline{LV}_{\text{program}} = \frac{\sum_{i=1}^{m} \overline{LV_i}}{m}$ (1)

The more, the average number of live variables, the more difficult it would be to develop and to maintain a software.

b. Average live variable span: The span is the number of statements between two successive references of the some variable[9,10]. The average span size (LS) for a program could be completed using the equation.

$$\overline{LS}_{\text{program}} = \frac{\sum_{i=1}^{n} \overline{LS_i}}{n} \quad (2)$$

c. Comments ratio: Comment ratio is defined as

$$CR = (s+c)/c \quad (3)$$

Where s denotes total lines of code and c represents total number of comment lines. The lower the ratio, the better is the readability, and the better the readability,

Corresponding Author: Manimala Puri, [2]IT Department, D.Y.Patil, COE, Pune, India

the better is the maintainability. Comments provide better readability and therefore the Comments Ratio is an important factor that affects maintainability.

d. Average cyclomatic complexity: McCabe[11] has defined Cyclomatic Complexity as

$$V = e - n + 2p \quad (4)$$

Where e is the number of edges in a program flow graph, n the number of nodes and p the number of connected components. If p= 1, then v= ∏ +1 where ∏ is the number of predicates in the program. The Average Cyclomatic Complexity (*ACC*) is defined as average of cyclomatic complexities of all modules.

Proposed fuzzy model: There are four inputs to the fuzzy model, namely Average Live Variables, Average Life Span of variables, Average Cyclomatic Complexity and Comments Ratio. Fig 1 shows the Fuzzy Model.

Knowledge Date Base

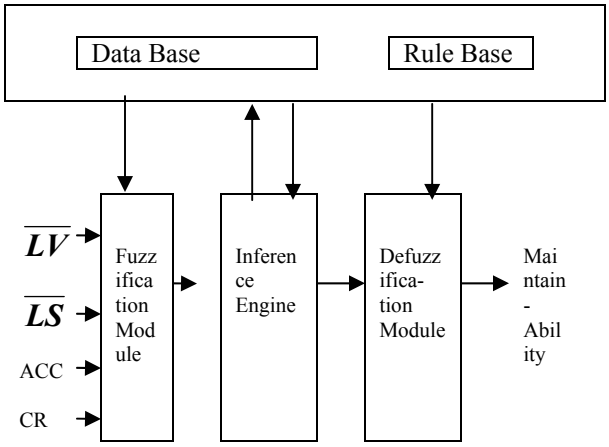

Fig 1: Fuzzy model for software maintainability

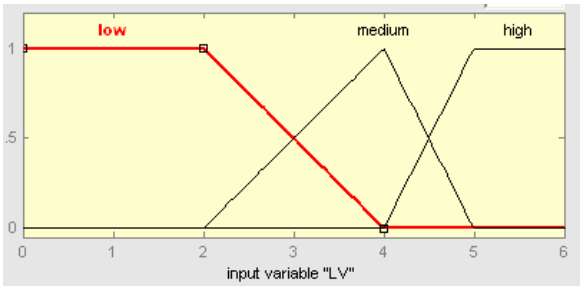

Fig. 2: Fuzzification of average live variable

This model considers all four inputs and provides a crisp value of maintainability using the Rule Base.

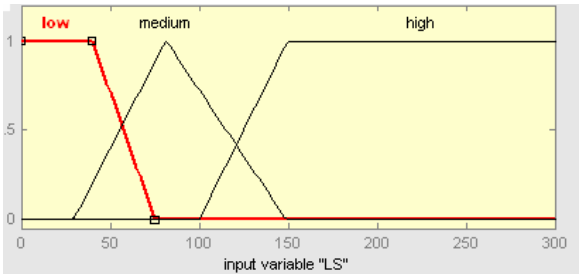

Fig. 3: Fuzzification of average life span

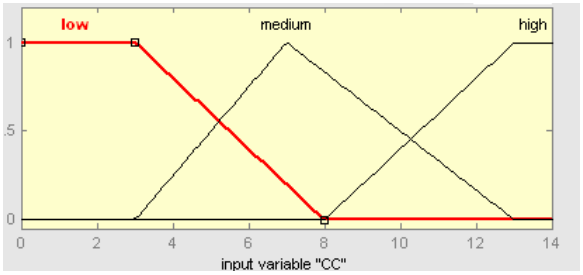

Fig. 4: Fuzzification of average cyclomatic complexity

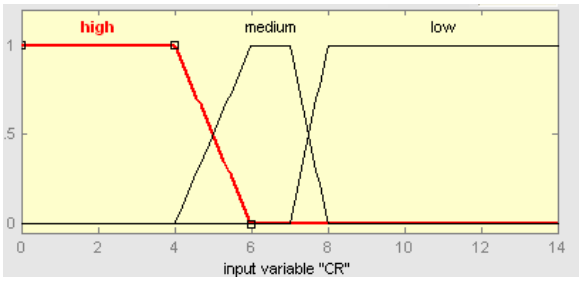

Fig 5: Fuzzification of comment ratio

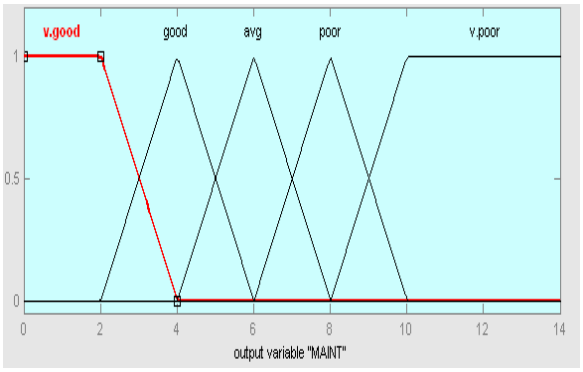

Fig 6: Fuzzification of output variable - maintainability

All inputs can be classified into fuzzy sets viz. Low, Medium and High. The output maintainability is classified as Very Good, Good Average, Poor and Very Poor.

In order to fuzzify the inputs, the following membership functions are chosen namely Low, Medium and High. They are shown in Fig. 2-5.

Similarly the output variable i.e. maintainability has five membership functions as shown in Fig 6.

MATERIALS AND METHODS

i. All the inputs and outputs were fuzzified as shown in Fig. 2 to 6.
ii. All possible combination of inputs were considered which leads to 3^4 i.e. 81 sets. The maintainability in case of all eighty-one combinations is classified as either Very Good, Good, Average, Poor or Very Poor by expert opinion. These lead to formation of 81 rules for the fuzzy model and some of them are shown below:

1. If (*CR* is low) and (*ACC*is low) and ($\overline{LV}$ is low) and ($\overline{LS}$ is low) then maintainability is very good.
2. If (*CR* is low) and (*ACC* is low) and ($\overline{LV}$ is low) and ($\overline{LS}$ is med) then maintainability is very good

.
.
.
.

81. If (*CR* is high) and (*ACC* is high) and ($\overline{LV}$ is high) and ($\overline{LS}$ is high) then maintainability is very poor.

iii. All eighty one riles are inserted and a rule base is created. Depending on a particular set of inputs, a rule will be fired.
iv. Mamdani style of inference is used.
v. Using the rule viewer ,output i.e maintainability is observed for a particular set of inputs using the MATLAB Fuzzy tool box.
vi. The output is also calculated theoretically using the Centre of gravity.

RESULTS

Output computation for the model: Let us say we have the following inputs to the model.

ACC=2,*CR*=12, $\overline{LV}$ =1, $\overline{LS}$ =130

When those inputs are fuzzified we find that *ACC*=2 belongs to fuzzy set low with membership grade 1, *CR*=12 belongs to fuzzy set low with membership grade 1, $\overline{LV}$ =1 belongs to fuzzy set low with membership grade 1 and $\overline{LS}$ = 130 belongs to fuzzy set low with membership grade = 0.25 and medium with membership grade 0.5.With these inputs, rule number 1 and 2 fire. During composition of these rules we get the following

Min (1,1,1,0.25)=0.25 Min (1,1,1,0.5)=0.5 .

When these two rules are implicated ,we find that the first rule gives maintainability very good to an extent of 0.25 and second rule gives the maintainability value good to the extent of 0.5. This is shown in Fig 7.

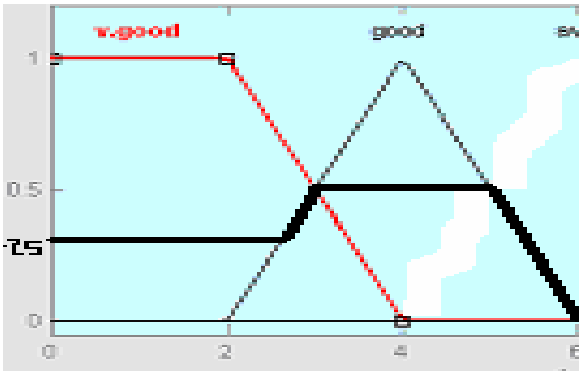

Fig. 7: Output computation of maintainability

Maintainability =

$$\frac{\int_0^{2.5}.25x\,dx+\int_{2.5}^{3}(mx+c)x\,dx+\int_3^5 0.5x\,dx+\int_5^6(mx+c)x\,dx}{\int_0^{2.5}.25dx+\int_{2.5}^{3}(mx+c)dx+\int_3^5 0.5dx+\int_5^6(mx+c)dx}$$

$$\frac{\int_0^{2.5}.25x\,dx+\int_{2.5}^{3}(0.5x-1)x\,dx+\int_3^5 0.5x\,dx+\int_5^6(-0.5x+3)x\,dx}{\int_0^{2.5}.25dx+\int_{2.5}^{3}(0.5x-1)dx+\int_3^5 0.5dx+\int_5^6(-0.5x+3)dx}$$

$$= \frac{\left[\frac{0.25x^2}{2}\right]_0^{2.5}+\left[\frac{0.5x^3}{3}-\frac{x^2}{2}\right]_{2.5}^{3}+\left[\frac{0.5x^2}{2}\right]_3^5+\left[\frac{-0.5x^3}{3}+\frac{3x^2}{2}\right]_5^6}{-[0.25x]_{00}^{2.5}+\left[\frac{0.5x^2}{2}-x\right]_{2.5}^{3}+[0.5x]_3^5+\left[\frac{-0.5x^2}{2}+3x\right]_5^6}$$

=3.2

Defuzzification: Defuzzification of the above output can be obtained by finding the Centre of Gravity[12] of the above fuzzy output.

The effect of these rules was observed also by simulating the model in MATLAB Fuzzy Tool Box. The maintainability for the above mentioned inputs comes out to be 3.2 which is the some as calculated above. The various surface views of the simulated model are shown in Fig. 8 and 9.

Empirical validation: In order to validate the model, ten procedure oriented software projects of undergraduate engineering students were considered. They were chosen only when proper set of input variables were available. Some logical errors were introduced in these projects and time take for corrective maintenance action was measured (Avg. CMT). The maintainability was also calculated using the proposed fuzzy model. The results are shown in Table 1.

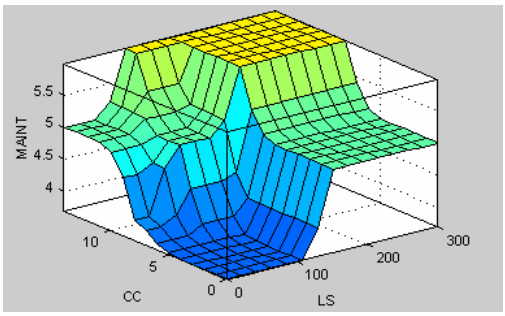

Fig 8: Surface view with $\overline{LS}$ input as x axis and *ACC* taken on y axis and maintainability on z axis

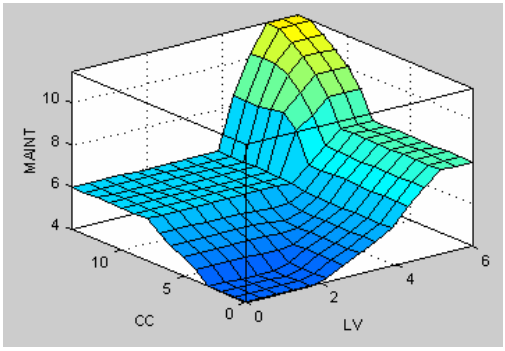

Fig 9: Surface view with $\overline{LV}$ input as x-axis and ACC taken on y-axis and maintainability on z axis

Table 1: Values of maintenance time and maintainability

P.No.	ACC	CR	$\overline{LV}$	$\overline{LS}$	Maint	Avg. CMT
1.	8.51	3.92	0.5	25.4	6	17.0
2.	11.5	7.74	2.5	132	5.39	16.10
3.	12.6	5.62	1.59	43.8	4.8	15.4
4.	5.28	8.30	4.41	238	6.87	18.0
5.	13.7	8.8	3.95	292	7.93	21.10
6.	7.43	7.32	2.32	118	4.49	15.0
7.	10.7	9.23	3.14	288	6.49	17.90
8.	9.37	6.89	3.14	141	6.49	17.20
9.	7.0	7.0	5.86	298	8	22.0
10.	10.7	8.8	6	300	10.5	25.2

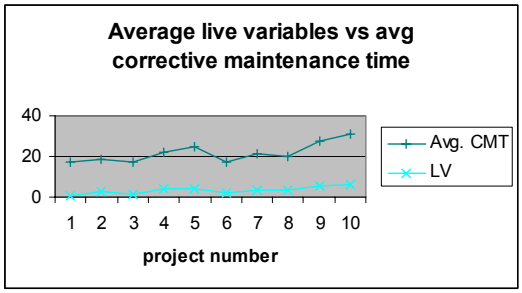

Fig.10: Avg. Live variable vs. avg. corrective maintenance time

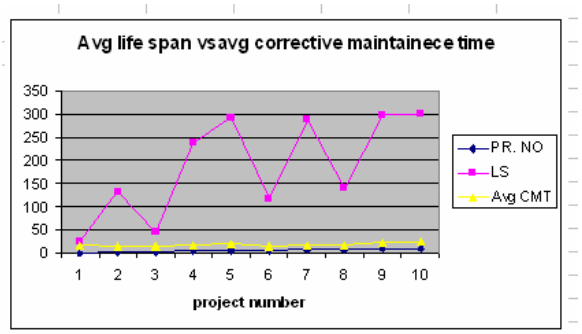

Fig. 11: Avg. life span vs. avg. corrective maintenance time

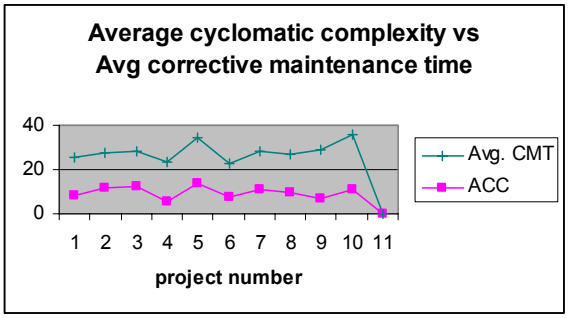

Fig. 12: Avg. cyclomatic complexity vs. avg. corrective maintenance time

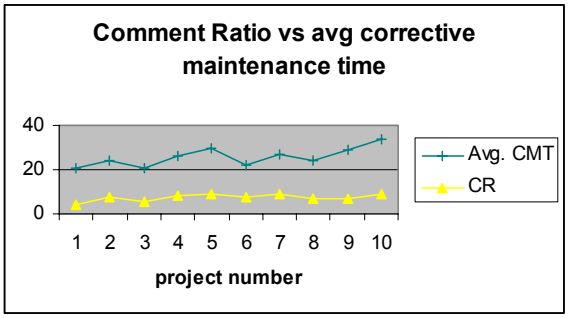

Fig. 13: Comment ratio vs. maintenance time

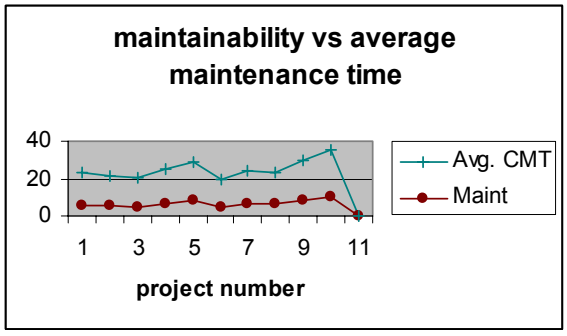

Fig.14: Maintainability vs. avg. corrective maintenance time

The values of average corrective maintenance time of these projects have been plotted against each of the four input metrics namely $\overline{LV}$, $\overline{LS}$,*ACC & CR* in Figs 10, 11 , 12 & 13respectively.

It can be seen there is hardly any co-relation between average maintenance time and the four inputs.These four metrics cannot individually predict the maintenance time.

On the other hand a plot of maintainability versus maintenance time is shown in Fig 14. This shows that integrated measure of maintainability is strongly co-related with maintenance time.

Thus the fuzzy model is validated and that the integrated value of maintenance gives better results than any individual input metric is also verified with the help of empirical results.

CONCLUSION

Maintainability can be estimated with the help of fuzzy model and the empirical results prove that the integrated measure of maintenance obtained from this model shows a strong co-relation to the maintenance time.

REFERENCES

1. Lamb, D.A., 1988. Software Engineering*:* Planning for Change. Prentice Hall, Engineering Cliffs, NJ.
2. Aggarwal, K.K. and Yogesh Singh, 2005. Software Engineering. Rev. Sec. Edn., New Age International Publisher.
3. Berns, G., 1984. Assessing Software Maintainability. Communications of the ACM, 27: 14-23.
4. Sneed, H. and A. Mercy, 1985. Automated Software Quality Assurance. IEEE Trans. Software Eng., 11Bi,9: 909-916.
5. Wake, S. and S. Henry, 1988. A model based on software quality factors which predicts maintainability. Proc. Conf. Software Maintenance, pp: 382-387.
6. Software Engineering Standards, 1994 Edition, IEEE.
7. Muthanna, S., K. Kontogiannis and B. Stacey, 2000. A maintainability model for industrial software systems using design level metrics. Proc. Seventh Working Conf. Reverse Eng.,Nov. 23-25, pp: 248-256.
8. Aggarwal, K.K., Yogesh Singh, Jitender Kumar Chhabra, 2003. A multiple parameter software complexity measure. J. CSI*,* 33: 22-30.
9. Elshoff J.L.L., 1978. An investigation into the effects of the counting method used on software science measurements. ACM SIGPLA Notices, 13: 30-45.
10. Aggarwal, K.K. and Yogesh Singh, 1994. A modified approach for software science measures. ACM SIGSOFT Software engineering Notes, USA.
11. McCabe, T.J., 1976. A complexity measure. IEEE Trans. Software Eng., SE-2: 308-319.
12. Roger Jang and Ned Gulley, 1995. Fuzzy Logic Toolbox for MATLAB. User's Guide. The Math Works Inc., USA.