DOI: http://dx.doi.org/10.26483/ijarcs.v8i8.4637

ISSN No. 0976-5697

Volume 8, No. 8,September-October 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

MAINTENANCE EFFORT PREDICTION MODEL USING ASPECT-ORIENTED COGNITIVE COMPLEXITY METRICS

G.Arockia Sahaya Sheela
Asst. Prof., Dept. of Computer Science
Holy Cross College Trichy
TamilNadu India.

Dr. A.Aloysius
Asst. Prof., Dept. of Computer Science
St. Joseph's College Trichy
TamilNadu India.

Abstract: Software development is a multifaceted process. It is challenging to define or measure software qualities and quantities and to determine a valid and concurrent measurement metric. In software development, a metric is the measurement of a particular characteristic of a program's performance or efficiency.The goal of software metrics is to improve understanding of a product or process. Aspect Oriented Programming (AOP) extends the traditional object-oriented programming (OOP) model to improve code reuse across different object hierarchies. AOP can be used with object oriented programming. AspectJ is an implementation of aspect-oriented programming for Java. Software maintenance is the most desired, but most elusive and difficult task in software engineering. The cost of maintenance is as high as 60% to 80% of the total cost of the software. So, plenty of this project are going on in software maintenance. Though, Aspect-oriented paradigm has made it easier, it remains the critical hotspot of research. One way of grappling with the maintenance problem, is to use the complexity metrics. Many studies were made to understand the relationship among complexity metrics, cognition, and maintenance. This paper wrestles with four newly proposed object-oriented cognitive complexity metrics to develop maintenance effort prediction models through various statistical techniques.Empirical study designs are made with ANOVA and experimented.Discussion on results proves the maintenance effort prediction models are more robust, more accurate, and can be employed to estimate the maintenance effort.

Keywords—maintenance effort prediction; cognitive complexity metrics; object-oriented metrics; software maintenance

I.INTRODUCTION

High software quality is the external hallmark of software engineering. Among the several software qualities maintainability and understandability are the most important and key qualities that are desired in the industry due to overall reduction in cost and effort [1]. The maintenance cost is as high as 60% to 80% of the total cost of the software [2]. In fact, it is the key for the survival of the product through the evolution as it faces many challenges from the constantly changing environments [3]. Software maintenance is the most desired, but most elusive and difficult task. Software maintenance is defined, as per ISO/IEC 9126 and IEEE 1219, "the process of modifying the software system or component after delivery to correct faults, improve performance or other attributes or adapt to a changed environment" [4]. There are four categories of maintenances, namely, corrective, perfective, adaptive, and preventive maintenance [3]. The corrective maintenance consumes about 21% and the adaptive and perfective maintenance takes about 75% of the maintenance effort. The perfective maintenance is the core problem of software maintenance during evolution [5]. Maintenance difficulty depends upon the complexity of the software system. To reduce the complexity, Aspect-Oriented (AO) paradigm is adopted which raises the cognition and eases the maintenance tasks [6]. Even the AO metrics, according to Wang and Shao [7], cannot reflect the real complexity of AO code since they consider only the structural aspect and do not bother about the cognitive aspects in calculating the code complexity. In fact, maintenance should be measured not only in terms structural complexity but also the amount of time taken to understand the program (cognitive aspect) and the effort needed to do the maintenance task [8]. In spite of all these measures, maintenance burden remains a critical area of research [9]. The relationship between the maintenance effort and AO metrics is complex and non-linear [10]. The cognitive weighted AO metrics further complicate it. Hence, the modeling and prediction of maintenance effort remain the hotspot of research and a lot of statistical models and sophisticated techniques are designed. This paper explores the relationship between themaintenance and four newly proposed complexity metrics by the authors.

II. SURVEY OF LITERATURE

Several studies have been conducted to examine the relationships among design complexity, program cognition and maintenance. As early as in 1976, Swanson et al., have categorized maintenance into corrective, perfective, adaptive, and later with Lientz added preventive maintenance [11,12] . Benestad et al. studied, how class-level measures of structural properties can be used to assess the maintainability of a software product as a whole [13].

In 2012, Al-Fawareh, studied various OO techniques like polymorphism, inheritance, dynamic biding, complex dependencies etc., from maintenance perspective [14]. Aloysius et al., in 2013, utilized three cognitive complexity metrics to develop a maintenance effort prediction model [9]. Michura et al. identified valuable attributes in determining the difficulty in implementing changes during maintenance [15].

III. EMPIRICAL STUDY DESIGN

The research design of empirical study is suggests that design complexity, maintenance task, and programmer ability influences maintenance performance. Maintenance performance is the dependent variable whereas design complexity and maintenance tasks are independent variables. This study is conducted to find the existence of relationship between cognitive complexity metrics and maintenance time, and to develop a model to predict the maintenance effort.

Maintainability is defined as the ease with which systems can be understood and modified [17]. In past studies, it has been measured as “number of lines of code changed [16] [18], time (required to make changes) and accuracy [19] [17], and “time to understand, develop, and implement modification” [20]. In this study, maintainability was measured as “time to understand, develop, and actually make modifications to existing programs [20]”. Accuracy has not been considered in the maintenance measurement because of the reasons that an inverse relationship exists between time (for making changes) and accuracy. All these metrics have been individually validated by comparing their values with similar metrics and have been found to be a better metrics.G. ArockiaSahaya Sheelaand Aloysius, the authors of this article, proposed the four design metrics namely CWMC [21], CWCAE[22], CWPA[23], and CWCoAR [24]these are the measures of design complexity considered in this study and mathematically defined, calibrated their cognitive weights, experimented with case studies.The second independent variable in the study is the maintenance task. Many of the researchers classify maintenance activities as adaptive, corrective, and perfective. Only two maintenance tasks are used in the study. They are perfective and corrective.

The design complexity is measured using cognitive complexity metrics, these metrics are to be validated. They are validated to find if these metrics are indeed valid metrics of design complexity in the contexts of both ‘Perfective Maintenance’ (treatment 1) and ‘Corrective Maintenance’ (treatment 2) tasks.

This study has been conducted to find the relationship between design complexity and maintenance time. It also proposes a model to predict the maintenance effort. There are numerous ways to assess the relationship between two variables. Some of them are t-test/ANOVA, correlation and regression. They can be used to see whether each complexity metric is a reliable indicator of expected maintenance time. The complexities for both the treatments, with each of the four metrics, CWMC, CWCAE, CWPA and CWCoAR are measured. Tests are conducted to validate the proposed metrics CWMC, CWCAE, CWPA and CWCoAR empirically. This is the primary objective of the experiment.

IV. EMPIRICAL EXPERIMENT

There are 122 developers who had two years of programming language experience in which they had at least six months of AOP experience. That the details are given in the table I

Table I. Summary of Programming Experience of Participants

Programming Experience	Mean	S
Total Programming Experience (Years)	43.73	17.3
AO Programming Experience (Years)	11.98	5.66
N = 122 Subjects		

Two independent treatments are used in the experiment, one involving ‘Perfective Maintenance’ and the other involving corrective maintenance, which constituted a required assignment Two versions of each treatment are constructed and designated as the “low-complexity” version and the “high-complexity” version based on their corresponding metric (CWMC, CWCAE, CWPA and CWCoAR) values. That the details are given in the table II.

Table II. Allocation of Subjects to Treatments

Maintenance	Perfective	Corrective	Total
Low Complexity	55	69	124
High Complexity	67	53	120
Total	122	122	244

Characteristics of the two systems as well as the corresponding metrics value are summarized in the Table III.

Table III. Characteristics of Programs used for Empirical Validation

	Quadrilateral (perfective)		Tractor – Trailer (Corrective)	
	Low Complexity	High Complexity	Low Complexity	High Complexity
#Classes	2	1	1	3
#Methods	5	18	10	14
CWMC	6	12	17	23
CWCAE	8	11	14	28
CWPA	12	22	23	40
CWCoAR	11	19	22	32

V. RESULTS AND DISCUSSION

For each treatment, a single factor ANOVA is performed to verify, if the dependent variable (maintenance time) for the two groups (high complexity and low complexity systems) are equal.The first and second treatment(Perfective and corrective) has the high-complexity version of the system had a higher MMT (126.13 for perfective and 159.77 for corrective) compared to the low-complexity version (109.09 for perfective and 106.10 for corrective). ANOVA is performed to test the statistical significance of this difference, and the results are verified. The analysis shows that the P-value is lesser than 0.00001. The system is categorized as high or low complexity, based on the values of the four metrics, it is concluded that a system with greater CWMC or CWCAE or CWPA or CWCoAR requires more time to perform a given maintenance task than the time required by a system with lower CWMC or CWCAE or CWPA or CWCoAR. Therefore, it is concluded that CWMC, CWCAE, CWPA and CWCoAR are valid complexity metrics, and there is a significant difference in the maintenance time required to make changes to systems, for the first treatment of all the four metrics. The table IV provides the difference level of complexity.

Table IV. ANOVAs for differences by Level of Complexity

Treatment	**Mean Maintenance Time (MMT)**		**F**	**P**
	Low Complexity	**High Complexity**		
1	109.09	126.13	36.15	<0.00001
2	106.10	159.77	55.36	<0.00001

A correlation analysis is applied in assessing the relationship between the metrics and maintenance time and the results are tabulated in Table V. From the Table V, it is observed that all the four metrics have high positive correlation. The maintenance effort prediction model used 80% data for model building and 20% for validating the model.

Table V Correlation Analysis for Model Building Data

Model Building	**CWMC**	**CWCAE**	**CWPA**	**CWCoAR**
80%	0.47	0.55	0.57	0.52

The results of regression analysis conducted to investigate the importance of the four complexity metrics (independent variables) in determining the maintenance time (dependent variable) are discussed in this section. Linear regression with one independent variable is performed for each of the four variables. Each of the variables, CWMC, CWCAE, CWPA and CWCoAR is found to have a statistically significant positive relationship with maintenance time. Based on the results, it is concluded that all the four metrics are valid predictors of maintenance time.

There are several criteria to evaluate the predictions of a model. The coefficient of multiple determinations adjusted R2 is used to indicate the amount of variance that is accounted for by the independent variables in a linear model. Because adjusted R2 tends to be an optimistic estimate of how well the model fits the population, adjusted R2 also compensates for the number of independent variables in the model.

Table VI. Regression Analysis - AOP-CCMS vs. Maintenance Time

Variable	**Test statistic n = 244**	**p-value α = .05**	**βi**	**Adjusted R2**
CWMC	8.28	<0.00001	2.962854	0.22
CWCAE	10.21	<0.00001	2.872796	0.30
CWPA	10.62	<0.00001	2.244747	0.32
CWCoAR	9.45	<0.00001	2.752814	0.27

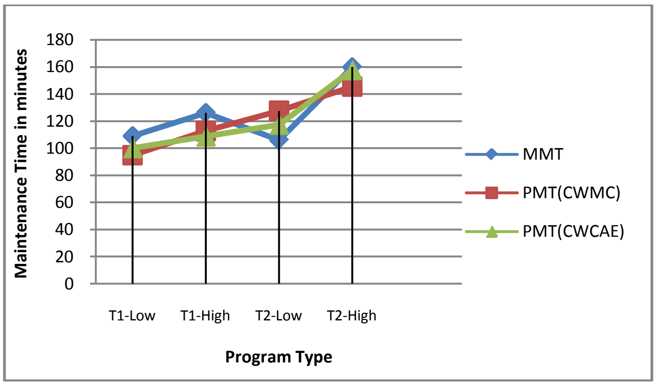
Predictive Maintenance Time (PMT) with each cognitive complexity metric can be derived from the data provided in Table VI. The predicting linear equations are listed below:

PMT = 77.1975 + 2.964761412* CWMC with 22% variance (1)

PMT = 77.1213+ 2.876116274* CWCAE with 30% variance (2)

PMT = 66.3855+ 2.245081749* CWPA with 32% variance. (3)

PMT = 62.6567+ 2.753505918* CWCoAR with 27% variance. (4)



Multiple regression analysis with all the four variables together is performed to determine the combined explanatory power of these variables.

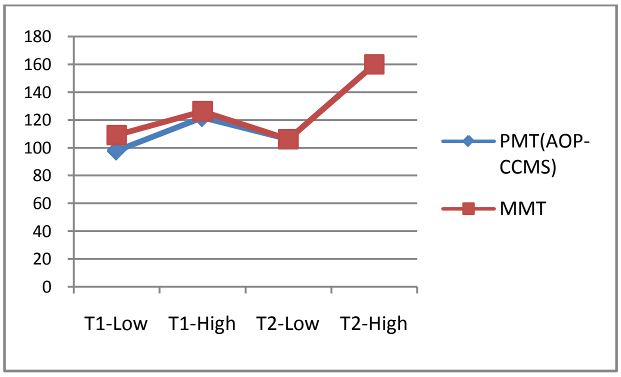
Table VII. Multiple Regression Analysis between AOP-CCMS and MMT

Variable	**Test statistic n = 244**	**p-value α = .05**	**βi**	**Adjusted R2**
CWMC	-2.52	0.012	-4.31	0.35
CWCAE	-0.24	0.81	-0.29	
CWPA	2.95	0.003	4.52	
CWCoAR	0.32	0.9	0.69	

The result of the regression analysis shown in Table VII confirms the percentage of variance as 35, which is greater than the percentage of variance of any other single metric value. This regression model is given as follows:

PMT = 63.990756-4.305158621*CWMC -0.289824794* CWCAE+ 4.52003111* CWPA+0.695203003*CWCoAR .....(5)

Thus, it is concluded that it is more appropriate to use combined cognitive metrics suite to predict maintenance effort.



VI. CONCLUSION AND FUTURE WORK

The main objective of this chapter is to empirically explore the validation of four Aspect-oriented cognitive complexity metrics of AOP-CCMS. For empirical validation, a controlled laboratory experiment is conducted to achieve the research objectives. Analysis of variance (ANOVA), correlation, and single and multiple regression analysis are used to quantitatively analyze the experimental data. It is found that each of the metrics CWMC ,CWCAE ,CWPA and CWCoAR metrics suite consisting of all the four are found to accurately measure the cognitive complexity of AO systems design. From the multiple regression models, it is concluded that combined metrics suite is a better predictor of maintenance effort than the individual cognitive complexity metrics and the existing metrics suite. Having empirically validated, the proposed the AOP-CCMS through maintenance effort prediction model, the following chapter summarizes the features of the proposed metrics suite and also provides directions for further research. For future work, the experiment can include other cognitivecomplexity metrics to bring out a bigger suite of maintenanceeffort estimation model. To confirm the results, programmersfrom software industry can be utilized. Further, large systemsfrom open sources can be studied for the same purpose.

VII. REFERENCE

[1] Sommerville, I., “Software Engineering,” 7th Ed., Addison Wesley, 2004.

[2] Ogheneovo, Edward E., “On the Relationship between Software Complexity and Maintenance Costs,” Jr. of Computer and Communications, vol. 2, no. 14, pp. 1-16, 2014.

[3] Mamdouh Alenezi and Mohammad Zarour, “Does Software Structures Quality Improve over Software Evolution? Evidences from Open-Source Projects,” Int. Jr. of Computer Science and Information Security (IJCSIS), vol. 14, pp. 61-75, February 2016.

[4] IQBBA, “Standard Glossary of Terms Used in Software Engineering,” Int. Qualifications Board for Business Analysis, 2011.

[5] Bennett, Keith H., and Václav T. Rajlich, “Software Maintenance and Evolution: A Roadmap,” In Proc. of the Conference on the Future of Software Engineering, ACM, pp. 73-87, 2000.

[6] Booch. G, “Object–Oriented Development,” IEEE Transactions on Software Engineering, vol. 12, no. 2, pp. 211-221, 1986.

[7] Yingxu Wang and Jingqiu Shao. J, “A New Measure of Software Complexity Based on Cognitive Weights,” IEEE Canadian Jr. of Electrical and Computer Engineering, pp.69-74, 2003.

[8] Aloysius A., “A Cognitive Complexity Metrics Suite for Object Oriented Design,” PhD Thesis, Bharathidasan University, Trichy, India, 2012.

[9] Aloysius, L. Arockiam, “Maintenance Effort Prediction Model Using Cognitive Complexity Metrics,” Int. Jr. of Advanced Research in Computer Science and Software Eng., vol. 3, no. 11, pp. 1599-1608, 2013.

[10] Kaur, Arvinder, Kamaldeep Kaur, and Ruchika Malhotra, “Soft Computing Approaches for Prediction of Software Maintenance Effort,” Int. Jr. of Computer Applications, vol. 1, no. 16, 2010.

[11] Swanson, E. Burton, “The Dimensions of Maintenance,” In Proceedings of the 2nd International Conference on Software Engineering, pp. 492- 497, IEEE Computer Society Press, 1976.

[12] Lientz B. P., Swanson E. B., “Software Maintenance Management,” Addison Wesley, Reading, MA, 1980.

[13] Benestad, Hans Christian, Bente Anda, and Erik Arisholm, “Assessing Software Product Maintainability Based on Class-Level Structural Measures,” Int. Conference on Product Focused Software Process Improvement, Springer Berlin Heidelberg, pp. 94-111, 2006.

[14] Al-Fawareh, Hamed J., “Modeling an Object Oriented for MaintenancePurposes,” Int. Jr. of Computers and Technology, vol. 3, no. 3, pp. 401- 405, 2012.

[15] John Michura, Miriam A. M. Capretz, and Shuying Wang, “Extension of OO Metrics Suite for SW Maintenance,” Hindawi Pub. Corporation, ISRN Software Engineering, vol. 2013, 14 pages, 2013.

[16] Li. Wand Henry. S, “Object Oriented Metrics that Predict Maintainability”, Journal of System Software, Vol. 23, Issue. 2, 1993, pp. 111-122.

[17] Gibson. V. R and Senn. J. A, “System Structure and Software Maintenance Performance”, Communications of the ACM Magazine, Vol. 32, Issue. 3, 1989, pp. 347-358.

[18] Li. Wand Henry. S, Kafura. D and Schulman. R, “Measuring Object-Oriented

[19] Design,” Journal of Object Oriented Programming, Vol. 8, No. 4, 1995, pp. 48-55.

[20] Crutis. B, Shepperd. S. B, Milliman. P, Borst. M. A, and Love. T,“Measuring the Psychological Complexity of Software Maintenance Tasks with the Halstead and McCabe Metrics”, IEEE Transactions Software Engineering, Vol. 5, Issue. 2, 1979, pp. 96-104.

[21] Rising. L. S, “Information Hiding Metrics for Modular ProgrammingLanguages,” PhD dissertation, Arizona State University, 1992.

[22] G.ArockiaSahaya Sheela, (2015). Analysis of Measuring the Complexity of Advice using a Cognitive Approach, International Journal of Applied Engineering Research, ISSN 0973-4562 Vol. 10 No.82 (2015). (Scopus Indexed)

[23]G.ArockiaSahaya Sheela, (2016). Design and Analysis of Aspect OrientedMetric CWCAE using Cognitive Approach,International Journal of Engineering Research & Technology (IJERT), ISSN: 2278 – 0181, Vol. 5 Issue 04, April, 2016.(Scopus Indexed)

[24] G.ArockiaSahaya Sheela, (2017). Design and Analysis of Aspect Oriented Metric CWCOAR using Cognitive Approach, IEEE, ISBN: 978-1-5090-5573-9, pp. 195-197.

