International Journal of Engineering Sciences Paradigms and Researches (IJESPR)
(Vol. 35, Issue 01) and (Publishing Month: October 2016)
(An Indexed, Referred and Impact Factor Journal)
ISSN: 2319-6564
www.ijesonline.com

A Critical Analysis of Object Oriented Software Complexity

Mukesh Bansal[1] and Chaitanya Purushottam Agrawal[2]

[1]Research Scholar, Makhanlal Chaturvedi National University of Journalism and Communication, Bhopal (India)
[2]Professor, Dept of Computer Science & Applications, Makhanlal Chaturvedi National University of Journalism and Communication, Bhopal (India)

Publishing Date: October 26, 2016

Abstract

The technological advancement and the benefits of the object oriented concepts make it unrealistic to design software without using object oriented languages. The use of object oriented concept has been increased due to quality provided on the proper usage of concepts. The proper usage of the object oriented concepts decrease the software complexity resulting better software quality. This paper discusses different software metrics to assess the object oriented software complexity. The paper focuses on the CK metric suite which includes different software metric like WMC, NOC, CBO, RFC etc, each of which has its own significance. These metrics are closely related to the factor i.e. Maintainability, Testability, Reusability, Understandability and Efficiency. These factors are independent to each other and used for to assess the complexity of the software. This paper has defined and evaluated factors combination which is used for the assessment of software complexity of object oriented software.

Keywords: *CK Metrics, Software Quality, Object Oriented, WMC, LOC.*

1. Introduction

Object oriented frameworks keep on sharing a noteworthy part of software improvement and client construct for these frameworks is in light of the ascent. This is on the grounds that there are tremendous motivating forces in adopting the object oriented strategy. The downside however is that most question arranged frameworks have a tendency to be very unpredictable i.e. complex. Henceforth, the nature of such frameworks takes priority and loads of time, cash and exertion is spent in guaranteeing it. One such technique that predicts nature of a product framework is by assessing key properties of the product using measurements. Software quality is particularly a better region when it comes than forecast in view of measurements. The presentation and consequent utilization of measurements as a way to assess the product quality has had profound and helpful effect on the general framework. Yet, the achievement of software quality appraisal through measurements is frustrated by the requirement for consistent approval to guarantee the precision of such expectations.

It is notable that the nature of programming is a "simple mindful, hard to characterize, and cannot be measured"[12] idea. With a specific end goal to clarify such an idea, a considerable measure of work has been done as of late. ISO/IEC 9126 Standard gives a system to surveying unpredictability of programming and meaning of 5 programming quality properties, including Reusability, Maintainability, Testability, Understandability and Efficiency. Diverse clients center around various programming quality properties. A protest situated programming more often than not contains countless properties, which can give more exact and far reaching depictions of programming's inward structure and nature. To date, there are huge quantities of measurements have been postured, among which Chidamber and Kemerer (C&K) metrics[17] are ended up being a common and valuable arrangement of Object-situated (OO) programming measurements, including DIT (Depth of the Inheritance Tree), NOC (Number Of Children), CBO (Coupling Between Object classes), LCOM (Lack of Cohesion in Methods), WMC (Weighted Methods for per Class), RFC (Response For a Class).

2. Review of Literature

Over the previous years, with the coming of new procedures, process driven administration numerous methodologies have been produced to address the issue of recognizing and rectifying configuration imperfections in an OO programming framework utilizing measurements. Also, with the regularly expanding number of programming

measurements being presented the undertaking supervisors think that its difficult to translate and comprehend the metric scores. Chidamber and Kemerer are the dominatingly referenced scientists, they proposed 6 measurements WMC,LCOM, RFC, CBO, NOC, DIT, with the assistance of which different programming quality characteristics (e.g. proficiency, multifaceted nature, understandability, reusability, viability and testability) can be estimated. They assert that utilizing a few of their measurements on the whole can help extend administrators and fashioners settle on better plan choices.

Disposition metric set model, proposed by Abreu [3] is another fundamental auxiliary technique for the question situated worldview. They were characterized to quantify the utilization of protest situated outline strategies, for example, legacy measurements, data concealing measurements, and polymorphism measurements. Abreu immovably recommended that measurements definitions and measurements ought to be supported as they assume vital part in outlining the question situated measurements. Inside the system that, numerous measurements that are connected to conventional practical improvement are additionally appropriate to question situated advancement, Rosenberg et al. [3] created nine measurements for question arranged framework, from which three were customary measurements i.e. Comment percentage, Lines of code, Cyclomatic Complexity and rest six measurements were same as CK measurements. They approved the six CK measurements at SATC and gave the connection between critical question arranged programming quality ideas, quality measurements and protest situated highlights as appeared in Table 2 [3]. Amjan Shaik et. al. in [4] performed factual investigation on the CK metric suite for Object situated frameworks. They found that if legitimately utilized, measurements could prompt a huge diminishment in cost of the general usage and quality change. L. Rosenberg et al. in [5] have recognized five characteristics for investigating plan and code of the product. These are proficiency, many-sided quality, understandability, reusability and testability/viability. Dr. Thapalyal, G. Verma in [6] played out an exact investigation of two measurements – CBO, WMC of CK metric suite to remove the relationship of these measurements with deserts.

3. Software Quality Factors

As a general rule object oriented advancement has demonstrated its incentive for frameworks that must be kept up and altered. The ideas of question arranged plan measurements are settled and numerous measurements identifying with item quality have been created and utilized. With protest arranged examination and outline approaches picking up notoriety, the time has come to begin exploring object situated plan measurements regarding programming quality. Estimating quality in the beginning time of programming improvement is the way to grow top notch programming. There must be an approach to survey protest arranged programming many-sided quality as right on time as conceivable in the improvement cycle [7]. McCall proposed a helpful order of variables that influence programming quality as appeared in figure 1.

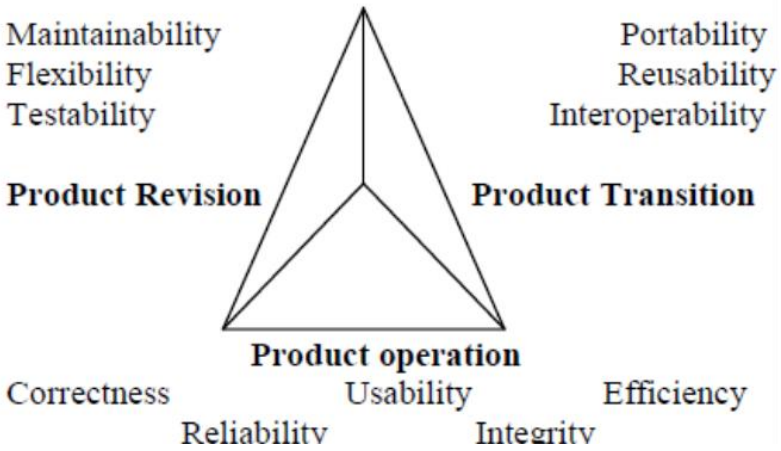

Figure 1: McCall's Quality factors [Source Pressman Fifth Edition]

4. Object Oriented Metrics

Chidamber and Kemerer (CK) [8] are the for the most part referenced scientists. CK measurements were characterized to gauge the intricacy of the product in connection to their effect on quality traits, for example, Reusability, Maintainability, Testability, Understandability and Efficiency and so forth. A few examinations have been directed to approve CK measurements**.**

Table 1: CK Metric Suite

CK Metric	Definition
WMC	Number of methods of a certain class without inherited methods
RFC	Number of methods that can be performed by a certain class regarding a received message
LCOM	Number of disjunctive method pairs of a certain class
CBO	Number of couplings between a certain class and all other classes
DIT	Maximal depth of a certain class in an inheritance structure
NOC	Number of direct subclasses of a certain class

CK measurements are gone for evaluating the plan of protest arranged framework instead of usage. This make them more suited to protest situated worldview as question arranged outline put incredible accentuation on the plan period of programming framework. The connection between essential protest situated programming quality ideas, CK measurements and question arranged (OO) highlights is given in Table 2 [5].

Table 2 Association of software Quality Factors and OO Features with CK metric Suite

CK Metric	Concept/Factor	OO Feature
DIT	Reusability, Efficiency, Understandability, Testability	Inheritance
NOC	Reusability, Efficiency, Testability	Inheritance
CBO	Reusability, Efficiency	Coupling
LCOM	Reusability, Efficiency, Complexity	Cohesion
WMC	Maintainability, Understandability, Usability, Reusability	Class/Method
RFC	Understandability, Usability, Testability	Class/Method

5. Analysis of Software Metrics for Object Oriented Software

The assessment and distinguishing proof of testability of programming depends on programming quality variables and their related measurements. The point of this work is to build up a model for the appraisal of multifaceted nature that can be estimated utilizing these measurements. A few elements have been widely perceived in OO like Inheritance, Coupling and Structure. We have recognized four fundamental measurements for the multifaceted nature assessment in OO programming.

It is deduced in our past research paper that four out of six measurements WMC, NOC, CBO and RFC are reasonable for many-sided quality and quality estimation [2]**.**

A. Inheritance Metrics

Number of children can be characterized as the quantity of prompt subclasses of a class. NOC measures the expansiveness of a class progressive

system. The higher the estimation of NOC, the less the issues, which is attractive [18]. NOC, in this way, essentially assesses effectiveness, reusability, and testability [19]. Legacy measurements is Number of Children (NOC).

B. Coupling Metrics

Coupling between objects [9] can be characterized as no. of coupled classes inside all classes. Low coupling is attractive for better outline. In [10], a metric suite for OO is given and coupling is considered as one of the principle measurements in structure. Coady et al. [11] gives isolate set of measurements to couplings in question arranged. Coupling metric is Coupling between Objects (CBO).

C. Structural Metrics

It is a mix of the total of the complexities of all strategies for a class and the arrangement of techniques that can possibly be executed in light of a message got by a protest of that class [1].

➢ The vast no. of techniques in a class, the more noteworthy the potential effect on kids. Classes with vast no. of techniques are probably going to be more application particular, constraining the likelihood of reuse. So WMC has negative effect or reusability of a class.

➢ The bigger the no. of techniques that can be summoned from a class through message, the more noteworthy the unpredictability of the class. So RFC has negative effect on reusability of a class.

Auxiliary Metric are as per the following: (I) WMC (ii) RFC

As, many-sided quality of OO Software essentially relies upon the measurements, for example, (I) Inheritance (ii) Coupling (iii) Structural. These measurements identify with the components i.e. Reusability, Maintainability, Testability, Understandability and Efficiency. So as to get to multifaceted nature of OOS, it is likewise exceptionally hard to decide level of commitment of these measurements in unpredictability. To conquer these challenges, we adjusted fluffy rationale procedure thinking about the four measurements as info variable and unpredictability as yield variable. These metrics has their role based on the application of the software. Different software metric uses and their corresponding factors already have been defined in the table 2. It can be easily determined that the increase in one factor may result in decrement in the other factor. That's why, based on the application different optimization technique can be applied to use the software metric to assess the software. Meta-heuristic optimization techniques are turning out to be increasingly well known in designing applications since they : (i) depend on rather basic ideas and are anything but difficult to execute; (ii) don't require angle data; (iii) can sidestep neighborhood optima; (iv) can be used in an extensive variety of issues covering diverse controls. Nature-inspired meta-heuristic calculations take care of optimization issues by copying organic or physical wonders. They can be gathered in three principle classes development based, material science based, and swarm-based strategies. Development based techniques are propelled by the laws of normal advancement. The hunt procedure begins with an arbitrarily produced populace which is advanced over consequent eras. The quality purpose of these strategies is that the best people are constantly consolidated together to frame the up and coming era of people. This permits the populace to be streamlined through the span of eras. The most mainstream development enlivened procedure is Genetic Algorithms (GA) that reenacts the Darwinian advancement. Other well known calculations are Genetic Programming (GP) and Biogeography-Based Optimizer (BBO). These optimization techniques depends upon the parameter tuning while some latest technique like teacher leaning based optimization are the algorithms doesn't depend upon the parameter. Such technique should optimize the software metric properly.

6. Conclusion

This paper describes the CK metric suite along with the relation of each metric with the different software quality factors. Moreover, this paper also establishes the relation between the software metric, the quality factor and the object oriented feature. This gives the opportunity to use different software metric as per the application area of the software. In future different optimization technique can be designed to select the appropriate software metric to assess any particular software.

References

[1] Goel, B. M., Bhatia, P. K., 2013. ACM SIGSOFT Software Engineering Notes, ACM,

New York, USA, Vol. 38 No. 4, July 2013, pp. 1-5.

[2] Bansal, M., Agarwal, C. P. 2014. Critical Analysis of Object Oriented Metrics in Software Development, 2014 Fourth International Conference on "Advanced Computing & Communication Technologies", IEEE Conference Publishing Services, R.G. Education Society, Rtk., 8-9 Feb., 2014 , pp. 197-201.

[3] J. Bansiya and C.G. Davis, 2002. A Hierarchical Model for Object-Oriented Design Quality Assessment, IEEE Transactions on Software Engineering, Vol. 28, No. 1.

[4] Boehm, B. W., Brown, J. R., and Lipow, M. L., 1976. Quantitative Evaluation of Software Quality. In IEEE Proceedings of the 2nd International Conference on Software Engineering, San Francisco, California, United States, pp: 592- 605.

[5] L. H. Rosenberg and L. Hyatt, 1997. Software Quality Metrics for Object- Oriented Environments, Crosstalk Journal.

[6] S. Jamali, 2006. Object Oriented Metrics, Sharif University of Technology.

[7] L. Rosenberg and L. Hyatt, 1995, Quality Metrics for Object-Oriented System Environments, NASA Technical Report.

[8] S. R. Chidamber and C. F. Kemerer, 1994. A Metrics Suite for Object Oriented Design, IEEE Transactions on Software Engineering, Vol. 20, No. 6, pp. 476–493.

[9] Aopmetrics project. http://aopmetrics.tigris.org/

[10] Chidamber S.R., Kemerer C.F., ―A Metrics Suite for Object Oriented Design‖, Software Engineering, IEEE Transactions, Vol. 20, No. 6, 476–493, 1994.

[11] Coady Y., Kiczales G., ―Back to the Future: A Retroactive Study of Aspect Evolution in Operating System Code‖, Published in proceedings of the 2nd International Conference on Aspect Oriented Software Development, ACM Press, pp. 50–59, New York, NY, USA, 2003.

[12] B. Kitchenham, "Software metrics in Software Reliaility Handbook [M]," London, New York: P. Rook. Elsevier Applied Science, 1990.

[13] Sivanandam, S.N., Sumathi, S., Deepa, S.N., ―Introduction to Fuzzy Logic using MATLAB‖, Springer, Heidel-berg, 2007.

[14] MacDonell S.G., Gray A.R., Calvert J.M., ―Fuzzy Logic for Software Metric Practitioners and Researchers‖, Published in Proceedings of the 6th International Conference on Neural Information Processing ICONIP, Perth, pp. 308-313, 1999.

[15] Ryder J., ―Fuzzy Modeling of Software Effort Prediction‖, Published in Proceedings of IEEE Information Technology Conference, pp. 53-56, Syracuse, New York, 1998.

[16] http://www.mathworks.in/help/fuzzy/building-systems-with-fuzzy-logic-toolbox-software.html (last accessed on 12/02/14).

[17] M.H. Tang, M.H. Kao, and M.H. Chen, "An empirical study on object-oriented metrics [C]," Proceedings of 6th International Symposium on Software Metrics, 4-6 Nov 1999, pp.242-249.

[18] Henderson-sellers, B. 1996. Object-Oriented Metrics, Measures of Complexity, rentice Hall.

[19] Bieman, J., and Karunanithi, S. 1993 Candidate reuse metrics for Object Oriented and Ada Software, In Proceedings of IEEE-CS First International Software Metrics Symposium.