Global Journal of Computer Science and Technology: C
Software & Data Engineering
Volume 20 Issue 2 Version 1.0 Year 2020
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

A Review of Metrics for Object-Oriented Design

By Neyole Misiko
UMMA University

Abstract- The ever-evolving body of empirical results do confirmation on the theoretical perspective the validity of OOD metrics whose validity is determined by them demonstrating that [1] they measure what they purport to measure. Quite often OOD metrics have been used as indicators of both the internal and external behaviors in the software development process. Software metrics especially for Object Oriented Systems literature often describe complex models with the focus to help predict various properties of software products and processes by measuring other properties. Usually designers are met with challenges to work with these measures especially when and how to use them. The very process of collecting these measurements leads to a better organization of the software process and a better understanding of what designers do as long as they confine to measurements that are meaningful. To this end therefore, the initiation of these metrics during the initial software development process is important. This paper elicits an understanding of the OOD metrics used in OOS development.

Index Terms: MOOD, OOD, metrics, software quality.

GJCST-C Classification: D.2.2

AREVIEWOFMETRICSFOROBJECTORIENTEDDESIGN

Strictly as per the compliance and regulations of:

© 2020. Neyole Misiko. This is a research/review paper, distributed under the terms of the Creative Commons Attribution-Noncommercial 3.0 Unported License http://creativecommons.org/licenses/by-nc/3.0/), permitting all non commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

A Review of Metrics for Object-Oriented Design

Neyole Misiko

Abstract- The ever-evolving body of empirical results do confirmation on the theoretical perspective the validity of OOD metrics whose validity is determined by them demonstrating that [1] they measure what they purport to measure. Quite often OOD metrics have been used as indicators of both the internal and external behaviors in the software development process. Software metrics especially for Object Oriented Systems literature often describe complex models with the focus to help predict various properties of software products and processes by measuring other properties. Usually designers are met with challenges to work with these measures especially when and how to use them. The very process of collecting these measurements leads to a better organization of the software process and a better understanding of what designers do as long as they confine to measurements that are meaningful. To this end therefore, the initiation of these metrics during the initial software development process is important. This paper elicits an understanding of the OOD metrics used in OOS development.

Index Terms: MOOD, OOD, metrics, software quality.

I. Introduction

Software metrics plays a key role in good software engineering. Measurement is used to assess situations, track progress and evaluate effectives of software products. But there exists a huge challenge in the measurement process due to lack of coordinated, comprehensive framework for understanding and using measurement [2]. Object-oriented approach to software development requires some specific set of metrics [3]. Various object-oriented measurements are used to evaluate and predict the quality of software products [4], where the empirical results are used to supports the theoretical validity of the Object-Oriented Software Product metrics [5]. The validity of these metrics needs to facilitate the accuracy that the metric measure what they purport to measure.

II. Software Engineering Metrics and Quality

According to Edward V. Berard [6] Metrics are units of measurement that refer to a set of specific measurements taken on a particular item or process. For software engineering metrics are units of measurement used to characterize software engineering products, processes and the people, hence assessing quality. Ahmad S et.al [7] indicated that Software metrics are measures that facilitate software developers and software analyst to preview into the efficiency of the software process and projects that are conducted using the process as framework. These metrics measures different aspects of software complexity hence play an important role in analysing and improving software quality [8].

Mahfuzul Huda et.al [9] argued that the quality of any object-oriented design is critical as it has a great influence on the overall quality of finally delivered software product. Further he asserts that Software quality is still a vague terminology since it has different meaning to different people, the way one measure quality depends on the viewpoint he/she takes [10]. Acceptable object-oriented design properties and associated metrics are helpful when utilized in the early stage of software development process, since the metrics determination is an important phase in testability estimation process [11].

Quality in the use of Object-Oriented Software Engineering metrics are available when the final product is in use in real conditions. Here the internal quality determines the external quality, while the external quality determines quality in use [12]. According to the GE model for describing software quality, presented by McCall et al. (1977), software quality is organized around three main types of quality characteristic:- factors which describe the external view of the software, as viewed by the users, criteria which describe the internal view of the software, as seen by the developer and the metrics which control and are defined and used to provide a scale and method for measurement.

With the help of software metric software designers are able to deeper understand the software product in an effective way as they use diverse measurements of computer software in development. Thus, though software metric we are able to measure some property of software's including their components considering that software quality metrics to be subset of software metrics they are helpful [7]. To this end, with the aid of OOD metric therefore, software professionals can then use object oriented metric suite to predict and enhance the maintainability of software with least error and best precision in an object-oriented paradigm [13].

III. Issues in Software Engineering Metrics

Berard E argued that if used properly, software engineering metrics enables us among others to qualitatively and quantitatively define success and failure by establishing the degree of success or failure and identify and quantify improvement [6]. The objective of the ISO/IEC 9126 standards is to address the human

Author: UMMA University. e-mail: jneyole434@gmail.com

limitations that canadversely affect the final software engineering development project. Some of the issues addressed include the change of focus after the start of a project. The standards provide clarity through agreeing on the project priorities and converting the compliance to measurable output values that can be validated against schema with total zero interventions, the standards therefore facilitate a common understanding of software engineering project's objectives and goals [14] These ISO/IEC 9126 standard further classified into four main parts: - the quality model, external metrics, internal metrics and quality in use metrics. However, the use of these design metrics is limited in practice due to the difficulty of measuring and using a large number of metrics.

Fenton and Neil [15] journal indicated that the major problem is in using such metrics in isolation. They argued that it was possible to provide a genuine improved management decision support system based on suchsimplistic metrics, but only by adopting a less isolationist approach. Much as software metrics play an important role in developing high quality software as well as to improve the developer's productivity [16] there comes the problem of identifying the right metrics to be used at a given stage of the OOD process.

Emphasis of introducing the metrics during the intimal software development is vital. OO designs are highly involved, often ill-defined, complex and iterative process. Their needs and specifications get more refined only as the design process moves toward its final stages. This therefore calls for effective metric tools that will help the designer make better-informed decisions with proven efficient knowledge representation schemes.

IV. Object-Oriented Design Metrics

Aggarwal et.al (2013) indicated that metrics for OO design entails measurements that are applied to the class and design characteristics [17], as they aim achieve quality in software process and product, This OO metrics measurement tools have yet to achieve the needed degree of maturity [18] they therefore need standardization [19]. Chidamber et.al [20] indicated that while metrics for the traditional functional decomposition and data analysis design approach measure the design structure and data structure independently, the object-oriented metrics need to focus on the combination of both the function and data as an integrated object. Despite the metric being traditional or new, it should be able effective to measure at least one or mere OOSD attributes of a software engineering product [21].

There exist various metrics for object Oriented designs otherwise called MOOD (Metrics for Object Oriented Designs). According to F.B. Abreu et al [22] metrics for Object Oriented Designs define the structural models of a software engineering design where they facilitate measurements of OO paradigms such as encapsulation, inheritance, polymorphism and message passing. These metrics are usually expressed to measure where the numerator defines the actual use of a feature for a design namely the method and attributes. The attributes represent the status of object in the system while method is used to maintain or modify the several kinds of status of the object [23].

Sahar et.al [24] stated that the most important measures that need to be considered in any software product is in the design quality. He established that design phase takes only 5–10 % of the total effort but a large part up to 80% of total effort goes into correcting bad design decisions [25]. The MOOD metrics include: - Method Hiding Factor (MHF), the Attribute Hiding Factor (AHF), the Method Inheritance Factor (MIF), the Coupling Factor (COF), the Attribute Inheritance Factor (AIF) and the Polymorphism Factor (PF) [17]. Each MOOD metric is associated with basic structural mechanisms of the object-oriented paradigm [26]. The MOOD metric set enables expression of some recommendations for designers [27].

Malhotra et.al [28] indicated that design of a system plays an essential role in ascertaining the system’s reaction to incoming changes, and well-chosen OO design metrics can function as an indicator of changeability. Gupta & Saxena [29] stated thatthe prediction of software defect is possible on the basis of historical data accumulated during implementation of similar or same software projects or it can be developed using design metrics collected during design phase of software development.

Chidamber and Kemerer [30] theoretical presentation on OO design metrics for software development life cycle are based upon OOD measurement theories that are used by OO software developers. The key requirements of metric measurements by Chidamber and Kemerer [20] focused on improving the quality of software with the help of a new metrics suite that consists of six design level metrics named WMC, DIT, NOC, CBO, RFC and LCOM [29]. According to Shyam Chidamber and Chris Kemerer [31] on the role of metrics for OOD indicated that the important components of process improvement is the ability to measure the process. Their paper provided the appreciation of development and empirical validation of sets of theoretically-grounded metrics of OO designs.

V. Oodmetrics for Analysis

Object Oriented Software Engineering product code is analyzed through object-oriented metrics, two suites of metrics are used, the Chidamber-Kemerer (CK) [20] and MOOD [1] [32] suites. Many of the OOD software’s usually fail due to poor quality especially when the estimation of software quality is not prioritized during the software development. Mago et.al [33] indicated that design metrics play an important role in helping developers to appreciate design aspects of software especially to the improvement of software quality. Thus, through the analysis of the OOD metric data one can forecast the quality of the object-oriented system. Boehm et.al [34] stated that to produce high-quality Object-Oriented applications a strong emphasis on design aspects is highly necessary. To this end therefore OOD software metrics among other metrics should make it possible for software engineers to measure and predict software processes, necessary resources for a project and products relevant for a software development effort. Software quality for OOD is the degree to which OO software possesses required combinations of attributes such as reliability, maintainability, efficiency, portability, usability and reusability.

Object oriented design are intended to capture the fundamental structure of an object-oriented program. The, set of components which can evaluate, represent and implement an object-oriented design include attributes, methods, objects/ classes, relationships and class hierarchies and must be addressed during the whole process of OOSD process. Measuring software quality in the early stages of software development is the key to develop high quality software [33]. During the OOD process analysis of model captures the logical information about the system, while the design model adds details to support efficient information access. This is important; however, the optimizing process must also be considered so as to make the implementation more efficient.

Despite this, design optimization should not be extreme since the ease of implementation, maintainability, and extensibility need to be considered. Often a perfectly optimized design is usually more efficient but less readable and reusable. Designers must strike a balance between the two. Factor to be considered in the analysis include: - addition of redundant associations [35], omission of non-usable associations [36], optimization of algorithms [37] and storage of derived attributes to avoid re-computation of complex expressions.

VI. Internal Metrics

Internal events are those that pass from one object to another object within a system. Dubey et.al [38] stated that metrics provide insight necessary to create and design model through the test. It also provides a quantative way to access the quality of internal attributes of the product, thereby it enables the software engineer to access quality before the product is build [39]. OOD metrics are thus crucial source of information through which a software developer takes a decision for design good software. For instance, through the Reliability metrics, the quality of internal product can be measured by the number of bugs in the software and by the duration of software metrics crash. The Class Method Complexity (CMC) metric defined as the summation of the internal structural complexity of all local methods is a theoretical basis and viewpoints. The metrics greatly affect the effort required to design, implement, test and maintain a class [40].

VII. External Metrics

Punia et.al [40] indicated that the external metrics are used to examine and reuse of an OO system. External events are those events that pass from a user of the system to the objects within the system. For example, mouse click or key–press by the user are external events. For instance, the MPC (Message Pass Coupling) metric addresses the external methods which are the number of send statements defined in a particular OOS class. When a message invokes numerous methods as a response, the class becomes more complicated and more testing and debugging is required [41].

Bidve and Khare [42] indicated that coupling in software has been associated with the maintainability and is used as predictors of external software quality attributes such as fault-proneness, impact analysis, ripple effects of changes, changeability. Shaik et.al [43] stated that external validation involves empirically demonstrating that the product metric is associated with some important external metric. Shaik et.al further states that high cognitive complexity leads to a component exhibiting undesirable external qualities, such as increased fault proneness and reduced maintainability. Accordingly, object-oriented product metrics that affect cognitive complexity will be related with fault-proneness. From the above, the underlying assumption is that such measures can be used as objective measure to predict various external quality aspects of the code or design artifacts [44].

VIII. Conclusion

Dubey et.al [38] indicated that the popularity of object-oriented design metrics is essential in software engineering for measuring the software complexity, estimating size, quality and project efforts. Object-oriented metrics assures to provide OOD that are reliable, maintainable and reusable software products. The initiation of various OOD metrics during the software initial development process in vital as this will enable designers eliminate bugs and limitations making the software product be of good quality. Increasingly, object-oriented design measurements are being used to evaluate and predict the quality of software [4] through prediction SE are able to improve the software product performance as well as enhance more user requirements during and after the OOS design.

© 2020 Global Journals

References Références Referencias

1. F. B. Abreu, "The MOOD Metrics Set," in ECOOP'95 Workshop on Metrics, 1995.
2. N. Fenton and J. Bieman, Software Metrics: ARigorous and Practical Approach, Third Edition, Colorado USA: CRC Press- Tayler & Francis Group, 2014.
3. D. Rodriguez and R. Harrison, An Overview ofObject-Oriented Design Metrics, United Kingdom: The University of Reading UK, 2001.
4. R. Harrison, S. J. Counsell and R. V. Nithi, "An Evaluation of the MOOD Set of Object-Oriented Software Metrics," IEEE Transactions on Software Engineering, vol. 24, pp. 491-496, 1998.
5. D. Glasberg, K. E. Emam, W. Melo and Madhavji, "Validating Object-Oriented Design Metrics on a Commercial Java Application," National Research Council 44146, 01 September 2000.
6. E. V. Berard, Metrics for Object-Oriented Software Engineering, USA: The Object Agency, Inc., 1998.
7. F. A. Sheikh, R. B. Mohd and H. Mohd, "A Comparative Study of Software Quality Model," International Journal of Science, Engineering and Technology Research (IJSETR), vol. 2, no. 1, pp.172-177, 2013.
8. S. Manik and S. Gurdev, "Analysis of Static and Dynamic Metrics for Productivity and Time Complexity," IJCA, vol. 30, no. 1, 2011.
9. H. Mahfuzul, A. Y.D.S. and K. M. H., "Testability Quantification Framework of Object-Oriented Software: A New Perspective," International Journal of Advanced Research in Computer and Communication Engineering, vol. 4, no. 1, pp. 289302, 2015.
10. H. Mahfuzul, A. Y.D.S. and K. M. H, "Measuring Testability of Object-Oriented Design: A Systematic Review," International Journal of Scientific Engineering and Technology (IJSET), vol. 3, no. 10, pp. 1313-1319, 2014.
11. A. Kout, F. Toure and M. Badri, "An empirical analysis of a testability model for object-oriented programs.," ACM, Inc, vol. 4, no. 36, 2011.
12. Wikipedia, "ISO/IEC 9126," Wikimedia Foundation, Inc., 5th April 2017. [Online]. Available: https://en.wikipedia.org/wiki/ISO/IEC_9126.[Accessed 16th July 2017].
13. M. Ruchika and C. Anuradha, "Application of Group Method of Data Handling model for software maintainability prediction using object-oriented systems," International Journal of System Assurance Engineering and Management, vol. 5, no. 2, p. 165–173, 2014.
14. Scalet et.al, "ISO/IEC 9126 and 14598 integration aspects," in The Second World Congress on Software Quality, Yokohama, Japan, 2000.
15. N. Fenton and M. Neil, "Software metrics: successes, failures and new directions," Journal of Systems and Software, vol. 47, no. 2-3, pp. 149-157, 1999.
16. M. Bansal and C. P. Agrawal, "Critical Analysis of Object-Oriented Metrics in Software Development," Advanced Computing & Communication Technologies (ACCT), pp. 197-201, 2014.
17. M. Aggarwal, V. K. Verma and H. V. Mishra, "An Analytical Study of Object-Oriented Metrics (ASurvey)," International Journal of Engineering Trends and Technology (IJETT), vol. 6, no. 2, pp. 76-83, 2013.
18. H. Zuse, Software Complexity Measures and Methods, Berline: Walter de Gruyter, 1991.
19. H. Zuse, "Foundations of Object-oriented Software Measures," in 3rd International Symposium onSoftware Metrics: From Measurement to Empirical Results (METRICS '96) IEEE Computer Society, Washington, DC USA, 1996.
20. S. R. Chidamber and C. F. Kemerer, "A Metrics Suite for Object Oriented Design," IEEE Transactions onSoftware Engineering, no. 20, pp. 1-5, 1994.
21. L. H. Rosenberg and L. E. Hyatt, Software Quality Metrics for Object-Oriented Environments, Greenbelt, MD 20771 USA: Goddard Space Flight Center, 1995.
22. B. F. Abreu, "Design metrics for OO software system,"inECOOP'95, Quantitative Methods Workshop, 1995.
23. V. L. V.L.Basili, L. Briand and W. L. Melo, "Avalidation of object-oriented Metrics as Quality Indicators," IEEE Transaction Software Engineering, vol. 22, no. 10, pp. 751-761, 1996.
24. R. R. Sahar and A. H. Hany, "Object oriented design metrics and tools a survey," IEEE, pp. 1-10, 2010.
25. R. Marinescu, "Measurement and Quality in Object oriented design," in In Proceedings 21st IEEE International Conference on Software Maintenance, USA, 2005.
26. J. A.-J. JUBAIR and M. S. KhairEddin, metrics for object-oriented design (MOOD) to assess java programs, Jordan: University of Jordan, 2001.
27. F. B. e Abreu, "Design Quality Metrics for Object-Oriented Software Systems," ERCIM News, 1000029 Lisboa, Portugal, 1995.
28. R. Malhotra and. M. Khanna, "Investigation ofrelationship between object-oriented metrics and change proneness," Int. J. Mach. Learn. & Cyber, vol. 10, no. 4, p. 273–286, 2012.
29. D. L. Gupta and K. Saxena, "Software bug prediction using object-oriented metrics," Sadhana-Indian Academy of Sciences, vol. 42, no. 5, p. 655–669, 2016.
30. S. R. Chidamber and C. F. Kemerer, "Towards a metric suite for object-oriented design," in OOPSLA'91 Conference Proceedings on Object-

© 2020 Global Journals

oriented Programming Systems, Languages, and Applications: ACM, New York, USA, 1991.
31. S. Chidamberand. C. Kemerer, "A Metrics Suite for Object-Oriented Design," IEEE Transactions on Software Engineering, pp. 476-492, 1994.
32. F. B. Abreu and W. Melo, "Evaluating the Impact of OO Design on Software Quality," in Third International Software Metrics Symposium, Berlin, 1996.
33. J. Mago and P. Kaur, "Analysis of Quality of the Design of the Object-Oriented Software using Fuzzy Logic," International Conference on Recent Advances and Future Trends in Information Technology, pp. 21-26, 2012.
34. B. W. Boehm, J. R. Brown and M. L. Lipow, "Quantitative Evaluation of Software Quality," in IEEE - Proceedings of the 2nd International Conference on Software Engineering, San Francisco, California, United States, 1976.
35. G. Génova, J. Llorens and J. M. Fuentes, "UML Associations: A Structural and Contextual View, "Journal of Object Technology-ETH Zurich, vol. 3, no. 7, pp. 83-100, 2004.
36. Z. Rashidi, "Properties of Relationships among objects in Object-Oriented Software Design, "AmirKabir University of Technology, Tehran, Iran, 2015.
37. P. Gandhi and P. K. Bhatia, "Optimization of Object-Oriented Design using Coupling Metrics," International Journal of Computer Applications, vol. 27, no. 10, p. 0975 – 8887, 2011.
38. S. K. Dubey, A. Sharma and A. Rana, "Comparison Study and Review on Object- Oriented Metrics," Global Journal of Computer Science and Technology, vol. 12, no. 7, pp. 1-11, 2012.
39. A. C. Shaik, B. Reddy, M. Prakashine and K. Deepti, "Metrics for object-oriented design software system: A Survey," Journal of emerging trend in engineer and applied science (JETEAS), pp. 190-198, 2010.
40. S. K. Punia, P. Kumar and A. Gupta, "A Review of Software Quality Metrics for Object-Oriented Design," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 6, no. 8, pp. 359-368, 2016.
41. T.Biggerstaff and C. Richter, "Reusability Framework Assessment, and Directions," IEEE Software, pp. 41-49, 1987.
42. V. S. Bidve and A. Khare, "Simplified Coupling Metrics for Object-Oriented Software," International Journal of Computer Science and Information Technologies (IJCSIT), vol. 3, no. 2, pp. 3839-3843, 2012.
43. A. Shaik, N. Satyanarayana, M. Huzaifa, N. Shaik,M. Z. Naveed, S. Rao and C. K. Reddy, "Investigate the Result of Object-Oriented Design Software Metrics on Fault-Proneness in Object Oriented Systems: A Case Study," Journal of Emerging Trends in Computing and Information Sciences, vol. 2, no. 4, pp. 201-209, 2011.
44. C. G. Desai, "Object Oriented Design Metrics, Frameworks and Quality Models," 27 November 2013. [Online]. Available: http://www.bioinfopublication.org/jouarchive.php?opt=&jouid=BIJ0000002. [Accessed 3rd August 2017].

© 2020 Global Journals