Volume 4, Issue 6, June 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Review on Software Quality Estimation

Gurwinder Singh
M-Tech student
mmec,mullana
ambala, India

Savita Wadhawan
Asstt.prof.
mmec,mullana
ambala, India

Abstract— Software quality is controlled by many types of uncertainties that occur during software development process which makes it difficult for the designer to evaluate the software quality. Software reuse has become a topic of much interest in the software community due to its potential benefits, which include increased product quality and decreased product cost and schedule. Although a large number of metrics have been proposed by researchers to access object-oriented design quality via reusability, they pose problems of their own, the most important being the ability to give a relevant interpretation of the measurement results. In this paper we present a brief review of the literature on software quality estimation.

Keywords— Object Oriented Design, Metrics, Software Quality, Reusability, Fuzzy Logic.

I. INTRODUCTION

It is far and wide established that object oriented development requires a dissimilar way of thinking than traditional structured development. The main advantage of object oriented design offers the advantages of modularity and reusability. Object oriented metrics are used to measure properties of object oriented design. Being a very popular development environment object oriented design improves the software productivity, reusability and flexibility of the software. Moreover, over the recent years, tit has emerged as a dominant practice in the software development. Day by day the demand for efficient software is increasing and the object oriented (OO) technique is able to satisfy this demand as it is the most powerful approach to develop efficient software. Object-oriented programming (OOP) is a programming paradigm that represents concepts as "objects" that have data fields (attributes that describe the object) and associated procedures known as methods. Objects, Which are usually instances of classes, are used to interact with one another to design applications and computer programs. Object oriented programming has many useful features, such as information hiding, encapsulation, inheritance, polymorphism and dynamic binding. These object-oriented features facilitate software reuse and component-based development. Software reusability is generally considered a way to solve the software development crisis. When we solve a problem we try to apply the solution to similar problems because that makes our work easy and simple. But one thing is for sure software reusability can improve software productivity. Software reuse has become a topic of much interest in the software community due to its potential benefits, which include increased product quality and decreased product cost and schedule. The most substantial benefits are derived from a product line approach, where a common set of reusable software assets act as a base for subsequent similar products in a given functional domain. The upfront investments required for software reuse are considerable, and need to be duly considered prior to attempting a software reuse initiative. Although a large number of metrics have been proposed by researchers to access object-oriented design quality via reusability, they pose problems of their own, the most important being the ability to give a relevant interpretation of the measurement results which in turn is due to the fact that threshold values for the metrics are difficult to set. In order to overcome this limitation, we will work on the fuzzy logic approach which, provide an integrated approach to analyze and interpret the object oriented metrics.

Object oriented design is concerned with developing an object-oriented component of a software system to apply the identified requirements. Metrics can help to evaluate and improve the design quality. Many metrics have been proposed for OOD. Metrics are a means for attaining more accurate estimations of project milestones, and developing a software system that contains minimal faults [1]. Project based metrics keep track of project maintenance, budgeting etc. Design based metrics describe the complexity, size and robustness of object oriented and keep track of design performance.

Software quality is one of the most important factors for assessing the global competitive position of any software company. We can estimate the overall design quality of the system from its design information. With OOA and object oriented design (OOD) methodologies gaining popularity, it is time to investigate OOD metrics with respect to software quality [2]. Various researchers have proposed many metric suites to evaluate the OOD quality the best out of them are the Chidamber and kemerer (CK) metric suite [3] and the Metrics for Object Oriented Design (MOOD) metric suite [4].

II. LITERATURE SURVEY

Many approaches have been developed over the years to address the problem of detecting and correcting design flaws in an object oriented (OO) software system using metrics. Moreover, with the ever increasing number of software metrics being introduced, the project managers find it hard to interpret and understand the metric scores. As Object Oriented

Metrics require very good understanding of Object-Oriented concepts and moreover, there is no single metric present which gives all features of Object-Oriented Software System.

B. W. Boehm et. al. [5] provided a well defined framework for assessing the software quality issues. The authors in this paper have proposed a conceptual framework for measuring software quality. A well defined hierarchy of the software quality characteristics is proposed. But, their framework is not complete as it has not addressed the relationship between the metrics and the software quality characteristics.

F.B. Abreu et. al. [7] is another basic structural method of the object-oriented paradigm. They were defined to measure the use of object-oriented design methods such as inheritance (MIF (Method Inheritance Factor), AIF (Attribute Inheritance Factor)) metrics, information hiding (MHF (Method Hiding Factor), AHF (Attribute Hiding Factor)) metrics, and polymorphism (POF (Polymorphism Factor), COF (Coupling Factor)) metrics. Abreu firmly suggested that metrics definitions and dimensions should be justified as they play important role in designing the object oriented metrics.

S. R. Chidamber et. al. [8] have defined six metrics viz. Weighted Methods per Class (WMC), Response sets for Class (RFC), Lack of Cohesion in Methods (LCOM), Coupling between Object Classes (CBO), Depth of Inheritance Tree of a class (DIT) and Number of Children of a class (NOC). Chidamber & Kemerer (CK) metrics measure design complexity in relation to their impact on quality attributes such as usability, maintainability, functionality, reliability etc. Several studies have been conducted to validate these metrics. They claim that using several of their metrics collectively helps managers and designers to make better design decision. Chidamber & Kemerer (CK) metrics have generated a significant amount of interest and are currently the most well known suite of measurements for object oriented (OO) software.

V. R. Basili et. al. [10] assessed the CK metrics and validated that five metrics of them appear to be useful to predict class fault proneness. They found that Chidamber & Kemerer's OO metrics were beneficial to envisage class fault-proneness during the early phases of the software life-cycle. Their empirical validation provides the practitioner with some empirical evidence which help demonstrate that most of these metrics can be useful quality indicators.

L. H. Rosenberg et. al. [11] developed nine metrics for object-oriented system, from which three were traditional metrics viz. Cyclomatic Complexity (CC), Lines of Code (LOC), Comment Percentage (CP) and rest six metrics were same as CK metrics. They validated the six Chidamber & Kemerer (CK) metrics at SATC and provided the relationship between important object oriented software quality concepts, quality metrics and object oriented features.

W. Li et. al. [12] proposed a new metric suite composed of Number of Ancestor Classes (NAC), Number of Local Methods (NLM), Class Method Complexity (CMC), Number of Descendent Classes (NDC), Coupling Through Abstract data type (CTA), and Coupling Through Message passing (CTM). These metrics are used to measure different internal attributes such as coupling, complexity and size.

R. Subramanyam et. al. [13] validated the association between Weighted Methods per Class (WMC), Coupling between Object Classes (CBO), and Depth of Inheritance Tree of a class (DIT) metrics and the fault counts, rather than fault-proneness. They analysed around 400 C++ and 300 Java classes, concluding that Chidamber & Kemerer (CK) metrics were significantly associated with faults counts, but they found that effectiveness of these metrics vary in the two programming languages investigated. While C++ classes found WMC, DIT, and interaction term (CBO*DIT) all significantly associated with faults counts, Java classes were significantly associated with faults through interaction term (CBO*DIT) only.

P. R. Srivastava et. al. [15] have proposed an approach to measure the software quality statistically. They identified that the software engineer has a responsibility to licit quality requirements that may not even be explicit at the outset and to discuss their importance and the difficulty of attaining them. They have identified the metrics and measurement approaches that can be used to assess the quality of the software product. Most of them can be measured subjectively because there are no solid statistics regarding them.

J. Mago et.al. [17] proposed a model based on Fuzzy logic to assess the quality of object oriented design, using CK metric suite and MAMDANI fuzzy inference engine.

J. S. Challa et. al. [18] made an attempt has been made by the authors to provide a tool for precisely quantifying software quality factors with the help of quality factors stated in ISO/IEC 9126 model. Due to the unpredictable nature of the software quality attributes, the fuzzy multi criteria approach has been used to evolve the quality of the software. Thus the quantification of quality parameters and integrating them into quality models is very essential.

S. K. Dubey et. al. [19] have identified the most important factors that impact the usability of object-oriented system and then they proposed a fuzzy model to determine the usability of an object-oriented software system. They used 4 input variables viz., effectiveness, efficiency, satisfaction, learn ability, on which usability of software depends. Based on expert's knowledge and experience, rule base with 81 rules for evaluating usability of object-oriented system was generated. They have concluded that the usability of object-oriented system can be improved by considering the defined characteristics and for this purpose, the developed fuzzy model will help the researchers, usability practitioners and software developers select best usable software system.

A. Handa et. al. [20] have suggested a new approach for estimating software development time. They have described the use of a set of tools called FULSOME (Fuzzy Logic for Software Metrics). They have developed a simple fuzzy logic system based on software metrics which performs acceptably when compared to regression-based models. The major difference between their work and previous works is that Two-sided Gaussian membership function in fuzzy technique is used for software development time estimation and then it's validated with the gathered data. The main benefit of this model is its good interpretability by using the fuzzy rules and another great advantage of this research is that it can put together expert knowledge (fuzzy rules) project data into one general framework that may have a wide

range of applicability in software estimation. The results showed that Two-sided Gaussian membership function is much better than other mentioned models.

K. LI et. al. [22] proposed the prediction model of software quality by fuzzy neural network based on rough set, which used the object-oriented software metric for research object. First the SOM network is adapted to discrete data. The attribute reduction algorithm based on rough set is used to reduce the matrix of metric factors. The neuron numbers of every layer in a fuzzy neural network are built according to the simplified rules from samples. After constructing the fuzzy neural network, they have used BP algorithm to train the fuzzy network. At last the model is demonstrated by experiments which show that the algorithm possesses a better performance than the fuzzy neural network, and simplifies the structure of network.

D. Gupta et. al. [23] have presented a case study of the different Software quality estimation techniques for building software quality model. They also compared various techniques like artificial neural network (ANN), Case Base Rule, Regression tree, Rule Based system, Fuzzy logic. On the basis of a comparative study Fuzzy Logic techniques offer a better Solution for designing Software quality model.

Sandhu et. al. [24] have presented a Comparative Analysis of Fuzzy, Neuro-fuzzy and Fuzzy Genetic Algorithms (GA) approaches that is performed to evaluate the reusability of software components and Fuzzy GA results outperform the other used approaches. The developed reusability model has been produced a high precision results as expected by the human experts.

Alcalá et. al. [25] have proposed a new linguistic model structure using weighted double-consequent fuzzy rules have been proposed with the aim of improving the performance of the so obtained models maintaining an acceptable description level. Its main interest lies in flexibilyzing the model structure in a different way from the usual one.

III. PROPOSED WORK

On the basis of the literature survey conclusions were drawn and the gap was analyzed one of which is that researchers criticized about the Chidamber & Kemerer (CK) metric suite is inadequate to evaluate external design characteristics. Experiments conducted by researchers have predicted that the Depth of inheritance (DIT) and Number of Children (NOC) metrics had a success rate of only 28% in predicting design complexity more over the relationship between software attributes and metrics has not been effectively expressed. The traditional models lack the ability to process data both qualitatively and quantitatively and were not appropriate to the case with uncompleted information. The traditional models do not provide a hybrid mechanism to interpret the metrics together. Researchers have analyzed Chidamber & Kemerer (CK) metrics and have found that these metrics address the class structures, their attributes and methods, but unable to assess the design properly and the resulting final system, which is the major requirement of the managers. Additionally, For creating an Object Oriented Software, a large number of rules are applied. These rules are typically formulated in two-valued logic. Two value logic suffers from following problems:

- The two valued logic method is unable to completely capture the developer's intuition and the software engineer's perception.
- It does not model explicitly the various contextual factors that can affect the validity of rules, e.g. Application domain changes in user's interest etc.
- The two valued logic lacks expressive ability.

IV. CONCLUSION

Software reuse has become a topic of much interest in the software community due to its potential benefits, which include increased product quality and decreased product cost and schedule. The most substantial benefits are derived from a product line approach, where a common set of reusable software assets act as a base for subsequent similar products in a given functional domain. The upfront investments required for software reuse are considerable, and need to be duly considered prior to attempting a software reuse initiative.

There are a lot of problems is faced by researchers in regards to access the object oriented (OO) design quality by reusability, but the present study at hand will work on fuzzy logic approach to analysis and interpret the reusability of object oriented Design.

REFERENCES

[1] E. Chandra and P. E. Linda, Assessment of Software Quality through Object Oriented Metrics, CIIT International Journal of Software Engineering, Vol. 2, Issue: 2, Feb 2010.

[2] M. Lorenz and J. Kidd, Object Oriented Software Metrics, Prentice Hall Publishing, 1994.

[3] H. Rathod and M. Parmar, Study of Fuzzy Logic Approach in Estimating Reliability of Component Based Software, International Journal of Global Research Analysis, Vol. 1, Issue: 5, pp. 21-23, ISSN No 2277 – 8160, Nov 2012.

[4] F. B. Abreu and W. Melo, Evaluating the Impact of Object Oriented Design on Software Quality, Proceedings of METRICS, 1996.

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

[6] A. Kaur, S. Singh, K. S. Kahlon and P. S. Sandhu, Empirical Analysis of CK & MOOD Metric Suit, International Journal of Innovation, Management and Technology, Vol. 1, No. 5, December 2010.

[7] F. B. Abreu and R. Carapua, Candidate Metric for OOS within Taxonomy Framework, Journal of System & Software, Vol. 26, No. 1, July 1994.

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

[9] H. Rathod and M. Parmar, Study of Fuzzy Logic Approach in Estimating Reliability of Component Based Software, International Journal of Global Research Analysis, Vol. 1, Issue: 5, pp. 21-23, ISSN No 2277 – 8160, Nov 2012.

[10] V. R. Basili, L. C. Briand, and W.L. Melo, A Validation of Object Oriented Design Metrics as Quality Indicators, IEEE Transactions on Software Engineering, Vol. 22, No. 10, pp. 751-761, 1996.

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

[12] W. Li, Another Metric Suite for Object Oriented Programming, Journal of Systems and Software, Vol. 44, No. 2, pp. 155-162, 1998.

[13] R. Subramanyam and M. S. Krishnan, Empirical analysis of metrics for object-oriented design complexity: Implications for software defects, IEEE Transaction on Software Engineering, 29(4), pp. 297-310, 2003.

[14] R. Malhotra, A.Kaur and Y.singh, Empirical validation of object-oriented metrics for predicting fault proneness at different severity levels using support vector machines, International Journal of System Assurance Engineering and Management, Vol. 1, Issue: 3, pp. 269-281, September 2010.

[15] P. R. Srivastava and K. Kumar, An Approach towards Software Quality Assessment, Communications in Computer and Information Systems Series (CCIS Springer Verlag), Vol. 31, No. 6, pp. 345-346, 2009.

[16] R. Harrison, L. G. Samaraweera, M.R. Dobie, and P. H. Lewis, Comparing Programming Paradigms: An Evaluation of Functional and Object-Oriented Programs, Journal of Software Engineering, Vol. 11, Issue: 4, pp. 247-254, July 1996.

[17] J. Mago and P. Kaur, Analysis of Quality of the Design of the Object Oriented Software using Fuzzy Logic, IJCA Proceedings of International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT 2012), pp. 21-25, April 2012.

[18] J. S. Challa, A. Paul, Y. Dada, V. Nerella, P. R. Srivastava and A. P. Singh, Integrated Software Quality Evaluation: A Fuzzy Multi-Criteria Approach, Journal of Information Processing Systems, Vol. 7, No. 3, 2011.

[19] S. K. Dubey, A. Rana and A. Sharma, Usability Evaluation of Object Oriented Software System using Fuzzy Logic Approach, International Journal of Computer Applications, 43(19), pp. 1-6, April 2012.

[20] A.Handa and G. Wayal, Software Quality Enhancement using Fuzzy Logic with Object Oriented Metrics in design, International Journal of Computer Engineering & Technology (IJCET), Vol. 3, No. 1, pp. 169-179, June 2012.

[21] P. Saxena and M. Saini, Empirical Studies to Predict Fault Proneness: A Review, Proceedings International Journal of Computer Applications, 22(8), pp. 41–45, May 2011.

[22] K. LI, L. Gong and J. Kou, Predicting Software Quality by Fuzzy Neural Network Based on Rough Set, Journal of Computational Information Systems, Vol. 6, pp. 1439-1448, May 2010.

[23] D. Gupta, V. K. Goyal and H. Mittal, Comparative Study of Soft Computing Techniques for Software Quality Model, International Journal of Software Engineering Research & Practices, Vol.1, Issue: 1, Jan 2011.

[24] Sandhu, Parvinder Singh, Dalwinder Singh Salaria, and Hardeep Singh, A comparative analysis of fuzzy, neuro-fuzzy and fuzzy-GA based approaches for software reusability evaluation, Proceedings of Worlds Academy of Science, Engineering and Technology, Vol. 39, pp. 342-345, 2008.

[25] Alcalá, Rafael, Jorge Casillas, Oscar Cordón, and Francisco Herrera, Linguistic modeling with weighted double-consequent fuzzy rules based on cooperative co-evolutionary learning, Integrated Computer-Aided Engineering 10, no. 4 (2003): pp. 343-355.

