Analysis of Quality of the Design of the Object Oriented Software using Fuzzy Logic

Jagmohan Mago
Assistant Professor
Ape jay College of Fine Arts,Jalandhar

Parwinder Kaur
Student,
M. Tech (Computer Science)
GNDU, Amritsar

ABSTRACT
Many of the software fail due to poor quality. Estimating software quality is an important task in the software development. The development of large software system is a time consuming and resource consuming activity. Software metrics are necessary to identify where the resources are needed; they are a crucial source of information for decision making. To produce high quality object oriented applications, a strong emphasis on design aspects, especially during the early phases of software development is necessary. Design metrics play an important role in helping developers to appreciate design aspects of software i.e. improve software quality. By analyzing the metric data we can forecast the quality of the object oriented system. In this paper we propose a model based on fuzzy logic which serves as an integrated means to provide an interpretation of the OOD metrics of the CK metric suite.

General Terms
Software Engineering and Artificial Intelligence.

Keywords
OOD, Metrics, Quality, Fuzzy Logic, CK Metrics.

1. INTRODUCTION
Object Oriented Design and Development is an interesting area of current research and many authors have done great deal of work in recent years. In fact Object Oriented Development requires not only a different approach to design and implementation, but also a different approach to software metrics. To produce high quality Object Oriented applications a strong emphasis on design aspects is highly necessary [9]. Software metrics 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 is the degree to which software possesses a desired combination of attributes such as reliability, maintainability, efficiency, portability, usability and reusability. A software quality estimation model allows the software development team to track & detect potential software defects. Such quality models will also help developers in building better quality programs. A number of well-known quality models are used to build quality software in industry.

Object oriented design is intended to capture the fundamental structure of an object oriented program. Thus, a set of components which can help to evaluate, represent and implement an object oriented design should include attributes, methods, objects (classes), relationships and class hierarchies. Software quality must be addressed during the whole process of software development. Measuring software quality in the early stages of software development is the key to develop high quality software. Product quality has some attributes such as functionality, effectiveness, understandability, reusability and maintainability. A large number of software metrics have been proposed in software engineering to measure the quality attributes of the software in early stages. We can estimate the overall design quality of the system from design information. With OOA and OOD methodologies gaining popularity, it is time to investigate OOD metrics with respect to software quality. Hence, there must be a way to assess object oriented software quality as early as possible in the development life cycle.

We can estimate the overall design quality of the system from its design information. With OOA and OOD methodologies gaining popularity, it is time to investigate OOD metrics with respect to software quality [6]. Although various researchers have proposed many metric suites to evaluate the OOD quality the best out of them is the CK metric suite. None of the metric alone can reflect the quality of design. Hence, some integrated means is required to combine them into a single output.

This paper proposes a model based on Fuzzy Logic to assess the quality of OO design, uses the CK metric suite and the MAMDANI fuzzy inference engine.

2. LITERATURE REVIEW
Over the past years, with the advent of new methodologies, process driven management many approaches have been developed to address the problem of detecting and correcting design flaws in an 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.

Chidamber and Kemerer are the predominantly referenced researchers, they proposed 6 metrics-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 (DIT), Number of Children of a class (NOC), with the help of which various software quality attributes (e.g. efficiency, complexity, understandability, reusability, maintainability and testability) can be measured. They claim that using several of their metrics collectively can help project managers and designers make better design decisions.

MOOD metric set model, proposed by Abreu [12] 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.

Within the framework that, many metrics that are applied to traditional functional development are also applicable to object-oriented development, Rosenberg et al. [12] 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 CK metrics at SATC and gave the relation between important object oriented software quality concepts, quality metrics and object oriented features as shown in Table 2 [12].

Amjan Shaik et. al. in [9] performed statistical analysis on the CK metric suite for Object oriented systems. They found that if properly used, metrics could lead to a significant reduction in cost of the overall implementation and quality improvement. L. Rosenberg et al. in [2] have identified five attributes for analysing design and code of the software. These are efficiency, complexity, understandability, reusability and testability/maintainability. Dr. Thapalyal, G. Verma in [15] performed an empirical study of two metrics – CBO,WMC of CK metric suite to extract the relationship of these metrics with defects.

3. SOFTWARE QUALITY FACTORS

In reality object oriented development has proved its value for systems that must be maintained and modified. The concepts of object oriented design metrics are well established and many metrics relating to product quality have been developed and used. With object oriented analysis and design methodologies gaining popularity, it is time to start investigating object oriented design metrics with respect to software quality. Measuring quality in the early stage of software development is the key to develop high quality software. There must be a way to assess object oriented software quality as early as possible in the development cycle [3]. The factors (McCall, 1977) that affect software quality can be categorized in two broad groups:

1. Factors that can be directly measured (e.g. defects recovered during testing) and

2. Factors that can be measured only indirectly (e.g. usability or maintainability)

McCall proposed a useful categorization of factors that affect software quality as shown in figure 1.

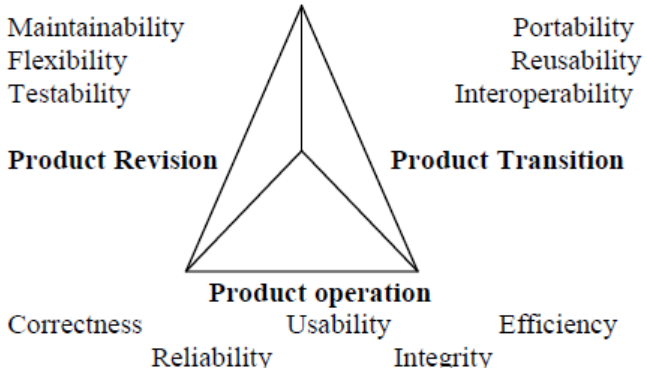


Figure 1 McCall's Quality factors [Source Pressman Fifth Edition]

4. OBJECT ORIENTED METRICS

Chidamber and Kemerer (CK) [1] are the mostly referenced researchers. They 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). CK metrics were defined to 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 CK metrics.

Table 1 The Metric Suite of Chidamber and Kemerer [1]

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 metrics are aimed at assessing the design of object oriented system rather than implementation. This make them more suited to object-oriented paradigm as object-oriented design put great emphasis on the design phase of software system. The relation between important object-oriented software quality concepts, CK metrics and object-oriented (OO) features is given in Table 2 [2].

Table 2 Relationship among CK metrics, OO software Quality Concepts and OO Features [2]

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

5. PROPOSED MODEL

All of the six metrics measures the different aspects of software quality and reflects the different properties of software quality. Use of the CK metric suite and other measures is growing gradually in the software industry. This is reflected in the increasing number of industrial software tools, such as Rational Rose, JHawk that enables automated computation of these metrics.

None of the metric alone can reflect the reliable quality of the software therefore some integrated means are required to combine them into a single output. The model to predict the quality of design of the software is built using the fuzzy logic approach shown in figure which uses the Mamdani Model as the rule based inference engine.

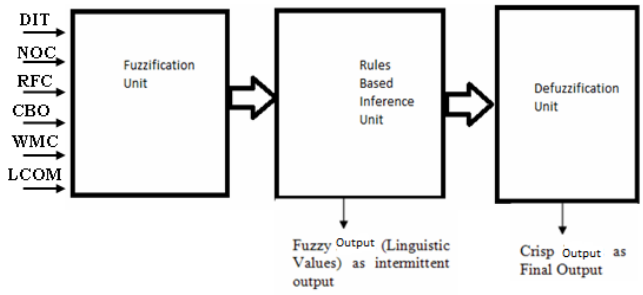


Figure 2: Proposed Fuzzy Logic based Framework

Fuzzy systems try to behave just like the processes of the brain with a rule base. The basic concept is inspired by the human processes where the decisional criteria are not clear cut, but blurred and it is difficult to find objective to make the decisions more precise and clear. Use of fuzzy sets in logical expression is known as fuzzy logic. Fuzzy logic as the name describes, the mode of reasoning to be approximate rather than exact. The use of a fuzzy decision process allows the simultaneous validation of all the rules; each input value is considered to be both above and below the corresponding threshold, but with gradual and typically different certainties. The end result will be the outcome of combining all of the partial results, each contributing its weight to the decision process.

A fuzzy subset F of a set S can be defined as a set of ordered pairs, each with a first element that is an element of the set S, and a second element that is a value in the interval [0, 1], with exactly one ordered pair present for each element of S. This defines a mapping between elements of the set S and values in the interval [0, 1]. The value zero is used to represent complete non-membership, the value one is used to represent complete membership, and values in between are used to represent intermediate degrees of membership. Fuzzy model consist of four modules. The first module is the fuzzification that transforms the crisp value(s) into the fuzzy values.[8] The fuzzy values are inferences based upon the rule base incorporate in knowledge based. These rules are supplied by the domain expert(s). All the outputs obtained from the inference engine are integrated and defuzzied by the defuzzification module that transform the fuzzy output to crisp value.

5.1 Working Of The Model

First all, the crisp values of four inputs are taken and degree to which each belongs to the membership function is determined [8]. A membership function (MF) is a curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1. The 0 value represents the completely out membership; 1 value represents the completely in membership; and all other values between 0 and 1 represents the intermediate membership. The input space is sometimes referred to as the universe of discourse. Depending upon the input values some rules from the knowledge based gets executed. All the inputs are considered parallel and combined with AND operator. The MIN/MAX membership operator is used to find out degree of membership of firing. The Mamdani inference engine is used to determine the degree of membership of firing. The technique used to defuzzification is the centroid that transforms the fuzzy values to the Linguistic variable.

5.2 Proposed Rules

In this model six input metrics are considered with each metric being defined by 3 membership functions viz, low, medium and high therefore rule base consists of all the possible combinations i.e. 36 = 729 rules. As low values of the Shyam Chidamber, and Chris Kemerer(C.K) Metrics are the most preferred ones, so we assign maximum weight to LOW and successively to MEDIUM and HIGH values. ASSIGNING: L=3 M=2 H=1

Adding up all the six values of Shyam Chidamber, and Chris Kemerer(C.K) Metrics. We divide the sum by six to get the average value.

Output weights from assigned inputs

$$O[w]=\frac{\text{Values [CBO+RFC+WMC+DIT+NOC+LCOM]}}{6}$$

Now comparing O[w] with A (set) we may virtually qualify the quality of software as:HIGH, MEDIUM, LOW.

Table 3 Rule Listing

RULES / CK Parameter	21	22	23	24	25	26
CBO	**L**	**L**	**L**	**L**	**L**	**L**
RFC	**L**	**L**	**L**	**L**	**L**	**L**
WMC	**L**	**L**	**L**	**L**	**L**	**L**
DIT	**H**	**H**	**H**	**H**	**H**	**H**
NOC	**L**	**M**	**M**	**M**	**H**	**H**
LCOM	**H**	**L**	**M**	**H**	**L**	**M**
QUALITY OF SOTWARE	**M**	**L**	**M**	**M**	**L**	**M**

Let us consider rule number 21 for explanation. If values of our selected parameters are as:

CBO=L

RFC=L

WMC=L

DIT=H

NOC=L

LCOM=H

Now working upon the above defined formula we get:

O [w] = Values [CBO+RFC+WMC+DIT+NOC+LCOM]/6

O [w] = [L+L+L+H+L+H]/6

O [w] =[3+3+3+1+3+1]/6

O [w] =14/6

O [w] =2.3333

Thus quality of software as per rule number 21 is medium.

5.3. Membership Functions for Input Metrics

The curves for the membership functions are defined as follows using the CK metric thresholds as shown in table below. The input metrics CBO, DIT, LCOM, NOC, WMC, RFC are divided into three stages Low, Medium and High.

Table 4 Threshold values for the CK metric Suite [4,5]

S.No	Metrics	Threshold value
1	WMC	0-15
2	DIT	0-6
3	NOC	0-6
4	CBO	0-8
5	RFC	0-35
6	LCOM	0-1

The type of membership function used in all the inputs is triangular membership function. This function has necessarily three parameters [a, b, c].

Triangular(x: a, b, c) =

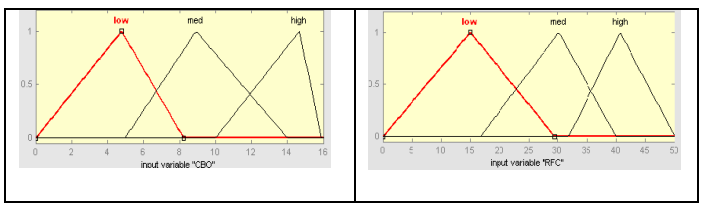
$$\begin{cases} 0 & x < a \\ (x-a)/(b-a) & a \le x \le b \\ (C-x)/(c-b) & b < x \le c \\ 0 & x \ge c \end{cases}$$

For instance, for DIT we take the broad range of 1-10. The three possible values are for low, medium and high are:-

Low= (0, 3.334, 5.751)

Medium= (3.45, 5.604, 7.5)

High= (6.1, 7.939, 10)



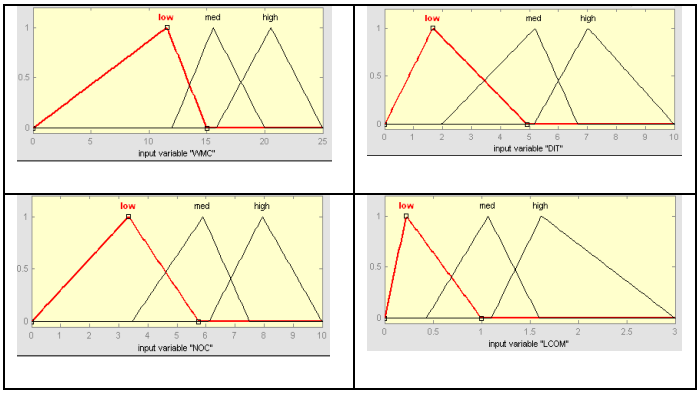


Figure 3 Membership functions for CBO,RFC,WMC,DIT,NOC,LCOM

5.4. Membership Function Assignment

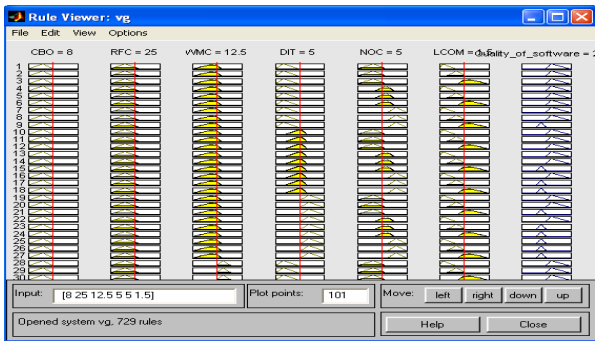
The following example illustrates how the membership grade is assigned to T. The inputs are fed to the fuzzification module and after Fuzzification of given values we find that CBO=3 belongs to fuzzy set LOW, WMC = 11 belongs to fuzzy set LOW, DIT = 4 belongs to fuzzy set LOW, NOC=3 belongs to fuzzy set LOW, LCOM=0 belongs to fuzzy set LOW and RFC = 12 belongs to the fuzzy set LOW. With these input values the following rules get fired.

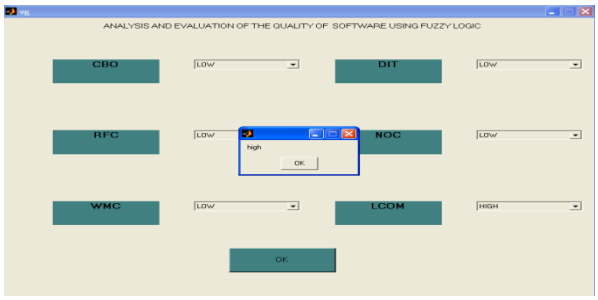
CBO (3)	WMC (11)	RFC (12)	NOC (3)	DIT (4)	LCOM (0)	Membership grade of T
LOW	LOW	LOW	LOW	LOW	LOW	MIN

The rule gives the output value as High which indicates the high software design quality.

6. DEFUZZIFICATION

The model is built using the Matlab tool for the fuzzy logic. The centroid technique of defuzzification is used. T is calculated from the MAMDANI model and the output value is calculated from the MAMDANI fuzzy model.





7. CONCLUSION

In this paper we have proposed a software quality model based on fuzzy logic for carrying out the quality estimation early at design phase of SDLC. In future, we can plan an extensive validation of our metric suite on a variety of different industrial environments and collect data to analyze the reliability of the proposed model. We can refine the metric suite by adding and deleting some of the metrics as per industry requirement. The user interface can be made more users friendly.

8. REFERENCES

[1] 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.

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

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

[4] Y. Zhou and H. Leung, 2006. Empirical Analysis of Object-Oriented Design Metrics for Predicting High and Low Severity Faults, IEEE Transactions On Software Engineering, Vol. 32, No. 10.

[5] 2010. Assessment of Software Quality through Object Oriented Metrics, CIIT International Journal of Software Engineering, Vol2.

[6] Fuzzy Logic: Theory and Applications. Prentice Hall of India, New Delhi.

[7] N. Kaur, M. Singh, 2011. A fuzzy logic approach to Measure the precise Testability index of software, Vol. 3 No. 2, pp. 857-865.

[8] A. Shaik, C. R. K. Reddy, B. Manda, Prakashini. C, Deepthi. K, 2010. An Empirical Validation of Object Oriented Design Metrics in Object Oriented Systems, Journal of Emerging Trends in Engineering and Applied Sciences, pp. 216-224.

[9] 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.

[10] McCall, J. A., Richards, P. K., and Walters, G. F., 1977. Factors in Software Quality. Griffiths Air Force Base, N.Y. Rome Air Development Center Air Force Systems Command.

[11] F. B. Abreu and R. Carapua, 1994. Candidate Metric for OOS within taxonomy framework, Journal of System & Softwrae, Vol. 26, No. 1.

[12] 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.

[13] C. Serban and H. F. Pop, 2008. Software Quality assessment using a fuzzy clustering approach, INFORMATICA, Vol. LIII.

[14] Dr. M. P. Thapaliyal, G. Verma, 2010. Software defects and object oriented metrics – An Empirical Analysis, International Journal of Computer Applications, Vol. 9.

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