International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 6, Issue 4, July- August 2017 ISSN 2278-6856

Empirical validation of MOOD metrics to predict Software Reuse

Parwinder Kaur Dhillon[1] , Pooja Dhand[2]

[1] Assistant Professor, Dev Samaj College for Women, Sector 45B, Chandigarh

[2]Assistant Professor, Apeejay College of Fine Arts, Jalandhar

Abstract

Validation of object oriented metrics is extremely important to ensure their relevance in the software industry. This paper finds the correlation of object-oriented metrics proposed by Abreu et al. with the degree of reuse of the design. Predicting the reusability in object oriented systems not only reduces the overall cost but also improves the overall quality of the software. The six merits of MOOD suite cover the different OO aspects viz Inheritance, coupling, encapsulation, and polymorphism. However, even if the designer has calculated these metrics, he is unable to derive a proper conclusion about the quality of design. This paper presents a fuzzy logic based model to analyze and interpret MOOD metrics for OOD quality based on reusability. Performance results prove that the MOOD metrics can be used to obtain an overall assessment of the reusability of software system. The proposed model will serve as an important decision making tool for the managers. Based on the results, it can be concluded that the proposed model could help for planning and focusing on those modules during the design and code that are likely to be reused in future.

Keywords: MOOD metrics, Reuse, Fuzzy Logic, OOD, software quality

1. INTRODUCTION

Object oriented design is becoming more popular in a software development environment. The primary benefit of object oriented design is its features of modularity and reusability. To quantify the attributes of object oriented design object oriented metrics are being utilized. These object oriented design metrics are an essential part of software development environment. The metrics for object oriented design focus on measurements that are applied to the class and design characteristics. These measurements allow the designers to access the software early during the development process, making changes that will reduce complexity and improve the continuing capability of the design [10].The object oriented approach offers an effective reuse of the software components. The process of reuse accelerates the software development and thereby resulting in high quality work in minimal time and lesser development effort [11]. OOD is easier to interpret, adapt and scale due to its modular structure, low coupling and high cohesion. But simply by applying object oriented programming doesn't give good results. The software developer needs to apply OO metrics to assess the quality of the design.

In this paper, we have developed a fuzzy logic based model to assess the degree of reuse of the OOD by analyzing the MOOD metrics suite. The paper is organized as follows: section 1 gives introduction to the problem, section 2 reviews the available literature for using OO metrics as quality predictors, section 3 presents the description of the empirical study viz. the data sets used, OO metrics used etc.; section 4 presents the system model in detail; section 5 discusses the performance results and finally section 6 concludes with the future enhancements that can be made to the proposed model.

2. LITERATURE REVIEW

In [6] the authors presented an empirical validation of software metrics suites to predict fault proneness in object-oriented (OO) components. They validated three OO metrics suites for to predict software quality in terms of fault-proneness: the Chidamber and Kemerer metrics, Abreu's Metrics, and Bansiya and Davis' Quality Metrics for Object-Oriented Design. They concluded that the class components in the MOOD metrics suite are not good class fault-proneness predictors.

In [7] a survey of three object oriented software metrics suite comprising of CK Suite, MOOD Suite and Lorenz and Kidd Suite was made. They suggest that MOOD suite is well defined, project level based, mathematically computable and provides thresholds that could be used to judge the metrics collected from a given design.

A systematic review of fault prediction using OO metrics was performed by Catal and Diri [8]. They included all papers related to software fault prediction and classified the studies with respect to metrics, methods and data sets.
In [11] the authors presented a case study of applying design measures to assess software quality. They analyzed six Java based open source software systems using a CK metrics suite to find out the quality of the system and possible design faults that will reversely affect different quality parameters such as reusability, understandability, testability, maintainability. They also offered a set of general guidelines for interpretation of reusability, understandability, testability, maintainability.

S.K Dubey in [12] provides a review of the OO metrics proposed in literature and the related state-of-the-art. They have also proposed a maintainability model based on the analysis of the relationship between object-oriented metrics and maintainability.

J. Mago in [16] proposed a model based on fuzzy logic as an integrated means to provide an interpretation of the OOD metrics of the CK metric suite.

R. Vir in [17] proposed a hybrid approach for investigating the extendibility/extensibility of classes in Object Oriented design. The hybrid approach will comprised of subset of CK metric suite and mood metric suite.

3. DESCRIPTION OF EMPIRICAL STUDY

3.1 Data Set

We have used two projects developed in VB.NET for this study. Both of them were developed by students doing their graduation degree. The requirements were clearly specified to them. The students considered for this project were beginners in object oriented systems development and did not have prior experience in developing object oriented systems. The first project is “Library Management System” for a college. The requirement specification was obtained from the existing Library management software and was used for this project. The next project is a “Hospital Management System”. Both the projects were monitored for their use of object oriented features in the design phase to avoid a procedural oriented approach.

3.2 Object Oriented Metrics Considered

Abreu et al [1] defined a set of six metrics to quantitatively measure the OO features referred to as MOOD (Metrics for Object Oriented Design) metrics. The set refers to all the basic structural aspects of the OO approach such as encapsulation (MHF, AHF), inheritance (MIF, AIF), polymorphism (POF), and message passing (COF). Each metric is a measure where the numerator represents the actual use of one of those features for a given design [4]. Two key features are used in every metric; they are methods and attributes.

3.2.1Attribute Hiding Factor (AHF)

The Attribute Hiding Factor measures the degree to which an attribute is invisible in a class. An attribute is visible if it can be accessed by another class or object. For a good design attributes should be "hidden" within a class by being declared as private [5, 6, and 14]. It is advantageous for the Attribute Hiding Factor to have a large value.

3.2.2Method Hiding Factor (MHF)

The Method Hiding Factor measures the degree to which methods in classes are invisible. For a good design methods must be encapsulated inside a class and not accessible for use to other objects. Method hiding increases reusability in other applications and decreases design complexity [5, 6, and 14]. A large value of Method Hiding Factor is desirable.

3.2.3Method Inheritance Factor (MIF)

The Method Inheritance Factor measures the degree to which the class architecture of an object oriented system makes use of inheritance for both methods and attributes [5, 6, and 14].

3.2.4Attribute Inheritance Factor (AIF)

AIF is defined as the ratio of the sum of inherited attributes in all classes in the system. AIF denominator is the total number of available attributes for all classes. It is defined in an analogous manner and provides an indication of the impact of inheritance in the object oriented software [5,6,14].

3.2.5Polymorphism Factor (POF)

Polymorphism Factor measures the degree of method overriding in the class inheritance tree [5,6,14]. PF is an indirect measure of the relative amount of dynamic binding in a system.

3.2.6Coupling Factor (CF)

The COF is defined as the ratio of the maximum possible number of couplings in the system to the actual number of coupling is not imputable to inheritance [5,6,14]. Research suggests that increased CF increases defect density and rework effort to detect and correct faults.

3.3 Data Collection Procedure

The metrics were calculated using the Project Analyzer v10.1 tool. Project Analyzer is a Visual Basic code review and quality control tool.

4. SYSTEM MODEL

In the proposed model, we have developed three different cascaded fuzzy logic controllers to estimate the degree of reuse of the OO software. In order to reduce the number of rules and system complexity, we have developed three different fuzzy controllers. The outputs of FLC-1 and FLC-2 are used as input to the FLC-3 which computes the final degree of reuse. The Mamdani type FIS is incorporated with carefully designed rules. Figure. 1 shows the overall design of these FLCs that are used to determine the reusability. In the following subsections, the details of all the fuzzy logic controllers are provided.

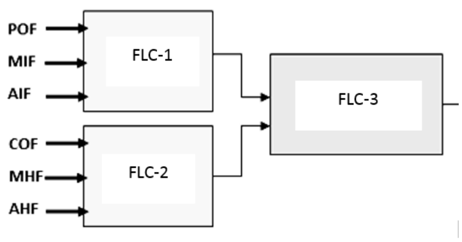

Figure 1 Fuzzy Logic based System Model

4.1 Design of Fuzzy Logic Controller 1

Fuzzy logic controller 1 computes the degree of reuse from three of the MOOD metrics (POF, MIF and AIF). As the value of these metrics increases, the reuse factor also increases. The inference model applied is MAMDANI. The knowledge base consists of a total of 27 rules. The three inputs are fed into the fuzzy systems. Depending on the input values of the metric, some rules out of the total 27 rules from the knowledge base gets fired. The technique used for defuzzification is 'Centroid of area' method. Figure 2 below show the design for FLC-1 and the membership function for the input variables. The input variable has three membership functions (Low, Medium, and High) and the output variable also has three singleton membership functions (Low, Medium, and High). The output variable reflects the probability of reuse of the class.

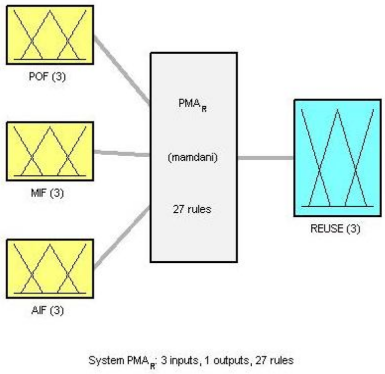

Figure 2 System model for FLC-1

4.1.1Membership Functions

A fuzzy set is completely characterized by its membership function (MF). A more convenient and concise way to define an MF is to express it as a mathematical formula.

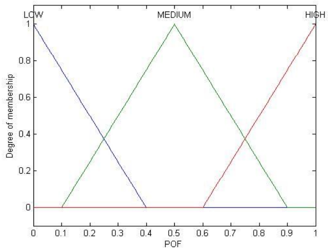

Figure 3 Input membership function for POF

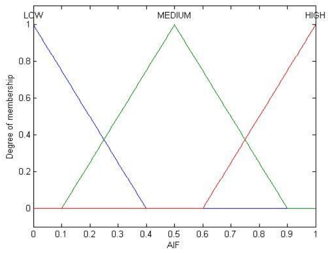

Figure 4 Input membership function for AIF

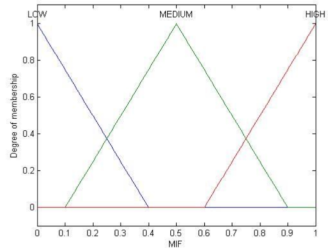

Figure 5 Input membership function for MIF

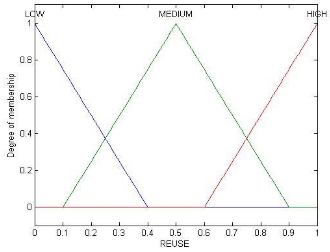

Figure 6 Output membership function for FLC-1

4.1.2Design of the Knowledge Base

We are using a model in which there are 3 inputs and each input is having three membership functions. By using all the possible combinations, the size of the rule base comes out to be 3*3*3=27 rules. The rules are implemented using MATLAB. Inference process used in this model is Mamdani. Rule Viewer is shown as below in Table 1.

Table 1 Knowledge Base for FLC-1

Rule No.	POF	MIF	AIF	Reuse
1	Low	Low	Low	High
2	Low	Low	Medium	High
3	Low	Low	High	High
4	Low	Medium	Low	High
5	Low	Medium	Medium	Medium
6	Low	Medium	High	Medium
7	Low	High	Low	High
8	Low	High	Medium	Medium
9	Low	High	High	Low
10	Medium	Low	Low	High
11	Medium	Low	Medium	Medium
12	Medium	Low	High	Medium
13	Medium	Medium	Low	Medium
14	Medium	Medium	Medium	Medium
15	Medium	Medium	High	Medium
16	Medium	High	Low	Medium
17	Medium	High	Medium	Medium
18	Medium	High	High	Low
19	High	Low	Low	High
20	High	Low	Medium	Medium
21	High	Low	High	Low
22	High	Medium	Low	Medium
23	High	Medium	Medium	Medium
24	High	Medium	High	Low
25	High	High	Low	Low
26	High	High	Medium	Low
27	High	High	High	Low

4.2 Design of Fuzzy controller 2

A fuzzy logic model is designed to predict the reuse factor of three of the MOOD metrics (COF, MHF and AHF) as these three metrics of the MOOD metric suite are shown to be very good predictors of the reuse proneness of Object Oriented Design. As the value of these metrics increases, the reuse factor decreases. To get the reuse factor, MAMDANI fuzzy inference model is applied. The three inputs are fed into the fuzzy systems. Depending on the input values of the metric, some rules out of the total 27 rules from the knowledge base gets fired. The Mamdani inference engine is used to ascertain the level of membership of firing. The technique used for defuzzification is Centroid method. Figure 7 below show the design for FLC-2 and the membership function for the input variables. The input variable has three membership functions (Low, Medium, and High) and each of the three output variables has three singleton membership functions (Low, Medium, and High). The output variable reflects the probability of reuse of the class.

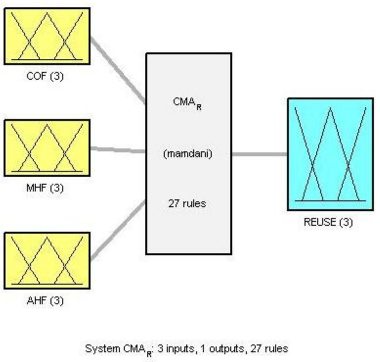

Figure 7 System model for FLC-2

4.2.1Membership Functions for Input Metrics

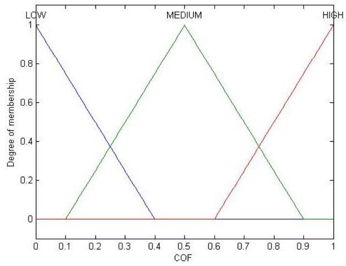

Figure 8 Input membership function for COF

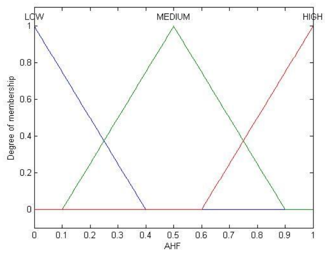

Figure 9 Input membership function for AHF

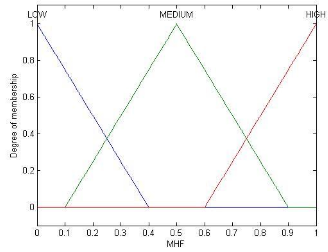

Figure 10 Input membership function for MHF

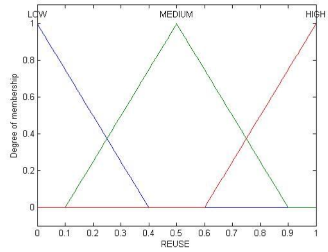

Figure 11 Output membership function for Reuse factor

4.2.2Design of the Knowledge Base

FLC-2 uses 3 inputs and each input is having three membership functions. Hence, the size of the rule base comes out to be 3*3*3=27 rules. Inference process used in this model is Mamdani. Rules are shown in table 2 below.

Table 2 Knowledge Base for FLC-2

Rule No.	COF	MHF	AHF	Reuse
1	Low	Low	Low	Low
2	Low	Low	Medium	Low
3	Low	Low	High	Low
4	Low	Medium	Low	Low
5	Low	Medium	Medium	Medium
6	Low	Medium	High	Medium
7	Low	High	Low	Low
8	Low	High	Medium	Medium
9	Low	High	High	Low
10	Medium	Low	Low	Low
11	Medium	Low	Medium	Medium
12	Medium	Low	High	Medium
13	Medium	Medium	Low	Medium
14	Medium	Medium	Medium	Medium
15	Medium	Medium	High	Medium
16	Medium	High	Low	Medium
17	Medium	High	Medium	Medium
18	Medium	High	High	High
19	High	Low	Low	Low
20	High	Low	Medium	Medium
21	High	Low	High	High
22	High	Medium	Low	Medium
23	High	Medium	Medium	Medium
24	High	Medium	High	High
25	High	High	Low	High
26	High	High	Medium	High
27	High	High	high	High

4.3 Design of Fuzzy Logic Controller 3

Fuzzy logic controller 3 takes the output of both FLC1 and FLC2. The main purpose of using this cascaded fuzzy logic controllers is to reduce the number of rules. The inference model used is Mamdani. The output variable gives the overall degree of reuse of the design. The value of the reuse varies between 0-1.

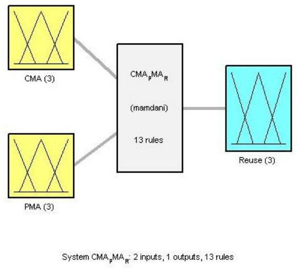

Figure 12 System model for FLC-3

4.3.1 Design of the Input and Output Membership function

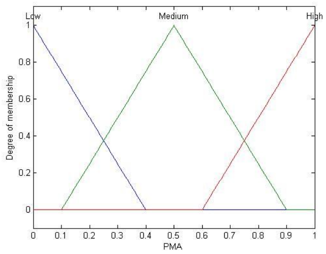

Figure 13 Input membership function for PMA

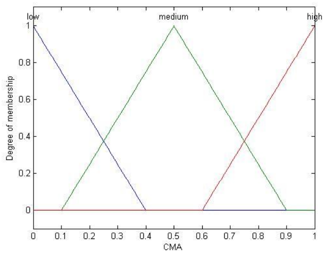

Figure 14 Input membership function for CMA

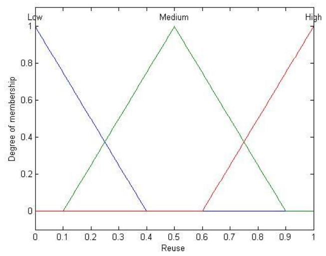

Figure 15 Output membership function for Reuse

4.3.2Design of the Knowledge Base

FLC-3 takes 2 inputs from the FLC-1 and FLC-2 each input is having three membership functions. There are total 13 rules. Rules are shown below in Table 3.

Table 3 Knowledge Base for FLC-3

Rule No.	FLC1	FLC2	Reuse
1	Low	Low	Low
2	Low	Medium	Low
3	Low	Medium	Medium
4	Low	High	Medium
5	Medium	Low	Low
6	Medium	Low	Medium
7	Medium	Medium	Medium
8	Medium	High	Medium
9	Medium	High	High
10	High	Low	Medium
11	High	Medium	Medium
12	High	Medium	High
13	high	High	High

5. RESULTS

Polymorphism arises from inheritance and suggest that in some cases overriding methods could contribute to reduce complexity and therefore to make the system more understandable and easier to maintain. While, research has shown that this metric is a valid measure within the context of the reuse. The graph in figure 16 below shows the relationship between input variable 'POF' and output variable 'Reuse'. It shows that as the value of POF increases the reuse factor of the class also increases.

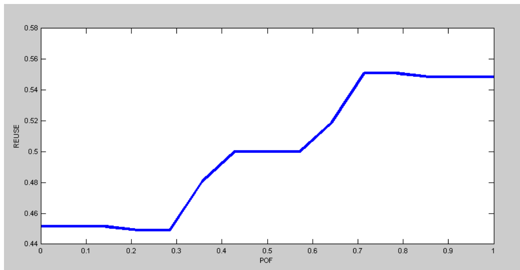

Figure 16 Relationship between *Reuse* and *POF*

MIF should not be too low and not too high either. Too high a value indicates either unnecessary inheritance. A low value indicates lack of inheritance. The graph in figure 17 shows the relationship between input variable 'MIF' and output variable 'Reuse'. It shows that as the value of MIF increases the reuse factor of the class also increases.

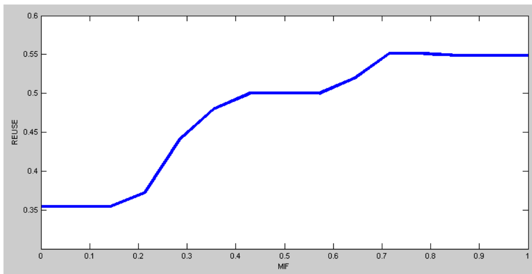

Figure 17 Relationship between *Reuse* and *MIF*

AIF expresses the level of reuse in the system. A threshold is maintained for AIF measure that is roughly around 65%. Higher values of AIF indicate high inheritance level, thereby leading to greater possibility of reuse. The graph in figure 18 below shows the relationship between input variable 'AIF' and output variable 'Reuse'. It shows that as the value of AIF increases the reuse factor of the class also increases.

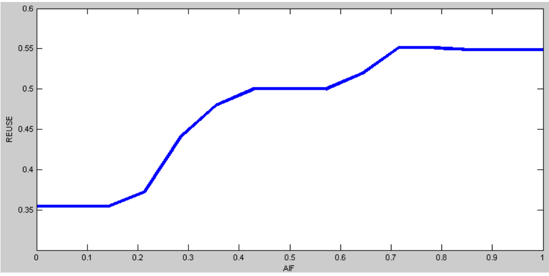

Figure 18 Relationship between *Reuse* and *AIF*

Results of Fuzzy controller 4

Coupling Factor (CF) has a very high positive correlation with all quality measures [11]. Therefore, as coupling between classes increase, the defect density and normalized rework also increase. Hence, coupling has a strong negative impact on software quality and should be avoided during the design because coupling relations increase complexity, reduce encapsulation and reuse. The graph in figure 19 shows the relationship between input variable 'COF' and output variable 'Reuse'. It shows that as the value of COF increases the reuse factor of the class decreases.

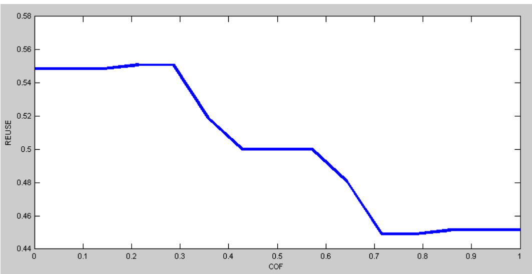

Figure 19 Relationship between *Reuse* and *COF*

Method hiding increases reusability and decreases complexity. A low MHF indicates inadequately abstracted design. A high MHF indicates very little functionality. It also indicates that the design includes a high percentage of specialized methods that are not available for reuse. The graph in figure 20 shows the relationship between input variable 'MHF' and output variable 'Reuse'. It shows that as the value of MHF increases the reuse factor of the class decreases.

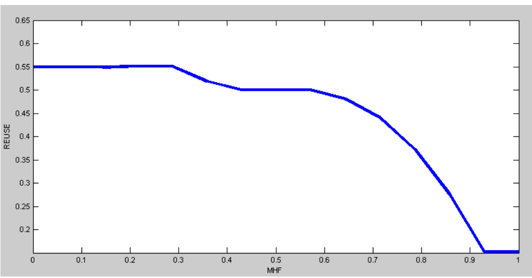

Figure 20 Relationship between *Reuse* and *MHF*

A low AHF indicates inadequately abstracted design. A large number of attributes are unprotected and the likelihood of errors is high. A high AHF indicates very little functionality. It may also indicate that the design includes a high section of specialized attributes that are not accessible for reuse. The graph in figure 21 shows the relationship between input variable 'AHF' and output variable 'Reuse'. It shows that as the value of AHF increases the reuse factor of the class decreases.

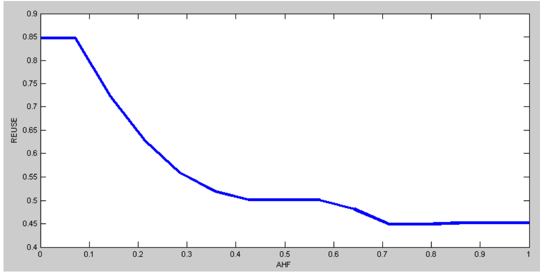

Figure 21 Relationship between *Reuse* and *AHF*

In the table 4 below, we present the empirical validation of the proposed model for the two projects and we see that the values tend to be generally lower for both the projects which mean that the reuse factor is medium.

Table 4 Empirical analysis of the proposed model

Metrics	Library Management system	Hospital Management System
MHF	0.305	0.897
AHF	0.375	0.667
MIF	0.491	1
AIF	0.676	1
COF	0.78	0.25
POF	0	0.8
O/P FLC1	0.511	0.462
O/P FLC2	0.5	0.847
Overall Reuse	0.414613	0.508424

6. Conclusion

This paper presents the results of a study conducted in which we experimentally investigated the suite of Object-Oriented (OO) design metrics introduced by Abreu et al. In order to do this, we assessed these metrics as predictors of reuse of classes. Based on our experimental results, the MOOD metrics appear to be useful to predict reuse factor of OOD during the early phases of the life-cycle. Based on the results we verified experimentally how much is the reuse of a design influenced by internal (e.g., Size, cohesion, encapsulation) and external (e.g., Inheritance, coupling) design characteristics of OO classes. The fuzzy logic based model proposed provides the practitioner with some experimental verification indicating that most of these metrics can be useful quality indicators.

Our future work includes:

- To conduct this study in an industrial setting: a sample of large-scale projects in the framework of the NASA, which should help us better understand the prediction capabilities of the suite of OO metrics described in this paper.
- To gain a better understanding of the impact of OO design strategies (e.g., Simple versus multiple inheritance) on how much effort is spent in each class independently.
- To extend the proposed model to other OO metrics proposed in the literature and develop new metrics that are more language specific.

REFERENCES

[1] Abreu, Fernando B. , Carapuca, Rogerio.: “Candidate Metrics for Object-Oriented Software within a Taxonomy Framework.", Journal of systems software 26, 1(July 1994).
[2] Abreu, Fernando B: "The MOOD Metrics Set," Proc. ECOOP'95 Workshop on Metrics, 1995.
[3] Abreu, Fernando B: “Design metrics for OO software system”,ECOOP’95, Quantitative Methods Workshop, 1995
[4] Abreu, Fernando B, Rita, E., Miguel, G. : “The Design of Eiffel Program: Quantitative Evaluation Using the MOOD metrics”, Proceeding of TOOLS’96 USA, Santa Barbara, California, July 1996.
[5] Roger S. Pressman: “Software Engineering”, Fifth edition, ISBN : 0077096770.
[6] Ramil, J.E and Lehman, M.M: “Metrics of evolution as effort predictors – a case study”. In Conf. Software Maintenance, pp. 163-172,
[7] Sharma, Aman Kumar, Arvind Kalia, and Hardeep Singh. "Metrics Identification for Measuring Object Oriented Software Quality." International Journal of Soft Computing. Volume-2, Issue-5, pp. 255-258, 2012.
[8] C. Catal, B. Diri, A systematic review of software fault prediction studies, Expert Systems with Applications 36 (2009) 7346–7354.
[9] Subramanyam R. and Krishnan M.S. “Empirical Analysis of CK Metrics for Object-Oriented Design Complexity: Implications for Software Defects” IEEE Transactions on Software Engineering, Vol. 29, No. 4,2003.
[10] Farooq A., Braungarten R., and Dumke R.R. “An Empirical Analysis of Object-Oriented Metrics for Java Technologies” 9th International Multitopic Conference, pp. 1-6, 2005.
[11] Jureczko, Marian, and Diomidis Spinellis. "Using object-oriented design metrics to predict software defects." Models and Methods of System Dependability. Oficyna Wydawnicza Politechniki Wrocławskiej,pp. 69-81, 2010
[12] Dubey, Sanjay Kumar, and Ajay Rana. "Assessment of maintainability metrics for object-oriented software system." ACM SIGSOFT Software Engineering Notes 36.5, pp. 1-7, 2011.
[13] Kulkarni, Umesh L., Y. R. Kalshetty, and Vrushali G. Arde. "Validation of ck metrics for object oriented design measurement." In Emerging Trends in Engineering and Technology (ICETET), 2010 3rd International Conference on, pp. 646-651. IEEE, 2010.
[14] Coad, P., "TogetherSoft Control Center," pp. http://togethersoft.com
[15] Goel, Brij Mohan, and Pradeep Kumar Bhatia. "Analysis of reusability of object-oriented systems using object-oriented metrics." ACM SIGSOFT Software Engineering Notes 38.4 (2013): 1-5.
[16] Mago, Jagmohan, and Parwinder Kaur.(2012). 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.
[17] Vir, R., Dhillon, P., & Dhillon, J. (2014). Fuzzy Logic Approach to Forecast the Extendibility/Extensibility in Object Oriented Design using an Integrated Model. International Journal of Computer Applications, 94(9).