# Fuzzy Logic Approach to Forecast the Extendibility/Extensibility in Object Oriented Design using an Integrated Model

Rajinder Vir
A.P
CTIEMT, Shahpur
Jalandhar, Punjab, India

Parwinder Dhillon
A.P
APEEJAY,Jalandhar
Punjab, India

Jaswinder Dhillon
A.P
DAVIET, Jalandhar
Punjab, India

## ABSTRACT
A number of researchers have conducted various empirical studies on the software metrics for Object Oriented design. The research proved that some of these metrices are very useful for forcasting the quality attributes of the software like extendibility/extensibility, effectiveness, reliability and maintainability. In this paper a hybrid approach is proposed for investigating the extendibility/extensibility of classes in Object Oriented design.Tthe hybrid approach will comprised of subset of CK netric suite and mood netric suite. These days a great demand occur for finding software measurement so that quality of software can be forcasted. Therefore software engineering require various quality models that can be used for forcasting the characteristics for quality such as extendibility/extensibility, effectiveness, reliability and maintainability. The main objective of this work is to experimentally forecast the association between OOD metrics and extendibility/extensibility.

## General Terms
Software Engineering, object Oriented design, extendibility, extensibility, Classes in object oriented Design.

## Keywords
MOOD Metric suite, CK metric suite, Fuzzy inference system, Mamdani inference model.

## 1. INTRODUCTION
Object Oriented Programming is a programming paradigm that represents concepts as objects and accompliced procedures known as methods. Objects are used so that applications and computer programs can be easily be interacted. OOP is essential for software developments, because it determines the structure of the software solution in an appropriate manner. Once the design is prepared, it is difficult to apply modifications and also it becomes very expensive. Therefore design should be focused from the beginning. Software metrics are the measurements that can be defined to forecast the quality of the software during the early phases of software development process. Metrics can be used to figure out the design quality. Many Metrics have been proposed for OOP.

It is analyzed from the previous research that CK metrics suite [13] and the MOOD metric suite [3] are found to be best to calculate the OOP quality. It is analyzed that software developers need combination of metrics to predict the quality of software. Hence there should be some integrated approach to combine these software metrics into a single output unit. Metrics offer a mechanism for attaining more accurate estimation of project milestones, and developing a software system that contains minimum faults [1]. There are a number of OOP software metrics available these days. These metrics are very helpful in fetching the information about the quality of OOP software. In this paper the way has been described how the evaluation of the extendibility/ extensibility using the software metrics has been done - CK metrics viz DIT and MOOD metrics viz MHF, AHF, AIF and MIF. The input metrics are divided into three linguistic terms low, medium and high.

The paper will proceed as follows: Section II will present the literature survey. Software quality along with its characteristics will be presented in section III. Introduction to software metrics will be presented in section IV. In next section V, paper will present the proposed integrated model based on fuzzy logic. In section VI paper will describe the evaluated experimental results. Finally in section VII, paper will discuss the related conclusion and the future scope.

## 2. LITERATURE REVIEW
A large number of metrics have been proposed in the past for so many years to confine the OO design, code and constructs. These metrics provide ways to assess the quality of software and their use in early phases of software development which can help software companies in evaluating large software development quickly and at a reasonable cost [4].

There have been large number empirical studies evaluating the impact of OO metrics on faulty classes. Saxena et al. [6] provided a review of all those empirical studies from 1995 to 2010 to predict software fault proneness with a specific focus on techniques used. Benlarbi et al. [11] surveyed that the basic premise behind the development of object oriented metrics is that they can serve as early predictors of classes that contain faults or that are closely maintain. They have shown that size can have an important confounding effect on the validity of object-oriented metrics. Khalsa [16] proposed an algorithm using fuzzy logic to measure fault proneness and defect density of the software development process and hence can be used to minimize rework.

Kamiya et. al. [5] proposed a new method to estimate the fault-proneness of an object class in the early phase, using several complexity metrics for object-oriented software. Four checkpoints were introduced into the analysis/ design/ implementation phase, and estimates were done on the fault-prone classes using applicable metrics at each checkpoint. Menzies et al. [14] compared Decision Trees, Naïve Bayes,

and 1-rule classifier on the NASA Software defect data. A clear trend was not observed & different predictors scored better on different data sets. Malhotra et. al. [8] builded a Support vector machine (SVM) model to find the relationship between object-oriented metrics given by Chidamber and Kemerer and fault proneness, at different severity levels. Malhotra [9] founded the relation between object oriented metrics and fault proneness using logistic regression method. The results were analyzed using open source software. Further, the performance of the predicted models was evaluated using Receiver Operating Characteristic (ROC) analysis.

## 3. SOFTWARE QUALITY

It is better to predict the software quality in the early phases of the software development. If software quality is evaluated after developing the software and after that it came to known that quality of software is not good as expected earlier, then again a lot of effort and work has to be done on the same software, so it is better to predict the software quality as early as possible. But there are limited standards that can be used to measure the quality. McCall proposed factors that affect the software quality.

Direct factors have direct effect to predict the software quality. Indirect factors have not direct effect to predict the software quality. Extendibility/extensibility comes under indirect factors. The quality of software is appraised by a various number of variables. Software functional quality is based on the functional requirements such as robustness or maintainability.Quality management system is the organizational structure, responsibility, procedures, activities, compatibilities, resources that together aim to ensure that software products will satisfy stated or implied needs.

## 4. SOFTWARE METRIC SUITES

Software metrics are the measurements used to predict the quality of software. Software metrics are used to obtain objective reproducible measurements that can be useful for quality assurance, performance, debugging, management and estimating costs. Metrics considered as most useful metrics to be those which measure the degree to which the software development effort reflects the priorities of its end users and developers. A large number of software metrics have been proposed by various researchers. Metrics can be characterized as follows:

1. System Size Metrics.
   - Lines of code
   - Function Count
   - Function Points
   - Number of Files
2. Object Oriented Metrics
   - Number of Classes
   - Number of Child classes
   - Number of Return Points
   - Depth of Inheritance Tree
3. Complexity Metrics
   - Boolean Expression Complexity
   - Class Data Abstraction Coupling
   - Cyclomatic Complexity
   - Function Interface Complexity
   - Npath Complexity

Out of these metrics the most popular metrics have been selected for the work, Chidamber and Kemerer metrics followed by MOOD metrics given by Abreu et al. the metrics are discussed in table1 and table 2.

**Table 1. CK Metric Suite [13]**

| Metric | Description |
|---|---|
| Weighted Methods per Class(WMC) | It defines the number of methods in a certain class. |
| Depth of Inheritance Tree(DIT) | It is a measure of how many ancestor classes can potentially affect a given class. |
| Number of Children(NOC) | Number of direct subclasses that a certain class contains. |
| Lack of Cohesion among Methods(LCOM) | Number of disjunctive method pairs of a certain class. |
| Coupling Between Objects(CBO) | Number of coupling between a certain class and all other classes. |
| Response For Class(RFC) | Number of methods that can be performed by a certain class in response to a received message. |

**Table 2. MOOD Metric Suite [17]**

| Metric | Description |
|---|---|
| Attribute Inheritance factor(AIF) | It is defined as the ratio of the sum of inherited attributes in all classes of the system. |
| Method Inheritance Factor(MIF) | The MIF metric states the sum of inherited methods in all classes of the system under consideration. |
| Attribute Hiding Factor(AHF) | Measure how well attributes and properties are encapsulated. |
| Method Hiding Factor(MHF) | Measure how well methods and variables are Encapsulated. |
| Polymorphism factor(POF) | The actual number of possible different polymorphic situation |
| Coupling Factor(COF) | Ratio of the maximum possible number of couplings in the system to the actual number of coupling is not imputable to inheritance |

| Metric | Software Quality Factors |
|---|---|
| **AIF** | Functionality, Effectiveness, Extendibility, Defect Proneness |
| **MIF** | Functionality, Effectiveness, Extendibility, Defect Proneness |
| **AHF** | Understandability, Complexity, Extendibility |
| **MHF** | Understandability, Complexity, Extendibility |
| **POF** | Complexity |
| **COF** | Complexity, Reusability |

**Table 4. Relationship between MOOD Metrics and Software Quality Factors [2]**

Research has shown that the CK Metric Suite does not account for the complexity that occurs from the Object Oriented Design factors such as extendibility and extensibility but the metrics proposed by Abreu are able to measure the object Oriented Design aspects properly [10]. The metrics AHF and MHF measure the information hiding aspects of the class, AIF and MIF metrics measure the inheritance aspects of the class. Hence, an integrated hybrid model is used to measure the extendibility/extensibility of the Object Oriented systems. The work described in this paper focuses on the use of Object Oriented (OO) metrics in predicting extendible/extensible classes.

**Table 3. Relationship between CK Metrics and Software**

| **Metric** | **Software Quality Factor** |
|---|---|
| WMC | Complexity, Usability, Reusability |
| DIT | Reusability, Understandability, Extendibility, Testability |
| NOC | Design |
| LCOM | Design, Reusability |
| CBO | Design, Reusability |
| RFC | Design, Usability, Testability |

## 5. PROPOSED MODEL

The proposed MODEL for calculating the extendible/extensible classes in OOD uses a subset of CK metrics and MOOD metrics.

A fuzzy logic model FCD-extend (Fuzzy Controller for extendibility/extensibility) is used to predict the extendible/extensible classes. The implemented model comprises of one metric from CK metric (DIT) and four metrics from MOOD metric suite (AIF, MIF, AHF, MHF). These five inputs are fed into the fuzzy systems. Depending upon the input values of the metric, some rules out of the total 243 rules from the knowledge base gets fired. The Sugeno inference engine is used to determine the degree of membership of firing. The technique used for defuzzification is *'wtaver'*. In this parer Fuzzy Inference system is used to implement the model.

Software quality is an important aspect used for predicting the extendibility or extensibility. It becomes difficult for the designer to calculate the software quality if any uncertainties occurs during software development process. Calculating the quality of OOD is a fuzzy evaluation process. Therefore for achieving the objective and empirical evaluation of the software quality based on extendibility, rule based logic system proposed by Zadeh has been used. Fuzzy Logic is a problem solving control which leads to implementation in the system ranging from simple, small and embedded micro controllers to large networks, multi channel workstations. It provides a definite conclusion to control the problems and take a decision as much as faster.

Fuzzy Logic is a technique used for modeling complex systems. It can be constructed either without any data or little data which means fuzzy logic is superior over other data

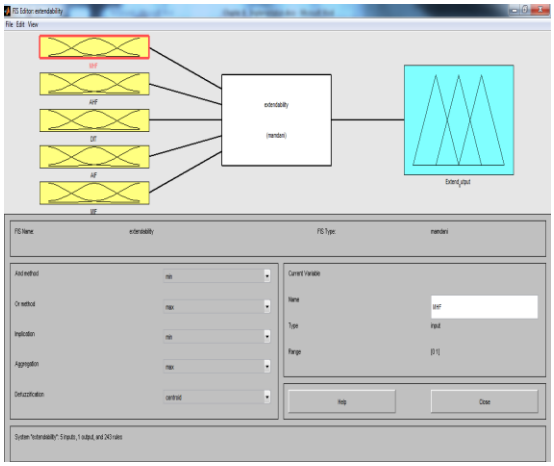

**Fig 1. Fuzzy Controller for Extendibility**

driven approach such as neural networks, regression analysis and case base reasoning [15].

Fuzzy Logic is considered as a better method for sorting and handling the data. It is based on the degrees of truth and ideas of fuzzy logic that work on the problems of computer understandability of natural language.

**Fig 2. Rule viewer for Extendibility/Extensibility**

## 6. EXPERIMENTAL RESULTS

The main four pillars of Maintainability are Reusability, Reliability, Robustness and extendibility or extensibility. Enhancing these four qualities of an Object Oriented Design, maintenance cost of the system can be reduced. In this thesis work a fuzzy logic based model is proposed to determine the extendibility/extensibility of an Object Oriented Design. With the help of this model, the software designers can reduce the maintenance cost as extendibility/extensibility takes the advantage of users' natural tendency to ask for "base-displacement" kinds of modifications to their systems [36]. Moreover, the high cost of building large software systems is due to the fact that most of the software development is done from scratch.

The class serves as the starting point for measuring the extendibility/extensibility of the Object Oriented system. Classes can be combined and modified to fit a new application by means of inheritance and polymorphism.

**MEMBERSHIP FUNCTIONS FOR INPUT METRICS**

Input parameters are assigned to the linguistic variables based on their values. To assign the input parameters to the

linguistic variables range of the input is measured. The curves for the membership functions used to predict the defect proneness using MOOD metric suite defined below. The input metrics DIT, MHF, AHF, AIF, and MIF are divided into three stages Low, Medium and high.

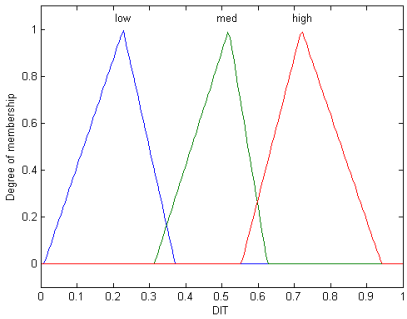


**Fig 3 Membership Function for input variable DIT**

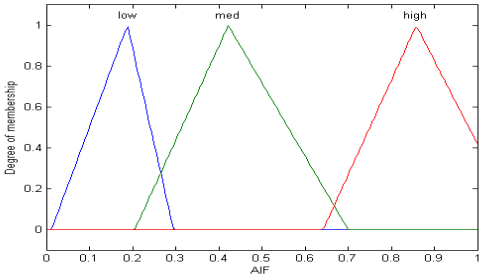


**Fig 4 Membership Function for input variable AIF**

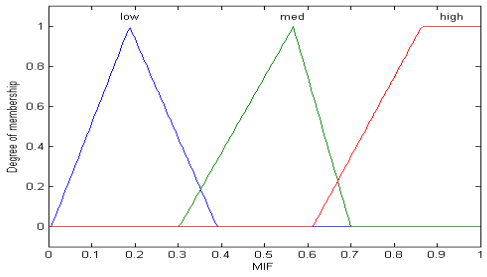


**Fig 5 Membership Function for input variable MIF**

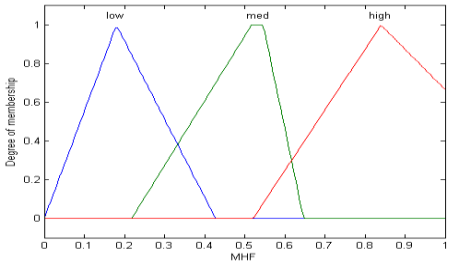


**Fig 6 Membership Function for input variable AHF**

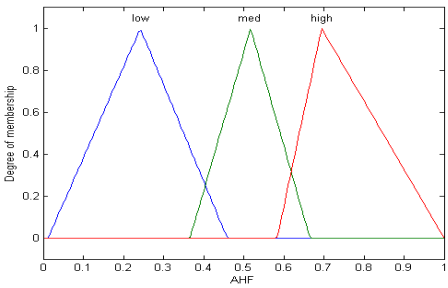


**Fig 7 Membership Function for input variable MHF**

To perform the empirical investigation for validating the fuzzy control model for extendibility/extensibility, In this paper various open source software and Project Analyzer tool has been used. After creating the rule base to depict the true picture following results were obtained as shown in the form of graphs in Fig 3, 4,5,16,7,8,9,10,11 & 12.

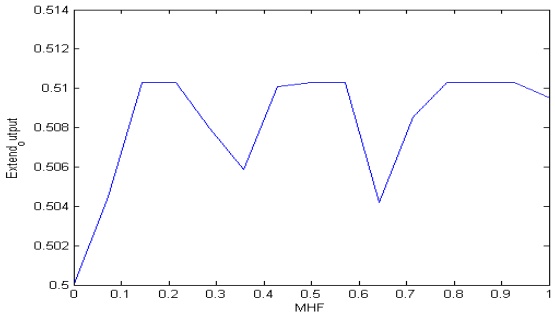


**Fig 8. Graph for MHF and Extendibility**

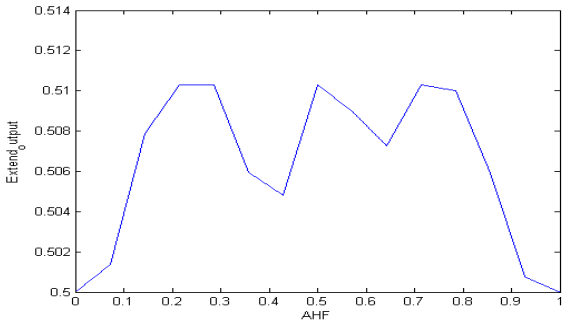


**Fig 9. Graph for AHF and Extendibility**

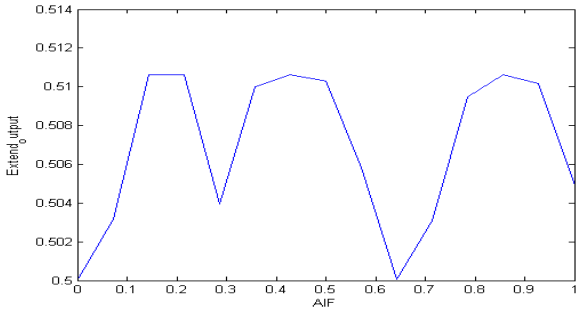


**Fig 10. Graph for AIF and Extendibility**

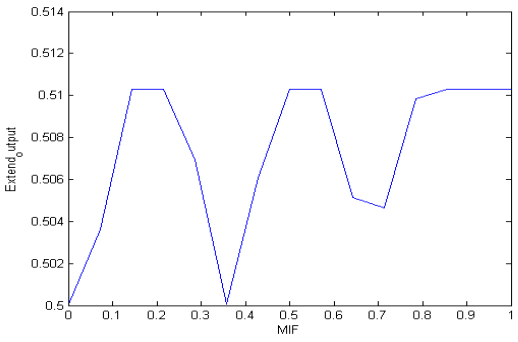


**Fig 11. Graph for MIF and Extendibility**

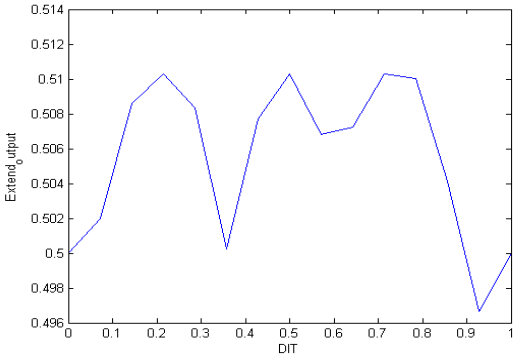


**Fig 12. Graph for DIT and Extendibility**

From the above results it is clear that classes can be extended by taking under considerations the various inheritance factors and hiding factors. In the fig 8 it is observed that with the increase of method hiding factors in class, chances of extending a class are also increased. Similarly in the fig 10 extendibility increases when the range of AIF is 0 and 1 and it varies between the range. This is because of the variation of number of classes.

## 7. CONCLUSION AND FUTURE WORK

The goal of this work is to analyze the performance of integrated CK-MOOD based on the performance characteristics of the software such as extendibility/extensibility, maintainability, reliability, defect Proneness, effectiveness, efficiency in a class. In this paper performance of proposed model has been analyzed using fuzzy logic approach. Metrics used for analyzing the performance are CK and MOOD metric suite. The model can be used for estimating the extendible/extensible classes in the early phases of software development which in result reduce the effort of the software developers. Therefore, this model can improve the quality of software by predicting the extendible/extensible classes early in the OOD.

In this papere work on extendibility/extensibility has been done. The work can be extended to find other software quality characteristics such as portability, reliability, maintainability, efficiency. The work can also be extended by taking other software metrics. In this paper one metric from CK metric suite and four metrics from MOOD metric suite for predicting the extendibility/extensibility of a class.

## 8. REFERENCES

[1] D. Bellin, Manish Tyagi and Maurice Tyler, "*Object-Oriented Metrics: An Overview*", Computer Science Department, North Carolina A, T State University, Greensboro, Nc 27411-0002.

[2] 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.

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

[4] K.K.Aggarwal, Y. Singh, A. Kaur, R. Malhotra, "*Software Reuse Metrics for Object-Oriented Systems*", Proceedings 3rd ACIS Int'l Conference on Software Engineering Research, Management and Applications (SERA'05), IEEE Computer Society, pp. 48-55, 2005.

[5] Kamiya, Toshihiro, Kusumoto, Shinji, Inoue, Katsuro "*Prediction of fault-proneness at early phase in object-oriented development*" Proceedings 2nd IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, (ISORC '99) , pp: 253 – 258, 1999.

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

[7] R. Bhatnagar, V. Bhattacharje and M. K. Ghose, "*A proposed novel framework for early effort estimation using fuzzy logic techniques*", Proceedings Global Journal of Computer Science and Technology, Vol 10, No. 14, 2010.

[8] 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*", Proceedings of International Journal of System Assurance Engineering and Management, Vol. 1, Issue 3, pp. 269-281, September, 2010.

[9] R. Malhotra, "*A Defect Prediction Model for Open Source Software*", Proceedings of the World Congress on Engineering 2012, Vol. II, July 4 - 6, 2012, London (UK). WCE 2012.

[10] R. Subramanyam and M. S. Krishnan, "*Empirical Analysis of CK Metrics for Object-Oriented Design Complexity: Implications for Software Defects*", IEEE Transactions on Software Engineering, Vol. 29, Issue: 4, pp: 297 – 310, April, 2003.

[11] S. Benlarbi, K. El Emam and N. Geol, "*Issues in Validating Object-Oriented Metrics for Early Risk Prediction*", by Cistel Technology 210 Colonnade Road Suite 204 Nepean, Ontario Canada K2E 7L5, 1999.

[12] S. K Dubey and A. Rana, "*A Comprehensive Assessment of Object-Oriented Software Systems Using Metrics Approach*", IJCSE, Vol. 02, pp. 2726–2730, 2010.

[13] 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.

[14] T. Menzies, K. Ammar, A. Nikora, and S. Stefano, "*How Simple is Software Defect Prediction*?" Journal of Empirical Software Engineering, October, 2003.

[15] L. A. Zadeh, "*Fuzzy sets, Information and Controll*", Vol. 8, pp. 338-353, 1965.

[16] Sunint K. Khalsa, "*A Fuzzified Approach for the Prediction of Fault Proneness and Defect Density*", Proceedings of the World Congress on Engineering, Vol. I, WCE 2009, July 1 - 3, 2009, London, U.K., 2009.

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

[18] L. H. Rosenberg and L. Hyatt, "*Software Quality Metrics for Object Oriented Environments*", Crosstalk Journal, 1997.

[19] J. Bansiya and C.G. Davis, "*A Hierarchical Model for Object-Oriented Design Quality Assessment*", IEEE Transactions on Software Engineering, Vol. 28, Issue: 1, 2002.

[20] A. Handa and G. Wayal, "*Software Quality enhancement using Fuzzy Logic with object oriented metrics in design*", International Journal of Computer Engineering and Technology (IJCET), Vol. 3, Issue: 1, pp: 169-179, June 2012.