Empirically Investigating the Effect Of Design Metrics On Fault Proneness in Object Oriented Systems

1. Amjan. Shaik, Professor and HOD-CSE, Ellenki College of Engineering and Technology (ECET), Patelguda, Hyderabad, India.

2. Dr.C.R.K.Reddy, Professor and HOD-CSE, Chaitanya Bharathi Institute of Technology (CBIT), Gandipet, Hyderabad, India.

3. Bala Manda,M.Tech(CSE), R&D, Ellenki College of Engineering and Technology (ECET), Patelguda, Hyderabad, India.

ABSTRACT

In the era of software metrics demand for quality software has undergone with rapid growth during the last few years. This is leading to an increase in the development of metrics for measuring the properties of software such as coupling, cohesion and inheritance that can be used in early Quality assessments. Much effort has been developed to the development and empirical validation of software metrics. Quality models that explore the relationship between these properties and quality attributes such as fault proneness, maintainability, effort or productivity are needed to use these metrics effectively. The goal of this work is to empirically explore the relationship between Object Oriented Design Metrics and Fault Proneness of object oriented system classes. We empirically analyzed and tested by Open Source Java projects.

Keywords: Object Oriented Metrics, Coupling, Cohesion, Inheritance, Empirical Analysis.

1. INTRODUCTION

There are several Metrics proposed for capturing the quality of object oriented design. These metrics provide ways to evaluate the quality of software and their use in earlier phases of software development life cycle. But how do we know which metrics are useful in capturing important quality attributes such as fault proneness, effort or productivity. Empirical studies of real systems can provide the relevant answers. More data based by empirical studies which are capable of being verified by observation or experiment are needed. The validation of software metrics has received much research attention by the software engineers. There are two types of validation that are recognized[9]:internal and external. Internal validation is a theoretical exercise that ensures that the metric is a proper numerical characterization of the property it claims to measure. External validation involves empirically demonstrating that the product metric is associated with some important external metric (such as measures of maintainability or reliability). These are also commonly referred to as theoretical and empirical validation, respectively .The metrics we investigate here consist of CK Metrics suite[2,10,11] and some other metrics. Univariate logistic regression models and principal component method are used as the basis for demonstrating the relationship between object oriented metrics and fault proneness[4].

Univariate Logistic regression analysis is carried out to test that size, coupling and inheritance increase fault proneness of a class where as cohesion decrease fault proneness of a class and find individual impact of metrics on fault proneness.

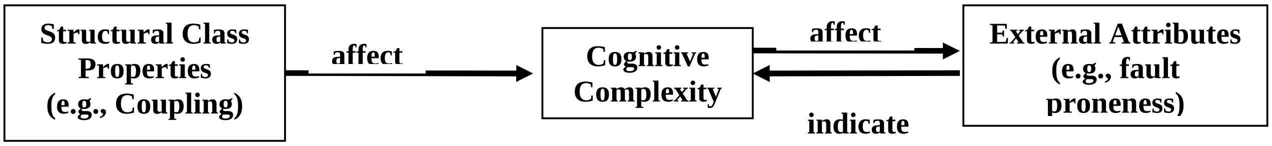
Principal component method of factor analysis is used to find whether all these metrics are in dependent or are capturing same underlying property of the object being measured.

2.RESEARCH BACKGROUND

In this section ,we present the theoretical and empirical basis of the object oriented metrics that we attempt to validate.

2.1 Cognitive Theory of Object Oriented Metrics

A theoretical basis for developing quantitative models relating product metrics and external quality metrics has been provided in [5] and is summarized in Fig[1].



Fig[1].Theoretical basis for the development of object oriented product metrics

This theory hypothesizes that the structural properties of a software component (such as its coupling) have an impact on its cognitive complexity. The cognitive complexity is an intervening variable between the structural properties of classes and fault-proneness. Cognitive complexity is defined as the mental burden of the individuals who have to deal with the component, for example, the developers, testers, inspectors, and maintainers. High cognitive complexity leads to a component exhibiting undesirable external qualities, such as increased fault-proneness and reduced maintainability. Accordingly, object-oriented product metrics that affect cognitive complexity will be related with fault-proneness.

It would provide us with a clear mechanism that would explain the introduction of faults into object-oriented applications. The current theoretical framework for explaining the effect of the structural properties of object oriented programs on external program attributes can be justified empirically. To be specific, studies that have been performed indicate that the distribution of functionality across classes in object-oriented systems, and the exacerbation of this through inheritance, potentially makes programs more difficult to understand. This suggests that highly cohesive, sparsely coupled, and low inheritance programs are less likely to contain a fault. Therefore, metrics that measure these three dimensions of an object oriented program would be expected to be good predictors of fault-proneness or the number of faults. The empirical question is then whether contemporary object-oriented metrics measure the relevant structural properties well enough to substantiate the above theory.

2.2 Empirical validation of Object Oriented Metrics on Fault Proneness

In this section, we review the empirical studies that investigate the relationship between object-oriented metrics and fault-proneness.

Metrics Studied

The metrics of coupling, cohesion, inheritance and size are the independent variables used in this study[10,11]. Our focus is on OO metrics that are used as independent variables in a prediction model that is usable at early stages of software development. The metrics selected in this paper are summarized in Table 1.

Metric	Definition
Coupling between Objects (CBO)	CBO for a class is count of the number of other classes to which it is coupled.
Coupling between Objects (CBO1)	Same as CBO, except that inheritance based coupling is not counted.
Lack of Cohesion (LCOM1)	It counts number of null pairs of methods that do not have common attributes.
Lack of Cohesion (LCOM2)	It measures the dissimilarity of methods in a class by looking at the instance variable or attributes used by methods.
Number of Children (NOC)	The NOC is the number of immediate subclasses of a class in a hierarchy.
Depth of Inheritance (DIT)	The depth of a class within the inheritance hierarchy is the maximum number of steps from the class node to the root of the tree and is measured by the number of ancestor classes.
Weighted Methods per Class (WMC)	The WMC is a count of sum of complexities of all methods in a class.
Response for a Class (RFC)	The response set of a class (RFC) is defined as set of methods that can be potentially executed in response to a message received by an object of that class.

IFCAIC ACAIC OCAIC FCAEC DCAEC OCAEC IFCMIC ACMIC DCMIC FCMEC DCMEC OCMEC IFMMIC AMMIC OMMIC FMMEC DMMEC OMMEC	These coupling metrics count number of interactions between classes. The metrics distinguish the relationship between the classes (friendship, inheritance, none), different types of interactions, and the locus of impact of the interaction. The acronyms for the metrics indicates what interactions are counted: • The first or first two characters indicate the type of coupling relationship between classes (A: Ancestor, D:Descendents, F: Friend classes, IF: Inverse Friends (classes that declare a given class a as their friend), O:Others, i.e., none of the above relationships). • The next two characters indicate the type of interaction: CA: There is a Class-Attribute interaction if class x has an attribute of type class y. CM: There is a Class-Method interaction if class x consist of a method that has parameter of type class y. MM: There is a Method-Method interaction if class x calls method of another class y, or class x has a method of class y as a parameter. • The last two characters indicate the locus of impact: IC: Import coupling, counts the number of other classes called by class x. EC: Export coupling, count number of other classes using class y.
Lines Of Code (LOC)	It is the count of lines in the text of the source code excluding comment lines

Table 1: Object-Oriented Metrics

Empirical Data Collection

Case Study1: We consider five Java projects [10] mentioned below that are licensed as GNU open source from various domains :
Project 1:BLACKDUCKKODERS
(http://www.koders.com/):10versions chosen
Project 2: STRAR UML-One of the UML tool to design UML diagrams
(http://www.osalt.com/staruml): 5.0 versions chosen.
Project 3: OpenOffice Draw 3.0
(http://www.openoffice.org/product/draw.html): 3.0 versions chosen
Project 4: InfraRecorder 0.50
(http://infrarecorder.org/):0.5 versions chosen
Project 5: Gimpshop 2.2.11
(http://plasticbugs.com/?page_id=294): 2.2.11 versions Chosen

Case Study2: To analyze the metrics chosen for this work, their values are computed for ten different systems. These systems are developed by M.Tech Students. The following relevant data was collected:
1. The design and source code of the java programs and
2. The faulty data found by the testing team.
The 10 systems under study consists of 200 classes out of which 130 are system classes and 70 are standard library classes available in java language. These classes contain functions to manipulate files, strings, lists, hash tables, frames, windows, menus, threads, socket connection etc. All metric values are computed on system classes whereas coupling and inheritance metrics are also calculated between 'system classes' and 'standard library classes'.

Observation

It was observed during testing on both the Case Studies the classes coupled with standard library classes were less fault prone than those coupled with system classes.

3.RESEARCH METHODOLOGY

In this section, we review the research methodology that investigate the relationship between object-oriented metrics and fault-proneness. The product metrics cover the following dimensions: coupling, cohesion, inheritance, and complexity. Coupling metrics characterize the static usage dependencies among the classes in an object-oriented system [6]. Cohesion metrics characterize the extent to which the methods and attributes of a class belong together [7].Inheritance metrics characterize the structure of the inheritance hierarchy.

Logistic Regression Model:

Logistic Regression (LR) model is the most widely used technique in literature to predict dependent variable from set of independent variables. In our work independent variable are OO metrics and dependent variable is fault proneness. LR is of two types: (i) Univariate LR (ii) Multivariate LR Univariate LR is a statistical method that formulates a mathematical model depicting relationship among each independent variable and dependent variable to determine if the measure is statistically related, in the expected direction, to fault proneness. Multivariate LR is used to construct a prediction model for the fault-proneness of classes. In this method combination of metrics are used to determine the effect on dependent variable.

In our research we used univariate logistic regression model. The general form of an LR model is :

$$\pi = \frac{1}{1+e^{-\left(\beta_0+\sum_{i=1}^{k}\beta_i x_i\right)}}, \qquad (1)$$

where π is the probability of a class having a fault, and the xs are the independent variables. In a univariate analysis only one xi, x1, is included in the model, and this is the product metric that is being validated:

$$\pi = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1)}}. \qquad (2)$$

When controlling for size, a second xi, x2, is included that measures size:

$$\pi = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2)}}. \qquad (3)$$

The magnitude of an association can be expressed in terms of the change in odds ratio as the x1 variable changes by one standard deviation. The odds ratio is a measure of association. The odds of an event, is the ratio of the number of ways the event can occur to the number of ways the event cannot occur. The odds ratio is the ratio of faulty classes and non-faulty classes. If a metric is not related to fault-proneness, then the odds ratio is equal to one. If there is a positive association, then the odds ratio will be greater than one, and if there is a negative association, then the odds ratio will be a fraction. Let D denote the presence of a fault (D . 1) or absence (D . 0), and let x be our coupling metric. Then,

$$\Pr(D=1|x) = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}} \qquad (4)$$

is the probability of a fault given the value of x. The probability of there not being a fault given the value of x is:

$$q_x = 1 - p_x =$$

$$\Pr(D=0|x) = 1 - \frac{1}{1+e^{-(\beta_0+\beta_1 x)}} = \frac{e^{-(\beta_0+\beta_1 x)}}{1+e^{-(\beta_0+\beta_1 x)}}. \qquad (5)$$

The odds of a class having a fault given the value of x is

$$\frac{p_x}{q_x} = \Psi(x) = e^{\beta_0+\beta_1 x)}. \qquad (6)$$

The odds of a class having a fault if the product metric value x is increased by one standard deviation is:

$$\Psi(x+\sigma) = e^{\beta_0+\beta_1(x+\sigma))}. \qquad (7)$$

The change in odds by increasing the value of x by one standard deviation is:

$$\Delta\Psi = \frac{\Psi(x+\sigma)}{\Psi(x)} = e^{\beta_1 \sigma}. \quad (8)$$

In this subsection we find the relationship of independent variables (OO metrics) with dependent variable (fault proneness). Univariate LR analysis is done on 85 system classes. The table 2 provides the coefficient (B), standard error (SE), statistical significance (sig), R2 statistic and odds ratio (exp(B)), for each measure. Metrics with no variance or lower variance are excluded from the table. The metrics with a significant relationship to fault proneness, that is, below or at the significance (named as Sig. in Table 2) threshold of 0.05 are shown in bold (see Table 2). The metrics that are not shown in bold do not have a significant relationship with fault proneness.

Metric	B	**S.E.**	**Sig.**	**R2**	**Exp(B)**
CBO	0.8436	0.2802	**0.0026**	0.1206	2.3246
CBO1	0.6180	0.2491	**0.0131**	0.077	1.8553
LCOM1	0.0612	0.0244	**0.0121**	0. 2155	0.0631
LCOM2	0.0800	0.0347	**0.0212**	0.1982	1.0832
DIT	0.7518	0.4279	0.0789	0.0344	0.4715
NOC	0.3147	0.2666	0.2379	0.0172	1.3698
LOC	0.0100	0.0033	**0.0025**	0.273	1.0101
RFC	0.1817	0.0410	**0.0000**	0.536	1.1993
WMC	0.2466	0.0646	**0.0001**	0.375	1.2796
OCAEC	0.0731	0.2552	0.7746	0.0000	1.0758
OCAIC	0.9381	0.3594	**0.0090**	0.077	2.5552

Table 2: Statistical results for fault proneness.

Principal Component Method:

Principal Component Method(PCM) is a mathematical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of uncorrelated variables called **principal components. PCM** is used to maximize the sum of squared loadings of each factor extracted in turn. The PC Method aims at constructing new variable (Pi), called Principal Component (PC) out of a given set of variables
Xj' *s*(*j* = 1,2,...., *k*) . The variables with high loadings help identify the dimension P.C. is capturing, but this usually requires some degree of interpretation. In order to identify these variables, and interpret the PC.s, we consider the rotated components. As the dimensions are independent, orthogonal rotation is used. There are various strategies to perform such rotation. We used the varimax rotation, which is the most frequently used strategy in literature. Eigenvalue (or latent root) is associated with each PC. It refers to the sum of squared values of loadings relating to dimension, and then the sum is referred to as eigenvalue. Eigenvalue indicates the relative importance of each dimension for the particular set of variables being analyzed. In our study, the PC.s with eigenvalue greater than 1 is taken for interpretation.

The coupling of system classes to system classes is counted separately from coupling of system classes to standard library classes. SL is suffixed with the metric name when coupling to standard library classes is counted. For instance CBO metric in such case is named as CBO_SL. The PC extraction method and varimax rotation method is applied on all metrics. The rotated component matrix is given in Table 3. The values above 0.7 (shown in bold in Table 3) are the metrics that are used to interpret the PC.s. For each PC., we also provide its eigenvalue, variance percent and cumulative percent. The interpretations of PCs are given as follows:

- P1: CBO_SL, OCAIC_SL, OCMIC_SL, CBO1_SL and OMMIC_SL measure coupling from standard library classes.
- P2: LCOM1, LCOM2, WMC and OCMIC. This dimension includes coupling, cohesion and size metrics. This indicates that import coupling and cohesion metrics have correlation with size.
- P3: OMMIC, RFC are coupling metrics. These metrics count import coupling from system classes through method invocations.
- P4: AMMIC_SL, OCAIC are import coupling metrics.
- P5: CBO, CBO1 are coupling metrics that count both import and export coupling.
- P6: NOC is an inheritance metric that counts number of children of a class.

PC	P1	P2	P3	P4	P5	P6
Cumulative%	32.608	44.97	56.010	63.676	70.424	75.603
Variance %	32.608	12.3	11.03	7.665	6.748	5.1788
Eigenvalue	6.84	2.59	2.31	1.60	1.41	1.08
CBO	0.12	0.00	0.18	0.14	**0.91**	-0.05
CBO_SL	**0.80**	0.15	0.07	0.37	0.12	0.16
CBO1	0.03	-0.03	0.18	-0.11	**0.94**	-0.01
LCOM1	0.28	**0.87**	0.26	0.06	-0.07	0.01
LCOM2	0.28	**0.88**	0.21	0.01	-0.08	0.00
DIT	-0.25	-0.14	0.36	-0.29	-0.28	-0.25
NOC	0.10	-0.07	0.17	-0.04	-0.08	**0.80**
LOC	0.27	0.41	0.68	0.02	0.05	0.17
RFC	0.20	0.34	**0.76**	0.15	0.07	0.17
WMC	0.35	**0.74**	0.49	0.16	0.01	0.17
OCAEC	0.48	-0.00	-0.04	0.41	-0.09	-0.41
OCAIC	0.10	0.19	0.08	**0.74**	0.04	0.39

Table 3: Results for Principal Component Method

4.CONCLUSION

In this study we first find the interrelationships among selected metrics and then found the individual and combined effect of selected metrics on fault proneness. The results of univariate LR analysis show that most of the import coupling and cohesion metrics are found related to fault proneness. On the other hand inheritance metrics were not found related to fault proneness . We are also applied Principal component method to these metrics to get the Fault proneness. The number of dimensions captured in PC analysis is much lower than the number of metrics. This simply supports the fact that many of the metrics proposed are based on comparable ideas and therefore provide somewhat redundant information. It was observed during testing the classes coupled with standard library classes were less fault prone than those coupled with system classes.

ACKNOLEDGEMENTS

The authors would like to thank Ellenki College of Engineering & Technology‘s Research & Development Center for their cooperation and support in data collection .

ABOUT THE AUTHORS:

Amjan Shaik is working as a Professor and Head, Department of Computer Science and Engineering at Ellenki College of Engineering and Technology (ECET), Hyderabad, India. He has received M.Tech. (Computer Science and Technology) from Andhra University. Presently, he is a Research Scholar of JNTUH Hyderabad. He has published and presented more than 30 Research and technical papers in International Journals , International Conferences and National Conferences. His main research interests are Software Engineering, Software Metrics, Software Quality and OOD.

Dr.C.R.K. Reddy is working as a Professor and Head, Department of Computer Science and Engineering at Chaitanya Bharathi Institute of Technology (CBIT),Hyderabad, India. He has received M.Tech.(Computer Science and Engineering) from JNTUH, Hyderabad and Ph.D in Computer Science and Engineering from Hyderabad Central University (HCU). He has widely published and presented Research and Technical Papers in International Journals, National Journals, International Conferences and National Conferences. He is a distinguished academician and administrator in CBIT. At Present 8 Research Scholars are doing Ph.D under his esteemed guidance. His main research Interests are Program Testing , Software Engineering , Software Metrics, Artificial Intelligence and Software Architectures.

Bala Manda is pursuing M.Tech (Computer Science and Engineering) from Ellenki College of Engineering & Technology, Hyderabad, India. She has received MCA from Andhra University, Visakhapatnam. She has published 2 research articles in International Journals. Her research interests are Software Metrics, Quality and Object-Oriented Languages.

REFERENCES:

[1] K.El Emam, W. Melo, J. Machado, “The Prediction of Faulty Classes Using Object-Oriented Design Metrics”, *Journal of Systems and Software*, vol. 56, 63-75, 2001

[2] S.Chidamber and C.Kemerer, “A metrics Suite for Object-Oriented Design ”,*IEEE Trans. Software Engineering*, vol. SE-20, no.6, 476-493, 1994.

[3] M-H. Tang, M-H. Kao, and M-H. Chen, ªAn Empirical Study on Object Oriented Metrics,º Proc. Sixth Int'l Software Metrics Symp., pp. 242-249, 1999.

[4] L. Briand, J. Wuest, J. Daly, and V. Porter, ªExploring the Relationships Between Design Measures and Software Quality in Object-Oriented Systems,º J. Systems and Software, vol. 51, pp. 245- 273, 2000.

[5] L. Briand, J. Wuest, S. Ikonomovski, and H. Lounis, ªA Comprehensive Investigation of Quality Factors in Object-Oriented Designs: An Industrial Case Study,º Technical Report ISERN-98-29, Int'l Software Eng. Research Network, 1998.

[6] L. Briand, J. Daly, and J. Wuest, ªA Unified Framework for Coupling Measurement in Object-Oriented Systems,º IEEE Trans. Software Eng., vol. 25, no. 1 pp. 91-121, Jan. 1999.

[7] L. Briand, J. Daly, and J. Wuest, ªA Unified Framework for Cohesion Measurement in Object-Oriented Systems,º Empirical Software Eng., vol. 3, pp. 65-117, 1998.

[8] L. Dales and H. Ury, ‘An Improper Use of Statistical Significance Testing in Studying Co-variables,’ International Journal of Epidemiology, vol. 7, no. 4, pp. 373-375, 1978.

[9] N. Fenton, ªSoftware Metrics: Theory, Tools and Validation,’ Journal of Software Engineering’., pp. 65-78, Jan. 1990.

[10] Amjan Shaik, C. R. K. Reddy, Bala Manda, Prakashini. C, Deepthi. K” An Empirical Validation of Object Oriented Design Metrics in Object Oriented Systems” International Journal of Emerging Trends in Engineering and Applied Sciences (IJETEAS) 1 (2): 216-224 (ISSN: 2141-7016).

[11] Amjan Shaik, C. R. K. Reddy, Bala Manda, Prakashini. C, Deepthi,” Metrics for Object Oriented Design Software Systems: A Survey ”International Journal of Emerging Trends in Engineering and Applied Sciences (IJETEAS) 1 (2): 190-198 (ISSN: 2141-7016).