# INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATION & MANAGEMENT

IJRCM

A Monthly Double-Blind Peer Reviewed Refereed Open Access International e-Journal - Included in the International Serial Directories
Indexed & Listed at:
Ulrich's Periodicals Directory ©, ProQuest, U.S.A., EBSCO Publishing, U.S.A., Cabell's Directories of Publishing Opportunities, U.S.A.
as well as inOpen J-Gage, India [link of the same is duly available at Inflibnet of University Grants Commission (U.G.C.)]
Registered & Listed at: Index Copernicus Publishers Panel, Poland
Circulated all over the world & Google has verified that scholars of more than 1388 Cities in 138 countries/territories are visiting our journal on regular basis.
Ground Floor, Building No. 1041-C-1, Devi Bhawan Bazar, JAGADHRI – 135 003, Yamunanagar, Haryana, INDIA
www.ijrcm.org.in

# CONTENTS

# CALL FOR MANUSCRIPTS

Weinvite unpublished novel, original, empirical and high quality research work pertaining to recent developments & practices in the area of Computer, Business, Finance, Marketing, Human Resource Management, General Management, Banking, Insurance, Corporate Governance and emerging paradigms in allied subjects like Accounting Education; Accounting Information Systems; Accounting Theory & Practice; Auditing; Behavioral Accounting; Behavioral Economics; Corporate Finance; Cost Accounting; Econometrics; Economic Development; Economic History; Financial Institutions & Markets; Financial Services; Fiscal Policy; Government & Non Profit Accounting; Industrial Organization; International Economics & Trade; International Finance; Macro Economics; Micro Economics; Monetary Policy; Portfolio & Security Analysis; Public Policy Economics; Real Estate; Regional Economics; Tax Accounting; Advertising & Promotion Management; Business Education; Management Information Systems (MIS); Business Law, Public Responsibility & Ethics; Communication; Direct Marketing; E-Commerce; Global Business; Health Care Administration; Labor Relations & Human Resource Management; Marketing Research; Marketing Theory & Applications; Non-Profit Organizations; Office Administration/Management; Operations Research/Statistics; Organizational Behavior & Theory; Organizational Development; Production/Operations; Public Administration; Purchasing/Materials Management; Retailing; Sales/Selling; Services; Small Business Entrepreneurship; Strategic Management Policy; Technology/Innovation; Tourism, Hospitality & Leisure; Transportation/Physical Distribution; Algorithms; Artificial Intelligence; Compilers & Translation; Computer Aided Design (CAD); Computer Aided Manufacturing; Computer Graphics; Computer Organization & Architecture; Database Structures & Systems; Digital Logic; Discrete Structures; Internet; Management Information Systems; Modeling & Simulation; Multimedia; Neural Systems/Neural Networks; Numerical Analysis/Scientific Computing; Object Oriented Programming; Operating Systems; Programming Languages; Robotics; Symbolic & Formal Logic and Web Design. The above mentioned tracks are only indicative, and not exhaustive.

Anybody can submit the soft copy of his/her manuscript **anytime** in M.S. Word format after preparing the same as per our submission guidelines duly available on our website under the heading guidelines for submission, at the email addresses: infoijrcm@gmail.comorinfo@ijrcm.org.in.

# GUIDELINES FOR SUBMISSION OF MANUSCRIPT

1. **COVERING LETTER FOR SUBMISSION**:

**DATED: _____________**

***THE EDITOR***
IJRCM

Subject: **SUBMISSION OF MANUSCRIPT IN THE AREA OF .**

**(e.g. Finance/Marketing/HRM/General Management/Economics/Psychology/Law/Computer/IT/Engineering/Mathematics/other, please specify)**

**DEAR SIR/MADAM**

Please find my submission of manuscript entitled '___________________________________________' for possible publication in your journals.

I hereby affirm that the contents of this manuscript are original. Furthermore, it has neither been published elsewhere in any language fully or partly, nor is it under review for publication elsewhere.

I affirm that all the author (s) have seen and agreed to the submitted version of the manuscript and their inclusion of name (s) as co-author (s).

Also, if my/our manuscript is accepted, I/We agree to comply with the formalities as given on the website of the journal & you are free to publish our contribution in any of your journals.

**NAME OF CORRESPONDING AUTHOR**:
Designation:
Affiliation with full address, contact numbers & Pin Code:
Residential address with Pin Code:
Mobile Number (s):
Landline Number (s):
E-mail Address:
Alternate E-mail Address:

**NOTES**:
a) The whole manuscript is required to be in ***ONE MS WORD FILE*** only (pdf. version is liable to be rejected without any consideration), which will start from the covering letter, inside the manuscript.
b) The sender is required to mention the following in the **SUBJECT COLUMN** of the mail:
**New Manuscript for Review in the area of (Finance/Marketing/HRM/General Management/Economics/Psychology/Law/Computer/IT/ Engineering/Mathematics/other, please specify)**
c) There is no need to give any text in the body of mail, except the cases where the author wishes to give any specific message w.r.t. to the manuscript.
d) The total size of the file containing the manuscript is required to be below **500 KB**.
e) Abstract alone will not be considered for review, and the author is required to submit the complete manuscript in the first instance.
f) The journal gives acknowledgement w.r.t. the receipt of every email and in case of non-receipt of acknowledgment from the journal, w.r.t. the submission of manuscript, within two days of submission, the corresponding author is required to demand for the same by sending separate mail to the journal.

2. **MANUSCRIPT TITLE**: The title of the paper should be in a 12 point Calibri Font. It should be bold typed, centered and fully capitalised.

3. **AUTHOR NAME (S) & AFFILIATIONS**: The author (s) **full name**, **designation**, **affiliation** (s), **address**, **mobile/landline numbers**, and **email/alternate email address** should be in italic & 11-point Calibri Font. It must be centered underneath the title.

4. **ABSTRACT**: Abstract should be in fully italicized text, not exceeding 250 words. The abstract must be informative and explain the background, aims, methods, results & conclusion in a single para. Abbreviations must be mentioned in full.

5. **KEYWORDS:** Abstract must be followed by a list of keywords, subject to the maximum of five. These should be arranged in alphabetic order separated by commas and full stops at the end.

6. **MANUSCRIPT**: Manuscript must be in ***BRITISH ENGLISH*** prepared on a standard A4 size ***PORTRAIT SETTING PAPER***. It must be prepared on a single space and single column with 1" margin set for top, bottom, left and right. It should be typed in 8 point Calibri Font with page numbers at the bottom and centre of every page. It should be free from grammatical, spelling and punctuation errors and must be thoroughly edited.

7. **HEADINGS**: All the headings should be in a 10 point Calibri Font. These must be bold-faced, aligned left and fully capitalised. Leave a blank line before each heading.

8. **SUB-HEADINGS**: All the sub-headings should be in a 8 point Calibri Font. These must be bold-faced, aligned left and fully capitalised.

9. **MAIN TEXT**: The main text should follow the following sequence:

   INTRODUCTION

   REVIEW OF LITERATURE

   NEED/IMPORTANCE OF THE STUDY

   STATEMENT OF THE PROBLEM

   OBJECTIVES

   HYPOTHESES

   RESEARCH METHODOLOGY

   RESULTS & DISCUSSION

   FINDINGS

   RECOMMENDATIONS/SUGGESTIONS

   CONCLUSIONS

   SCOPE FOR FURTHER RESEARCH

   ACKNOWLEDGMENTS

   REFERENCES

   APPENDIX/ANNEXURE

   It should be in a 8 point Calibri Font, single spaced and justified. The manuscript should preferably not exceed ***5000 WORDS***.

10. **FIGURES &TABLES**: These should be simple, crystal clear, centered, separately numbered &self explained, and **titles must be above the table/figure**. **Sources of data should be mentioned below the table/figure**. It should be ensured that the tables/figures are referred to from the main text.

11. **EQUATIONS**:These should be consecutively numbered in parentheses, horizontally centered with equation number placed at the right.

12. **REFERENCES**: The list of all references should be alphabetically arranged. The author (s) should mention only the actually utilised references in the preparation of manuscript and they are supposed to follow **Harvard Style of Referencing**. The author (s) are supposed to follow the references as per the following:

- All works cited in the text (including sources for tables and figures) should be listed alphabetically.
- Use (**ed.**) for one editor, and (**ed.s)** for multiple editors.
- When listing two or more works by one author, use --- (20xx), such as after Kohl (1997), use --- (2001), etc, in chronologically ascending order.
- Indicate (opening and closing) page numbers for articles in journals and for chapters in books.
- The title of books and journals should be in italics. Double quotation marks are used for titles of journal articles, book chapters, dissertations, reports, working papers, unpublished material, etc.
- For titles in a language other than English, provide an English translation in parentheses.
- The location of endnotes within the text should be indicated by superscript numbers.

**PLEASE USE THE FOLLOWING FOR STYLE AND PUNCTUATION IN REFERENCES:**

**BOOKS**

- Bowersox, Donald J., Closs, David J., (1996), "Logistical Management." Tata McGraw, Hill, New Delhi.
- Hunker, H.L. and A.J. Wright (1963), "Factors of Industrial Location in Ohio" Ohio State University, Nigeria.

**CONTRIBUTIONS TO BOOKS**

- Sharma T., Kwatra, G. (2008) Effectiveness of Social Advertising: A Study of Selected Campaigns, Corporate Social Responsibility, Edited by David Crowther & Nicholas Capaldi, Ashgate Research Companion to Corporate Social Responsibility, Chapter 15, pp 287-303.

**JOURNAL AND OTHER ARTICLES**

- Schemenner, R.W., Huber, J.C. and Cook, R.L. (1987), "Geographic Differences and the Location of New Manufacturing Facilities," Journal of Urban Economics, Vol. 21, No. 1, pp. 83-104.

**CONFERENCE PAPERS**

- Garg, Sambhav (2011): "Business Ethics" Paper presented at the Annual International Conference for the All India Management Association, New Delhi, India, 19–22 June.

**UNPUBLISHED DISSERTATIONS AND THESES**

- Kumar S. (2011): "Customer Value: A Comparative Study of Rural and Urban Customers," Thesis, KurukshetraUniversity, Kurukshetra.

**ONLINE RESOURCES**

- Always indicate the date that the source was accessed, as online resources are frequently updated or removed.

**WEBSITE**

Garg, Bhavet (2011): Towards a New Natural Gas Policy, Political Weekly, Viewed on January 01, 2012 http://epw.in/user/viewabstract.jsp

# EFFECTIVENESS OF SOFTWARE METRICS IN FINDING THE FAULTY CLASSES

*POOJA*
*M. TECH. STUDENT, UNIVERSITY SCHOOL OF I.T., GGS INDRAPRASTHA UNIVERSITY, DELHI*
*ASST. PROFESSOR*
*DRONACHARYA COLLEGE OF ENGINEERING*
*KHENTAWAS*

## ABSTRACT

*Low quality design leads to error-prone software that is difficult to understand, maintain and evolve. Thus, in order to improve its quality, software should be continuously inspected by examining the source code to identify potential flaws. Software metrics is a great tool in quality management of a complex software. This paper examines the effectiveness of software metrics in finding error-prone classes. The study is divided in two parts: - First part is the 'Implementation Part' which examined two different JAVA projects and study software metrics effectiveness in finding the error prone classes and analyze design of which project is better than the other. Second part is 'Survey Part' in which data was collected from various software firms and practical usage of software metrics in finding the faulty classes was studied.*

## KEYWORDS

Faulty Classes, Object Oriented Software Metrics, Software Metrics Effectiveness, Software Quality Management.

## 1. INTRODUCTION

**1.1** We are living in 'Machine Age'. Given any task today's technology can build a machine to fulfill it and to run that machine we need complicated software. As in case of advance robotics, air traffic control, nuclear plants, banking operations etc etc. In order to develop and maintain these complex software we need to measure the level of complexity and then require to maintain it. The tool for measurement is known as software metrics. According to *Fenton and Pfleeger (Fenton and Pfleeger, 1997)* "Measurement is the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to describe them according to clearly defined rules". The *IEEE* defines software metrics as "the quantitative measure of the degree to which a system, component, or process possess a given attribute" (*IEEE, 1990*). This means, that a software metric is a clearly defined rule, that assigns values to entities, that are part of a software system, e.g., components, classes or methods. Metric is the unit of measurement of software attributes like size, cost, time required, complexity etc. A lot of time and resource are required for the development of large software systems. So accurate planning and proper allocation of resources is mandatory for different software activities. Software metrics are necessary to identity where the resource is needed.

**1.2** To release a zero defect product is the dream of every developer. In order to achieve defect less product, companies spend 50 to 80% of their software development effort on testing. Therefore reducing testing effort may increase productivity, reduce cost and optimize resources. Software design is the backbone of software development life cycle. Identification of faulty-modules in design phase reveals an effective and efficient test plan execution. Software metrics are the measurement tools to be used to assess software products or related process. Object Oriented Programming involves complex processes (encapsulation, inheritance, coupling and cohesion). A common observation in complex software is that they have errors. Software metrics quantitatively measures the software and predict where errors are likely to occur thence make the work of software tester more effective and efficient.

**1.3** In the succeeding section software metrics evolution, types, features and threshold value concept is defined to have a better understanding of 'Effectiveness Of Software Metrics In Finding Error Prone Classes'.

## 2. SOFTWARE METRICS OVERVIEW

### 2.1 TYPES OF SOFTWARE METRICS

The Software Metrics can be divided on the basis of what it measures Fenton and Pfleeger divided software metrics into four categories *(Fenton and Pfleeger, 1997):*

I) Process Metrics: - It measures attributes of development of process. It is used to improve software development and maintenance

II) Product Metrics: - Product metrics describe the characteristics of the product such as size, complexity, design features, performance, and quality level

III) Resource Metrics: - Resources Metrics measures the resources, that were utilized as part of a process. Like people, hardware, or software needed for the processes

IV) Project Metrics: - It measures overall productivity of project team like percentage of project completed in given time frame.

### 2.2 SOFTWARE METRICS EVOLUTION CRITERION

Several researchers have recommended properties that software metrics should possess to increase their usefulness.' *Weyuker*' has developed a formal list of desiderata for software metrics and has evaluated a number of existing software metrics using these properties. He has given nine properties that provide a language for software metrics development. These are Noncoarseness, Granularity, Nonuniqueness (Notion of Equivalence), Design Details are Important, Monotonicity, Nonequivalence of Interaction, Permutation, Renaming Property and Interaction Increases Complexity.

The Metrics Evolution Criterion consists of following five models: -

I) Unit Definition: - A unit is defined for all measures including ratio, scale, nominal, and ordinal. There are four types of the unit definition models - reference to a standard, reference to a wider theory, reference to conversion from another unit, and reference to a model involving several attributes.

II) Instrumentation: - An instrumentation model determines the method to capture a measure. Instrumentation model is closely related to the unit definition model. The instrumentation model is used to take the measurements. There are two types of instrumentation models: the direct representational model and the indirect theory-based model.

III) Attribute Relationship:- An attribute is composed of other attributes; the attribute relationship model defines the relationship among the attributes. There are two types of attribute relationship models: definition and predicative.

IV) Measurement Protocol: - A measurement protocol model is concerned with how to measure an attribute consistently on a specific entity. Measurement protocol model determines the measurement method so that the measure of a specific attribute on a specific entity is consistent and repeatable.

V) Entity Population: - The Entity population model sets the normal values of a metric.

### 2.3 FEATURES OF A GOOD SOFTWARE METRICS

A good software metrics must include theoretical base, empirical properties, must be practical (i.e. relevant to practitioners in organizations / institutes), must be evaluated, cost effective and robust (Robust means it must be relatively insensitive to (intuitively) insignificant changes in the process or product)

## 3. THRESHOLD VALUE

Thresholds are a simple method to separate values. The values that are greater than a threshold value are considered to be problematic, the values below are okay. Thus, by defining thresholds a simple analysis of measured values is possible. This mechanism can also be applied to software metrics. For example, by defining a threshold for a metric that measures the size of an entity, all metric values that are above the threshold mark the entity as too large. Thresholds for

software metrics are often used in the context of fault-proneness. Thresholds can also be used to define other aspects as problematic, e.g., the maintainability or the understandability. For simplicity, we assume that thresholds are upper bounds.
Let m a metric with threshold t that defines a lower bound, i.e., entities x are considered to be problematic if m(x) < t. This is equivalent to 1/m(x) > 1/t if m(x) and t are non-negative, as metrics and thresholds usually are. By defining a new metric m'(x) = 1/m(x) and a new threshold t' =1/t a new metric with the opposite order is defined and with t' a threshold is obtained that defines an upper bound.
Depending on the organization, the programming language, the tools used, the qualification of the developers and other factors that are project dependent a different threshold value might be better. This is a problem, as each organization – and maybe even each project – has to define thresholds that are chosen depending on its environment.

## 4. SOFTWARE METRICS AND ITS EFFECT
In this section some metrics for classes are defined along with their effect in finding the error proneness.
### 4.1 WEIGHTED METHODS PER CLASS (WMC)
WMC is a measure for the complexity of a class. The complexity of a class is measured indirectly using the sum of the complexity of its methods. For a class C that defines the set of methods M WMC is computed as

$$WMC(C) = \sum_{m \in M} complexity(m)$$

The WMC metric is intended to measure the combined complexity of a class' local methods
Effect of WMC in finding the faulty classes: -
1) The number of methods and the complexity of methods involved is a predictor of how much time and effort is required to develop and maintain the class.
2) The larger the number of methods in a class the greater the potential impact on children, since children will inherit all the methods defined in the class.
3) Classes with large numbers of methods are likely to be more application specific, limiting the possibility of reuse.
Threshold value for WMC is 100
### 4.2 CLASS METHOD COMPLEXITY (CMC)
The CMC metric is the summation of the internal structural complexity of all local methods, regardless whether they are visible outside the class or not (e.g. all the public and private methods in C++). The CMC metric captures the complexity of information hiding in the methods of a class. This attribute is important for the creation of the class in an OO design because the complexity of the information hiding gives an indication of the amount of effort needed to design, implement, test, and maintain the class.
Effect of CMC in finding the faulty classes:-
1. The CMC metric is directly linked to the effort needed to design, implement, test, and maintain a class. The more complex a class' methods are, the more effort is needed to design, implement, test, and maintain the methods.
2. The more complex a class' methods are, as measured by the internal complexity of the methods, the more effort is needed to comprehend the realization of information hiding in a class.
### 4.3 NUMBER OF LOCAL METHODS (NLM)
The NLM metric is the number of the local methods defined in a class which are accessible outside the class (e.g. public methods in C++). The total number of local methods that can be directly invoked from outside the class is the value of the NLM metric.
Effect of NLM in finding the faulty classes: -
1. The NLM metric is directly linked to a programmer's comprehension effort when a class is reused in an OO design. The more local methods a class has, the more effort is required to comprehend the class' behavior.
2. The larger the local interface of a class, the more effort is needed to design, implement, test, and maintain the class.
3. The larger the local interface of a class, the more influence the class has on its descendent classes.
The unit for the NLM metric is ``method''.
### 4.4 COUPLING BETWEEN OBJECTS (CBO)
Coupling describes the interdependence between modules. CBO relates to the notion that an object is coupled to another object if one of them acts on the other, i.e., methods of one use methods or instance variables of another. As stated earlier, since objects of the same class have the same properties, two classes are coupled when methods declared in one class use methods or instance variables defined by the other class. Coupling can be done through inheritance, abstract data type and message passing.
Effect of CBO in finding the faulty classes: -
1) Excessive coupling between object classes is detrimental to modular design and prevents reuse. The more independent a class is, the easier it is to reuse it in another application.
2) In order to improve modularity and promote encapsulation, inter-object class couples should be kept to a minimum. The larger the number of couples, the higher the sensitivity to changes in other parts of the design, and therefore maintenance is more difficult.
3) A measure of coupling is useful to determine how complex the testing of various parts of a design is likely to be. The higher the inter-object class coupling, the more rigorous the testing needs to be.
### 4.5 LACK OF COHESION METHOD (LCOM)
The LCOM is a count of the number of method pairs whose similarity is 0 (i.e., σ () is a null set) minus the count of method pairs whose similarity is not zero. The larger the number of similar methods, the more cohesive the class. If none of the methods of a class display any instance behavior, i.e., do not use any instance variables, they have no similarity and the LCOM value for the class will be zero. The LCOM value provides a measure of the relative disparate nature of methods in the class. A smaller number of disjoint pairs (elements of set P) implies greater similarity of methods.
Effect of LCOM in finding the faulty classes: -
1) Cohesiveness of methods within a class is desirable, since it promotes encapsulation.
2) Lack of cohesion implies classes should probably be split into two or more subclasses.
3) Any measure of disparateness of methods helps identify flaws in the design of classes.
4) Low cohesion increases complexity, thereby increasing the likelihood of errors during the development process
### 4.6 RESPONSE FOR A CLASS (RFC)
This metric measures the size of the response set of a class. The response set consists of all methods that can be invoked by calling a method from the class.
The response set for the class can be expressed as : -

$$RS = \{M_i\} \cup_{all\ i} \{R_i\}$$

where $\{R_i\}$= set of methods called by method i and
$\{M\}$ = set of all methods in the class.
Effect of RFC in finding the faulty classes: -
1) If a large number of methods can be invoked in response to a message, the testing and debugging of the class becomes more complicated since it requires a greater level of understanding required on the part of the tester.
2) The larger the number of methods that can be invoked from a class, the greater the complexity of the class.
3) A worst case value for possible responses will assist in appropriate allocation of testing time
Threshold value for RFC is 100

**4.7 DEPTH OF INHERITANCE TREE (DIT)**

Depth of inheritance of the class is the DIT metric for the class. In cases involving multiple inheritance, the DIT will be the maximum length from the node to the root of the tree.DIT is a measure of how many ancestor classes can potentially affect this class. By observing the DIT metric for classes in an application, a senior designer or manager can determine whether the design is "top heavy" (too many classes near the root) or "bottom heavy" (many classes are near the bottom of the hierarchy).

Effect of DIT in finding the faulty classes: -

1) The deeper a class is in the hierarchy, the greater the number of methods it is likely to inherit, making it more complex to predict its behavior.
2) Deeper trees constitute greater design complexity, since more methods and classes are involved.
3) The deeper a particular class is in the hierarchy, the more methods and classes are involved.
Greater the potential reuse of inherited methods.
4) Designing a class is a relatively simple task, but the testing could become more complicated due to the high inheritance.

Threshold value for DIT is 6

**4.8 NUMBER OF CHILDREN (NOC)**

NOC = number of immediate subclasses subordinated to a class in the class hierarchy.NOC relates to the notion of scope of properties. It is a measure of how many subclasses are going to inherit the methods of the parent class.

Effect of NOC in finding the faulty classes: -

1) Greater the number of children, greater the reuse, since inheritance is a form of reuse.
2) Greater the number of children, the greater the likelihood of improper abstraction of the parent class. If a class has a large number of children, it may be a case of misuse of subclassing.
3) The number of children gives an idea of the potential influence a class has on the design. If a class has a large number of children, it may require more testing of the methods in that class.

Threshold value for NOC is 6

**4.9 NUMBER OF ATTRIBUTES (NOA)**

The Number Of Attributes metric is used to count the average number of attributes for a class in the model. Hence it is used to measure size of class. Attributes are the data members / variables / fields which stand for stored information.

Effect of NOA in finding the faulty classes:-

1) NOA measures class size. Hence larger the number of Attributes in a class the larger the no. of methods used thence the testing and debugging of the class becomes more complicated since it requires a greater level of understanding required on the part of the tester. So, the greater the complexity of the class.
2) It is desirable that classes communicate with as few others as possible and even then, that they exchange as little information as possible". So, larger the NOA value the more is complexity during coupling and inheritance .Hence it limits understandability and maintainability.
3) Attributes with Conflicting Names" i.e. In some cases overriding of attributes exists ( If a subclass defines an attribute with the same name as a visible attribute of a its superclass).In such cases complexity of class increases. So, NOA should be kept low. A class with too many attributes may cause coincidental cohesion.

Threshold value for NOA is 20

## 5. IMPLEMENTATION WORK

In Object Oriented Programming inheritance among classes and the size of class are the main criterion to determine the complexity of a class. This section focuses on two metrics and their effect in finding the Faulty Classes i.e. Number Of Attributes (NOA) and Depth Of Inheritance Tree (DIT).

Two different JAVA projects (which in this paper will be referred as 'Project A' and 'Project B') were examined and analyzed through JAVA Codes for DIT and NOA metrics (i.e. DITFinder.java and NOAFinder.java respectively) .The analytical evaluation is as follow : -

**5.1 DATA COLLECTION AND INTERPRETATION FOR DEPTH OF INHERITANCE TREE (DIT)**

The Java Code DitFinder was executed on both Projects and the observations are as follow: -

**TABLE 1: OUTPUT OF Ditfinder.java FOR PROJECT A**

| DIT | No. Of Classes |
|---|---|
| 1 | 41 |
| 2 | 14 |
| 3 | 6 |
| 4 | 3 |
| 5 | 29 |
| 6 | 25 |
| Total | 118 |

**FIGURE 1: HISTOGRAM OF DIT METRICS FOR PROJECT A**

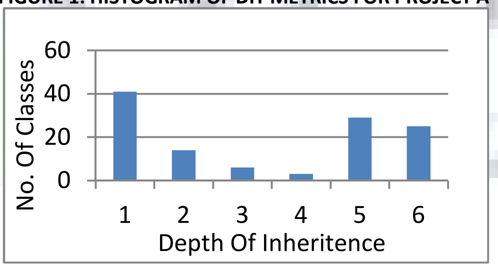
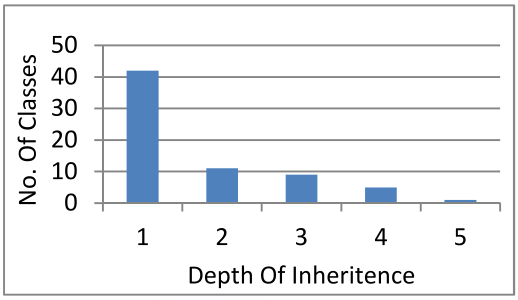



**TABLE 2: OUTPUT OF Ditfinder.java FOR PROJECT B**

| DIT | No. Of Classes |
|---|---|
| 1 | 42 |
| 2 | 11 |
| 3 | 9 |
| 4 | 5 |
| 5 | 1 |
| Total | 68 |

FIGURE 2: HISTOGRAM OF DIT METRICS FOR PROJECT B




As concluded from Table 1 and Table 2 the maximum value for DIT for 'Project A' is 6 and for 'Project B' is 5, respectively. Hence both are within threshold. So there exist no error-prone classes.

**5.2 DATA COLLECTION AND INTERPRETATION FOR NUMBER OF ATTRIBUTES (NOA)**

The Java Code NOAFinder was executed on both Projects and the observations are as follow: -

**TABLE 3: OUTPUT OF Noafinder.java FOR PROJECT A**

| NOA | No Of Classes | NOA | No Of Classes |
|---|---|---|---|
| 0 | 29 | 17 | 1 |
| 1 | 12 | 18 | 2 |
| 2 | 16 | 19 | 1 |
| 3 | 10 | 20 | 2 |
| 4 | 1 | 22 | 3 |
| 5 | 1 | 23 | 1 |
| 6 | 3 | 25 | 1 |
| 7 | 5 | 38 | 1 |
| 8 | 3 | 55 | 1 |
| 9 | 4 | 57 | 1 |
| 10 | 1 | 60 | 1 |
| 11 | 6 | 63 | 1 |
| 13 | 1 | 64 | 1 |
| 14 | 3 | 69 | 1 |
| 15 | 2 | 83 | 1 |
| 16 | 2 | Total | 118 |

FIGURE 3: HISTOGRAM OF NOA METRICS FOR PROJECT A

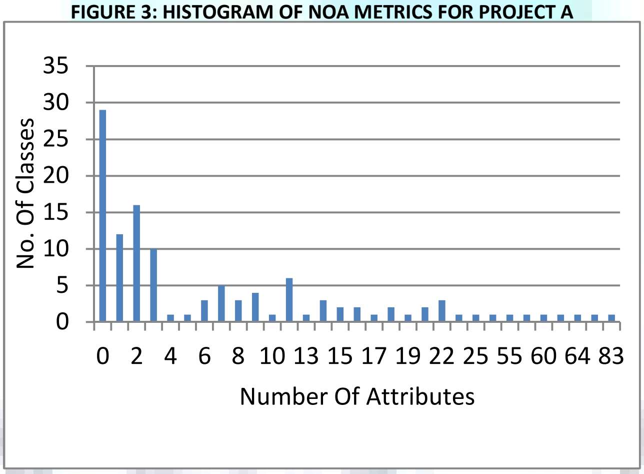
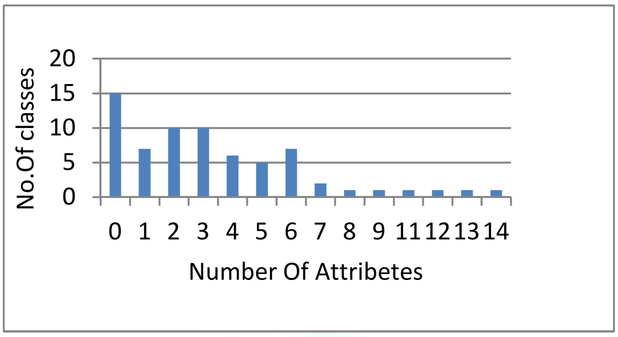



**TABLE 4: OUTPUT OF Noafinder.java FOR PROJECT B**

| NOA | No.Of Classes |
|---|---|
| 0 | 15 |
| 1 | 7 |
| 2 | 10 |
| 3 | 10 |
| 4 | 6 |
| 5 | 5 |
| 6 | 7 |
| 7 | 2 |
| 8 | 1 |
| 9 | 1 |
| 11 | 1 |
| 12 | 1 |
| 13 | 1 |
| 14 | 1 |
| Total | 68 |

FIGURE 4: HISTOGRAM OF NOA METRICS FOR PROJECT B




As observed in Table 3 'Project A' 13 classes are having NOA greater than 20 i.e. greater than threshold value. So, these are error prone classes and hence must be redesigned.

For 'Project B' as it can be observed from Table 4 $NOA_{max}$=14 i.e. below threshold. So, no alteration in code is required. Hence no faulty class found.

Comparing Figure 3 and Figure 4 it can be concluded that design of 'Project B' is better than 'Project A' and 'Project A' can be re-designed to remove the error prone classes .

Hence using NOAFinder.JAVA and DITFinder.JAVA (NOA and DIT metrics) 13 Faulty Classes were found and it was observed that design of 'Project B' is better than 'Project A'.

## 6. INDUSTRIAL SURVEY

To observe the practical application of software metrics in finding the faulty classes a questionnaire was prepared and data was collected from 45 software developers from 3 firms (15 each firm) i.e. a) Tata Consultancy Services, Gurgaon , b) IBM Gurgaon , c) HCL Noida. Through questionnaire data was collected about the type of metrics used in the organization, major advantage of using metrics, its effectiveness in resource planning, budgeting, scheduling, process improvement, staff performance and finding the faulty classes .After analyzing data it was observed that all three firms rely on Software Metrics for its software development and its a very cost effective tool to measure the productivity of project team. 42 out of 45 developers (93.33 %) Strongly Agree/Agree that software metrics is an effective tool in finding the faulty classes. 1 Developer is neutral on its Effectiveness and only 2 Developers disagree/strongly disagree on its effectiveness in finding fault prone classes. Software metrics is used in almost every software firm to develop fault prone software.

## CONCLUSION

It was investigated that whether the object oriented metrics could predict the fault prone probability in the classes and through implementation (13 faulty classes were found using software metrics) and industrial survey (42 out of 45 i.e. 93.33% developers agree that software metrics is a great tool in finding the error prone classes) it was observed that-software metrics is an effective tool in finding the faulty classes. Hence in present World where our daily life relies on complex software like banking, air traffic controls etc where flaws or defects costs a lot , software metrics is a great tool to control quality of large complex program and must be applied to each and every complex software for fault prone ,easily maintainable and cost effective development.

## REFERENCES

**JOURNALS**

1. Chidamber S.R. and Kemerer C.F., "*Towards a Metrics Suite for Object-Oriented Design*," IEEE Transactions On Software Engineering, Vol. 20, No. 6, June 1994
2. Khan R.A. and Mustafa K.: *Fault Proneness Model for Object-Oriented Software: Design Phase Perspective,* Information Technology Journal 7: 698-701, 2008
3. Li W.: *Another metric suite for object-oriented programming*, The Journal of Systems and Software 44 pages 155-162, 1998

**PAPERS**

4. Abreu F.B. and Carapuça R. : *Object-Oriented Software Engineering : Measuring and Controlling the Development Process*, 4th Int. Conf. on Software Quality", McLean, VA, USA, 3-5,October 1994
5. Alikacem E.H., Sahraoui H., *Rule-Based System for Flaw Specification and Detection in Object-Oriented Programs*, 13th TOOLS Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE+), 2010.
6. Benlarbi S., Emam K.,Goel E., and Rai S.N.: *Thresholds for Object-Oriented Measures,* National Research Council Canada Institute for Information Technology ERB-1073 ,Mar 2000
7. Beyer D., Lewerentz C., and Simon F.: *Impact of Inheritance on Metrics for Size, Coupling, and Cohesion in Object-Oriented Systems ,* Software Systems Engineering Research Group Technical University Cottbus, Germany 2001
8. Herbold S., Grabowski J., and Waack S.: *Calculation and Optimization of Thresholds for Sets of Software Metrics,* Technical Report No. IFI-TB-2010-01, ISSN 1611-1044, 2010
9. Mills E.E.: *Software Metrics*, SEI Curriculum Module SEI-CM-12-1.1, Dec 1988

# REQUEST FOR FEEDBACK

**Dear Readers**

At the very outset, International Journal of Research in Computer Application and Management (IJRCM) acknowledges & appreciates your efforts in showing interest in our present issue under your kind perusal.

I would like to request you to supply your critical comments and suggestions about the material published in this issue as well as on the journal as a whole, on our E-mails i.e. **infoijrcm@gmail.com** or **info@ijrcm.org.in** for further improvements in the interest of research.

If you have any queries please feel free to contact us on our E-mail infoijrcm@gmail.com.

I am sure that your feedback and deliberations would make future issues better – a result of our joint effort.

Looking forward an appropriate consideration.

With sincere regards

Thanking you profoundly

**Academically yours**

Sd/-

**Co-ordinator**

# ABOUT THE JOURNAL

In this age of Commerce, Economics, Computer, I.T. & Management and cut throat competition, a group of intellectuals felt the need to have some platform, where young and budding managers and academicians could express their views and discuss the problems among their peers. This journal was conceived with this noble intention in view. This journal has been introduced to give an opportunity for expressing refined and innovative ideas in this field. It is our humble endeavour to provide a springboard to the upcoming specialists and give a chance to know about the latest in the sphere of research and knowledge. We have taken a small step and we hope that with the active co-operation of like-minded scholars, we shall be able to serve the society with our humble efforts.

## Our Other Journals