ISSN: 2348 - 2117

International Journal of Engineering Technology and Computer Research (IJETCR)
Available Online at www.ijetcr.org
Volume 3; Issue 4; July-August-2015; Page No. 193-198

Reliability Assessment and Feasibility Study of Software Metrics in Object Oriented Environment.

Aditya Kaushik[1], Sachin Jain[2]

1Research Scholar, SGVU, Jaipur.

2Assistant Professor, SGVU, Jaipur.

adityakaushik413@yahoo.in

Abstract

Software is the set of codes prepared by the developers in particular development environment. Software is intended to meet the ad- hoc purpose of the clients (who ever uses it). There are different phases for the development of a software viz. elicitation, analysis, coding, testing, implementation etc. These all phases mentioned are followed in sequence by the development team for preparing software.

We can measure the software in terms of its effectiveness in fulfilling our requirements by Software metrics. With the concept of software metrics only, Software quality is ensued. The metric helps the different stakeholders to perform the feasibility analysis of the softwares.

There has been a shift in pattern of our coding style from time to time i.e. previously, it was the procedural. Now it has been replaced by the OO. So, we are required to focus on to the new enhanced suites which have maximum efficiency. This paper focuses on the feasibility analysis of the software metrics along with the proposal of the new metrics suite.

Keywords: Elicitation, OO, metrics, UML etc

I. INTRODUCTION:

Computers and Software, no doubt, are one of the biggest innovations in the history of mankind. Software contributes to a fast, reliable, less effort taking and high computation environment in which the output and computation totally and automatically depends on the machines. Software Engineering is considered to be the science of Development and Maintenance of the software. We oftenly use different types of SDLC cycle for the development purpose.

- Waterfall model.
- Spiral model.
- Prototype model.
- Incremental model.
- RAD model etc.

The software is required to fulfill the different criteria as per the software quality assurance (SQA). Software quality is the feature of the software which tells about the aptness of the given software and helps to generate the software project with higher precision and efficiency. Quality control is the sort of iterative and repeated mechanism to analyse the present behavior of the system to the expected behavior/outcome.

Software metrics is a tool which is used to measure the efficiency of the s/w and the extent to which the software meets the expectation of the users. There are generally 3 kinds of metrics:

- Project metrics
- Progressive metrics
- Productivity metrics

There are a few kinds of the metrics which has been considered in the course of the study and Research. Those are as follows:

- KC metric suite.
- RC martin metric suite.
- McCabe metric suite.

Defect/ fault/bugs/errors can be verified based on the different testing technique. These can be categorized into white box technique and the black box testing technique.

In white box testing, we test any software by providing input to them & and generating the output. These outputs are then matched with the expected outcome. No internal structure is checked. The white box testing is yet another type of testing activity in which the internal structure of the code is checked.

There is certain more technique available to us which are written below:

- Unit testing.
- Integration testing.
- System testing.

In the unit testing, the individual smallest unit is tested and verified. In integration testing, the different modules are combined and tested in the different fashion. In

Corresponding author: Aditya Kaushik

system testing there are different pattern of testing like functional testing, acceptance testing etc.

II. LITERATURE SURVEY

There has been a variety of forms in which the research work in the software testing has been done.

These mutants can also be verified or checked through the use of the UML diagrams without going for the code review and can be used for the consistency checking very effectively [l]. An entire deposit of the test-suite can be generated or developed and analyzed by the use of the Genetic algorithm [2]. the process of validation of the software as well as enhancing the functionality in quite less amount of time. This may help us as the "software recommend tool" for the process of verification [3]. . There are 2 distinct 1ocal searches which can be used in union with the AVM are the "Geometric search" and the "lattice Search" approach [4]. The study and research done by *Kishor lukke et. al*. states about the graphical features and the assets of the softwares [5]. There are certain practices which minimize the effort in the again making and preparation of the test-cases even if the model is been altered [6]. These are the time, space and monetary issue factors which are very much desired to be enhanced and improvised for the better project prospect. This can be done through the hi-graph concept and the allied technique [7] which was proposed by *Hycham aboutaleb* in 20l5. choosing of the different metric suite that are in the architectural pattern, metric suite choosing, efficiency enhancement and the modeling [8]. ANN contains detail explained directed-graph which is highly keen to bring out outcome or the outputs [9]. the analytical review applying to the diverse concepts of the soft-computing technique for checking and verifying the different software components [l0]. *Alberto* illustrates about the obtaining or generating the technical worth for universal metrics [11]. Applications made or developed is evolved, scalable & enriched with the higher level of association along with user and functionality [l2]. KNN procedure residue in general unaltered with escalating number of communicating predictors and concurrently gives better-quality performance more widely applied many distinct rectilinear regressions [l3]. As far as the prediction of the software's fault judgment in advance can be done on the basis of the parameter like linear-regression, ANN, logical-regression [l4]. A portion of work done by the numerous researchers on the substantiation of the usefulness of the Object based system. *Yeresime Suresh et. al*. assumed many orthodox metrics & Object-based metrics and evaluated it using apt techniques and captured in sequence for the developers, coders and testers [l8].

III. PROPOSED METHODOLOGY

In the above mentioned topic, we have to establish the reliability and feasibility assessment using the different metric suites available. Our focus will be to deal with the classes and objects and check for the quality prediction and analysis of the object oriented software.

There are 2 things which have been done in this research area. Firstly, we studied and analysed the above three mentioned metric suites and after that we proposed a metric suite.

KC metric suite:

This suite was proposed by the *Kermerer and Chidamber* in 1994. This deals with the OO system which consists of the following metrics for the verification of the effectiveness of the software projects:

- WMC
- DIT
- RFC
- NOC
- CBO
- LCOM

WMC stands for the "Weighted method per class". It is the sum of all the weights or the complexity of the given software code.

DIT is "Depth of inheritance tree. It is the longest chain of the inheritance hierarchy.

RFC is Response from the class.

NOC is no of the children. It is the total no of sub classes which inherits from a base class.

CBO is the coupling between the objects. It tells about the degree to which the different objects are coupled together.

LCOM is lack of cohesion in methods.

R. C Martin metric suite:

This metric suite was proposed by the Robert Cecil Martin. This metric suite deals with the following metrics:

- Afferent coupling (Ca)
- Efferent coupling (Ce)
- Instability
- Abstractness
- Normalized distance from main sequence.

Afferent coupling is the no of the external classes that depends upon a certain class within a different package.

Efferent coupling is the no of the classes within any package that depends upon a certain class within a different package.

Instability is the mathematical proportion of the Efferent to the total no of coupling.

Abstractness is the total number of abstract classes compared to the number of the other classes in an evaluated package.

McCabe metric suite:

- Cyclomatic complexity
- Size
- Comment percentage.

Cyclomatic complexity is the entity which tells about the functional behavior of the system. It tells about the total feasible and executable path.

Size is the constraint which tells about the overall dimension of the software in terms of the no. of lines of codes.

Comment percentage is the metric which is evaluated by the fraction no. of comments to loc. and from this fraction number of the blank line is subtracted.

IV. SIMUTION AND RESULT

The formulae used in our work were to calculate the above mentioned parameters are as follows:

- ***Code coverage= No of unit tested / Total size.***

Code coverage is the total amount of the code which has been verified or tested by applying the different criteria. It is used to measure the discrepancy in passing of the jump statements, iterations, loops etc.

- ***Test effectiveness = defect/(defect + defects found during user acceptance testing)***

Test effectiveness is used to measure the efficiency of the code. It deals with the total no of the defects in the codes and the defect found during the acceptance testing (α and β testing).

- ***Defect density = total defect / size in loc***

Defect density deals with the total amount of the defect encountered in per unit size of the code. It deals with the amount of concentration of the defect in the softwares

- ***Defect ratio = defect / F.P.***

It is the mathematical formula to quantify the amount o the defect in the comparison of the function point. It is the important term which may tell about the concentration of the defect in respect to the F.P.

- ***Reliability = Mean time b/w failure- (total operating time/no. of failure)***

It is one of the most important concepts which tells that to what extent the system can be trustworthy. It deals with the concept of defect and the compilation time.

- ***defect removal efficiency= no. of defect during testing (DT)/DT + defect after delivery***

It tells about to what extent the defect present in the system can be removed. It deals with the defect while testing as well as the defect after the delivery.

Cyclomatic complexity= E - N + 2

ANALYSIS OF METRIC SUITES

KC metric suite:

Metrics	Input
WMC	8
CBO	3
DIT	3
NOC	1
LCOM	0

RC Martin metric suite:

Metrics	Input 1
Afferent Coupling	2
Efferent coupling	4
Instability	0.34
Abstractness	0

Mc Cabe Metric suite:

Metrics	Input
Cyclomatic complexity	1
Size in LOC	60
Method lines of code	10

Proposed metric suite:

Metrics	Input
Code coverage	0.98
Test effectiveness	1
Defect Density	0.016
Defect Ratio	0.33
Reliability	0.5
DRE	0.5
Cyclomatic complexity	1

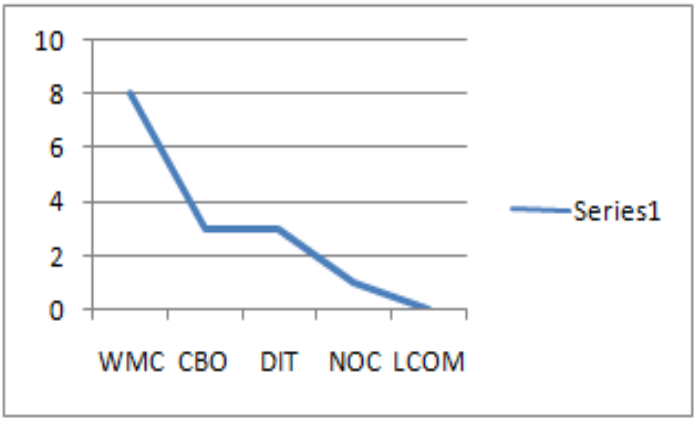

Fig: KC Metric suite

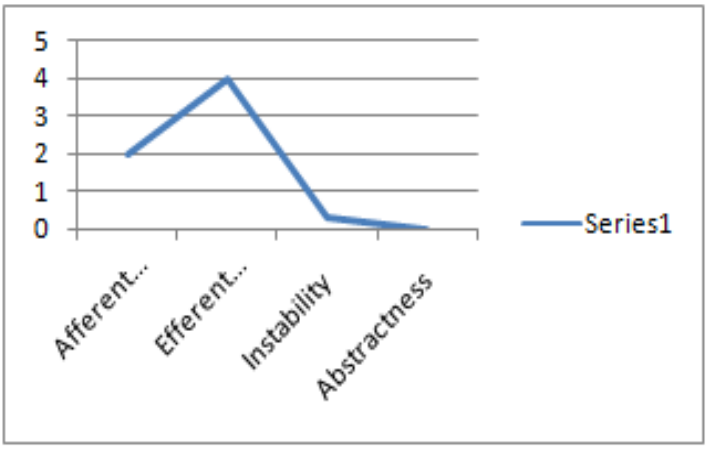

Fig: RC Martin metric suite

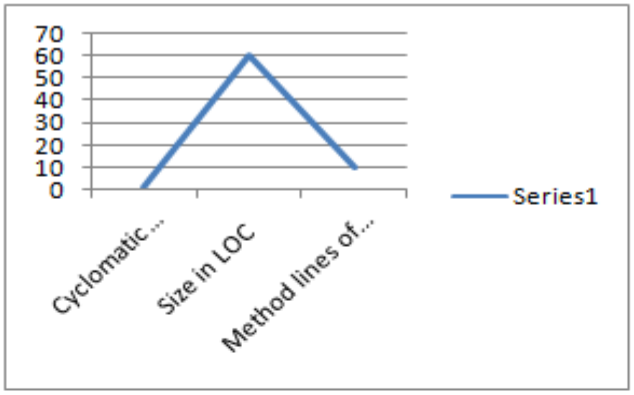

Fig: McCabe metric suite

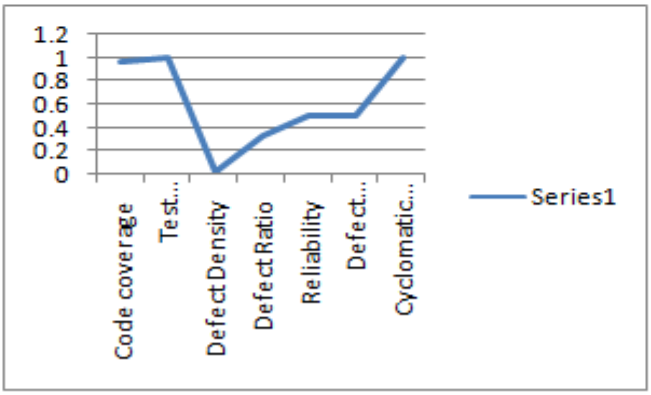

Fig: Proposed metric suite

V. CONCLUSION

The entire above metric suite doesn't deals with the defect in any sort of manner. These are the metric suite does can't find the reliability constraint. Through this, we can't access that the whole of the system at one time. The both of the metrics can't analyse the system to the concept of how the code is being covered or executed.

The proposed metric suite can tell about the details of the defects, bugs, any infinite-loops, processing time etc. it tells about the total time taken to execute the program etc. The term reliability consists of these distinct and different patterns. The software has the multiple dimensions from which it can be analyzed. The pattern which is being observed from the above mentioned work (all 3 metric suites mentioned above) related to it. The proposed metric suite can play major role in the analysis of the software from the different dimension like amount of code (size), functionality, faults/ bugs/ error. It also identifies the error-prone zone of any of the given software. This has the importance in the analysis of the software regarding the user's perspective and views. Overall, it testifies the software relating to the proper configured software and obtains the reliability of the software along with the defect elimination procedures.

VI. REFERENCE

1. Lei M. et. al. A method of software specification mutation testing based on UML for consistency checking- Procedia Engineering 15 (2011) 110 – 114
2. Gordon F. et. al. A memetic algorithm for whole test case suite generation- Elsevier-The Journal of Systems and Software 103 (2015) 311–327
3. Marina P. et. al. Semi –automated tool recommender for software development process-Elsevier Electronic Notes in Theoretical Computer Science 302 (2014) 95–109
4. Joseph K. et. al. Design and analysis of different alternation variable searches for search based software testing- Elsevier B.V 2014
5. Lukke K. et. al. Application of graph theory to OO software engineering for development of Metric suite- Acedemia.edu 2015
6. Gordon F et. al. Handling model changes regression testing and test suite update with model checkers- Elsevier Electronic Notes in Theoretical Computer Science 190 (2007) 33–46

7. Hycham A. et. al. Measuring the complexity of a high graph based system model formalism and metrics- Elsevier, Procedia Computer Science 44 (2015) 11 – 20.2015
8. Andrei V. et. al. Software architecture & detailed designed evaluation- Elsevier Procedia Computer Science 43 (2015) 41 – 52
9. Aggarwal G et. al. Neural network approach to measure reliability of software modules a review- IJAES, Vol 2, issue 3, april 2013
10. Gupta D et. al. Comparative study of soft computing technique for software quality model-vol 1, issue 1, jan 2011
11. Mitchell A. et. al. Relationship between static and dynamic coupling metrics- Elsevier B.V.2006
12. Alberto N. et. al. A Methodology for obtaining universal code metrics- Procedia Technology 7 (2013) 336 – 343
13. Goyal R. et. al. Suitability of KNN regression in the development of interaction based software fault prediction models- IERI Procedia 6 (2014) 15 – 21.
14. Yeresime S. et. al. Statistical and machine learning methods for software fault prediction using CK metrics- a comparative analysis- ISRN Software Engineering Volume 2014 (2014), Article ID 251083, 15 pages
15. Chidamber S. et. al. Metrics for object oriented design-IEEE transaction on software engineering vol 20, no 6, june 1994
16. Khan A. et. al. An empirical validation of object oriented design quality metrics- J. King Saud Univ., Vol. 19, Comp. & Info. Sci., pp. 1-16, Riyadh (1427H./2007)2006
17. Jeddah al dallal, Transitive based object oriented lack of cohesion metrics- Procedia Computer Science 3 (2011) 1581–1587
18. Yeresime S. et. al. Effectiveness of software metrics for object oriented system- Procedia Technology 6 (2012) 420 – 427