Volume 5, Issue 2, February 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Quality of Service Analysis of Software Reliability and Testability Using a New Dynamic Metric

Vaid Singh, Charnpreet Kaur
RIMT, Mandi Gobindgarh,
Punjab, India

Abstract— ***Today it is very essential to ensure quality of service of software for meeting user and customer requirements. Quality of service is analyzed by software metrices. Dynamic metrices give better results as compared to static metrices and are used to calculate performance during run time behaviour of system. As the technology changes there is the need for adding some new features to the system for competing with modern technology. when some new features like reliability and testability are added into a software then these may be conflict with existing features like performance ,installibility,documentation,maintainability,availiability.Genetic algorithm is used to overcome this problem. Genetic algorithm gives the best population according to fitness function from the set of existing population. It finds the optimal solution for the problem occurred by the conflict between new features and existing features .optimal solution results increasing performance of the system and the quality of the end product.***

Keywords— Quality, Testability, Reliability, Software, Static, Dynamic

I. Introduction

1.1Quality

Quality is an important factor in market. In the past few years we have seen a significant change in the field of software development. There is an increasing use of the object-oriented engineering for the software development. Introduction of software metrics helps to know about different aspects of software, internal as well as external quality aspects. Software metrics provides effective information about external aspects of software such as reliability, availability, performance and also helps in effort estimation required to develop software and in testing. As the use of software metrics increase, the needs of the customer also increase. Now days, the customers are ready to pay more money for the good quality product. The quality of software increases with the removal of bugs. As early as the bugs are found, the quality of the software can be improved.

1.2 Software Quality

Software Quality is the measure to which a process of method and components of a system meets the requirements that are already specified. We can also say that in which it can meet customers or users requirements also. It follows two types of criteria. Internal Criteria, in which it is not visible to the user and it is code-dependent and is for developer only. Then we have external criteria which is an experience in operational mode by users when running the software**.**

1.3 Measuring a Software

There are number of reasons which motivate us to measure software. Measuring software helps us to.

- examine the cost and effort
- To improve production of software.
- To improve the quality of software and to increase reliability of the system.
- Reduce future Maintenance needs.
- To improve the development process so as to increase product quality.
- In order to satisfy the customer needs and expectations.
- We need measurement on software so that we can understand and agree on product quality.
- To make better estimation of cost, schedule, complexity and effort needed for software development process.

1.4 Software Quality Metrics

Software metrics explains the activities related with measurements in software engineering. The metrics are applied to software development process and the product so as to get the meaningful information. Software metric is the measurement of some properties of software. Software metrics can be classified into the three categories. First is the, Product metrics which describes the characteristics of the product, these characteristics includes size of product, complexity of the system, design aspects, performance as well as the quality level. Secondly we have Process metrics that can be used to improve software development and maintenance. Lastly, third category is Project metrics which describes the project characteristics and execution. The software metrics are more close to the product and process metrics than the project metrics.

a. Product quality metrics includes four types of metrics, mean time to failure, defect density, customer problems and customer satisfaction. The metrics mean time to failure and defect density is the intrinsic product quality metrics. These measure the number of "bugs" or in other words number of "defects" in software. Customer problem metrics measures the number of problems that are faced by the user when he operates the software. The customer satisfaction metric works on five point scale to know satisfaction level of the user, that is, whether a customer is very satisfied, satisfied, neutral, dissatisfied or very dissatisfied and has weighing factors 100%, 75%, 50%, 25% and 0% respectively.

b. Process quality metrics, comprises of metrics as defect density during machine testing, defect arrival pattern during machine testing, phase based defect removal pattern, defect removal effectiveness.

c. Software Maintenance metrics, focuses on fixing the defects quickly maintaining the quality. These metrics are fix backlog and backlog management index, fix response time and responsiveness, percent delinquent fixes and fix quality.

II. DYNAMIC METRICS

Dynamic metrics helps to evaluate the dynamic behaviour of an application at run-time. The results of dynamic measures are much more accurate than the static measures. The complexity of the dynamic behaviour of real-time applications motivates a shift from static metrics to dynamic metrics. Dynamic metrics are those which perform analysis on the executable or in other words running code. Dynamic metrics includes the reliability modelling along with complexity measures. These metrics depends upon the input given to the system so as to make the system run. In order to know that how many tests have failed, this can only be done dynamically, that is, when the program is running. Dynamic Metrics are derivative from a study of code while it is executing. Thus, dynamic metrics can only be calculated on the software as it is executing. For example: extent of class usage, Dynamic Coupling, and Dynamic Lack of Cohesion.

Dynamic metrics are separated according to following attributes:

a. Cohesion: It refers to how closely and strongly the modules relates to each other. It is expressed as high cohesion and low cohesion. The modules that have high cohesion are preferred more over the modules with low cohesion.

b. Coupling: It shows the dependency of one program module on another module. It is expressed in the form of low coupling and high coupling. Low coupling is correlated to high cohesion and vice-versa.

c. Inheritance: Inheritance metrics measure various aspects of inheritance such as depth and breadth in a hierarchy and overriding complexity .The inheritance metrics give us information about the inheritance tree of the system. Inheritance is a key feature of the OO paradigm. This mechanism supports the class hierarchy design and captures the IS-A relationship between a super class and its subclass

d. Polymorphism: Polymorphism means using the similar function additional than once. Polymorphism is a software characteristic that required to be measured dynamically. This metric was examined mainly in the context of software reusability next to with the determining total quality of the system. Various metric are used for measuring polymorphism, which provides an objective and precise mechanism to detect and quantify the dynamic polymorphism.

III. NEED FOR RELIABILITY AND TESTABILITY

Reliability and Testability are both required in Dynamic Metric. When Testability and Reliability are added in Dynamic Metric, these help in calculating the overall performance of system. Testability Allows us to calculate how easily software could be tested .Testing is done in isolation. Testing of code is performed by dividing the code into modules. Each module is tested and if no error is found then integrated into other error free module. After that overall system is tested. There is the probability of less errors when testability is performed in system.Testabily also increase maintainability of product as testable products has less maintenance cost. Reliability means how long a system can run without making a failure operation. Mean time between failures helps in calculated reliability of system. When any stem is updated its reliability may affected as new changes may or may not compatible with existing system. So after every update, system is verified and its defect are removed to make it extra realiable.After the amendment system is tested for finding errors occurred by adding new features in system. So Reliability is a never ending process as system becomes obsolete after some time and it needs updating to compete them with latest technology.

IV. GENETIC ALGORITHM

A Genetic Algorithm (GA) is a programming technique that mimics biological evolution as a
Problem-solving strategy. The process usually begins with randomly generated population of chromosomes, which represent all possible solution of a problem that are considered candidate solutions

Working steps of Genetic Algorithm are:

1. [START] Generate random population of n chromosomes i.e. suitable for the problem.
2. [FITNESS] Evaluate the fitness f(x) of each chromosome x in the population.
3. [NEW POPULATION] Create a new population by repeating following steps until the new population is complete.

a) [SELECTION]: Reproduction (or selection) is an operator that makes more copies of better strings in a new population. Reproduction is usually the first operator applied on a population.

b) [CROSSOVER]: A crossover operator is used to re-combine two strings/parents to get better new two strings/children. It is important to note that no new strings are formed in the reproduction phase.

c) [MUTATION]: Mutation adds new information in a random way to the genetic search process. It is an operator that introduces diversity in the population whenever the population tends to become homogeneous due to repeated use of reproduction and crossover operators.

d) [ACCEPTING] place new offspring in the new population.

4. [REPLACE] use new generated population for the further run of the algorithm.
5. [TEST] if the end condition is satisfied then stops and re-turns the best solution in current population.
6. [LOOP] Go to step 2.

Genetic algorithm has been used in many of the research works related to optimization of the test cases. The Genetic algorithm has been proposed to use in software engineering because it solves the optimization problems. The results provided are good enough and are optimal. Genetic algorithm is also known as heuristic search algorithm.

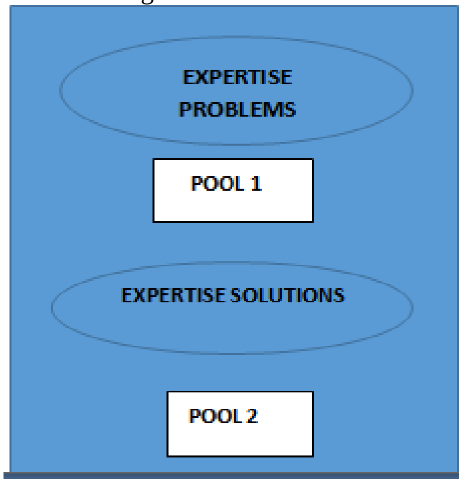


Figure 1

In the figure, here there are two pools. In the first pool there are problems that are defined by the experts whereas in the second pool the solutions are shown that are given by the expertise. In our methodology Genetic algorithm gives the best possible solution for the problems that are occurred when new features are added in the existing system.

V. RESULTS

In the following results we are comparing two dynamic metrices .One is using basic components and second is using Genetic Algorithm. The platform used to evaluate the both Dynamic Metrices is MATLAB. The name MATLAB stands for matrix laboratory. MATLAB is a numerical computing environment and fourth generation programming language. MATLAB is a high performance language for technical computing. It integrates computation, visualization, and programmings in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. The results of the metrices under the following parameters are implemented:

1. Performance: Performance is characterized by the amount of useful work accomplished by a system compared to the time and resources used.

2. Installsability: It is the capability of software that how easily the software can be made understood to a layman about its functions/purpose.

3. Maintainability: Propensity to facilitate updates to satisfy new requirements. Thus the software product that is maintainable should be well-documented, should not be complex, and should have spare capacity for memory, storage and processor utilization and other resources.

4. Documentation: Documentation is the information that describes the product to its users. It consists of the product technical manuals and online information

5. Availability: Availability is expressed in qualitative terms, indicating the extent to which a system can continue to work when a significant component or set of components goes down.

1. Along Testability

Comparison via Graph for both metrices

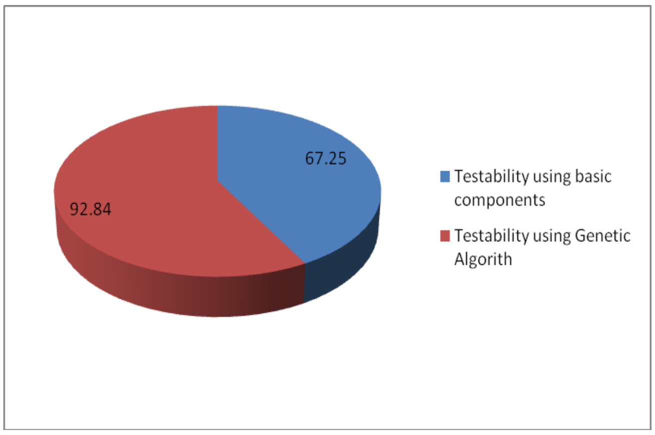
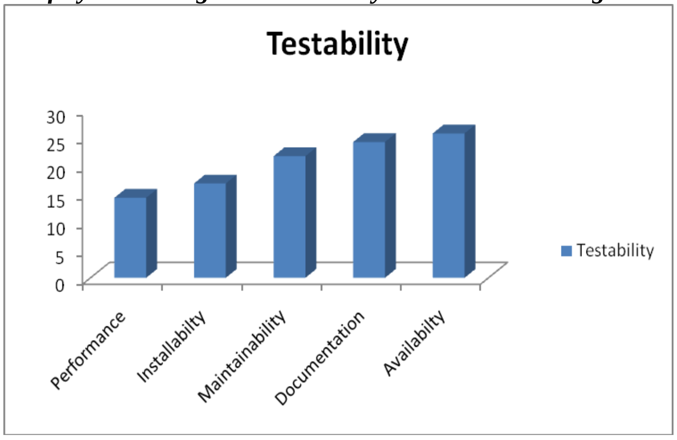
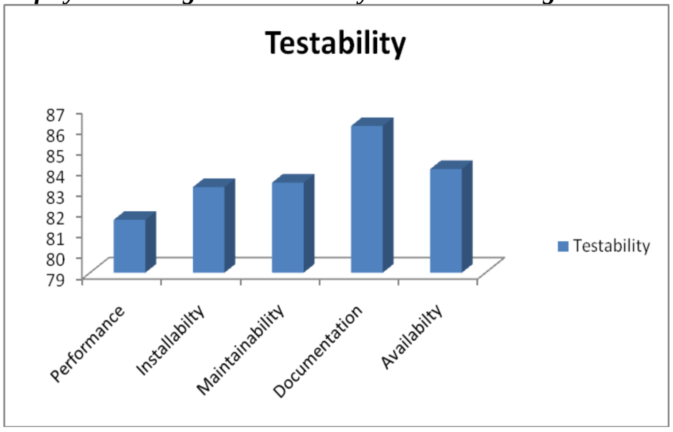


Figure showing quality increase when genetic algorithm is used in dynamic metric

A. Quality of Service analysis of System along with Testability without Genetic algorithm



B. Quality of Service analysis of System along with Testability with Genetic algorithm



2. Along Reliability
Comparison via graph for both metrices

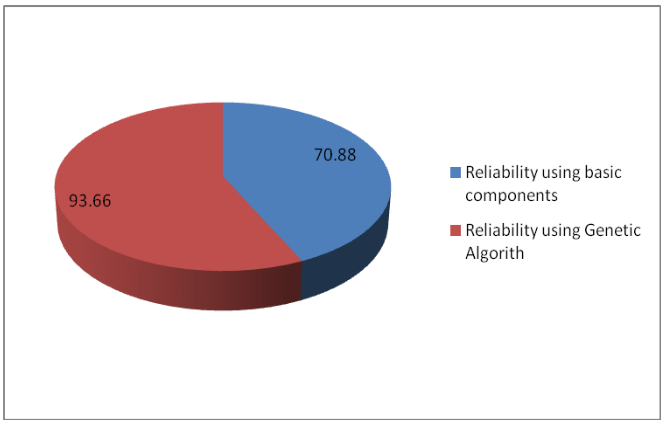
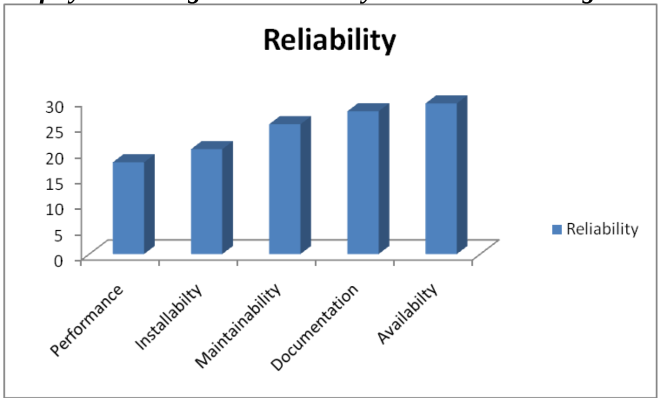
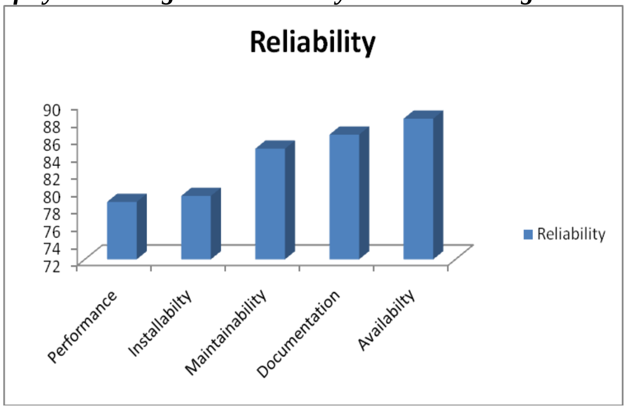


Figure showing quality increase when genetic algorithm is used

A. Quality of Service analysis of System along with Reliability without Genetic algorithm



B. Quality of Service analysis of System along with Reliability with Genetic algorithm



V. CONCLUSION AND FUTURE SCOPE

In this paper we have evaluated the quality of service of both the dynamic matrices. After the detailed analysis of our results we conclude that dynamic metric using genetic algorithm gives better quality analysis as compared to dynamic metric using basic components in terms of performance, maintainability, install ability, documentation and availability. Quality of system is enhanced when Reliability and Testability is included using genetic algorithm in a dynamic metric. In future scope we intended to improve the performance of system using dynamic metric through other optimization methods.

ACKNOWLEDGMENT

I would like to thank all the staff members of the RIMT, Mandi Gobindgarh, Punjab, India.

REFERENCES

[1] Gregg Rothermel, Roland H.Untch, ChengyunChu,Mary Jean Harrold,(2001) “Prioritizing Test Cases for Regression Testing", CSE Journal Articles, Paper 9
[2] Y. Hassoun, R. Johnson, S. Counsell, (2004) “A Dynamic runtime coupling metric for meta-level architectures”, Software Maintenance and Reengineering, pp. 339-346.
[3] Aine Mitchell, James F.Power, (2004) “Run-Time Cohesion Metrics: An Empirical Investigation [4] Yua Jiang et.al, (2008) “Comparing design and code metrics for software quality prediction”, 4th International workshop on Predictor models in software engineering, pp. 11-18.
[5] ParvinderS.Sandhu, Satish Kumar Dhiman, Anmol Goyal, (2009) “A Genetic Algorithm based Classification Approach for finding Fault Prone Classes”, World Academy of Science Engineering and Technology.
[6] PayalKhurana, Puneet Jai Kaur, (2009) “Dynamic Metrics at Design Level”, International Journal of Information Technology and Knowledge Management, Volume 2, No. 2, pp. 449-454.
[7] Er.Iqbaldeep Kaur, Dr. P.K.Suri, Er.AmitVerma, (2010) “Characterization and Architecture of Component Based Models”, International Journal ofAdvanced Computer Science and Applications, Volume 1, No.6.
[8] Varun Gupta, Jitender Kumar Chhabra, (2011) “Dynamic Cohesion Measures for Object-Oriented Software”, Journal of Systems Architecture, pp. 452–462.
[9] Deepak Arora, Pooja Khanna, AlpikaTripathi, Shipra Sharma, Sanchika Shukla (2011) “Software Quality Estimation through Object Oriented Design Metrics”, International Journal 100 of Computer Science and Network Security, Volume 11, No.4.
[10] Dr. Gurdev Singh, ManikSharam, (2011) “Analysis of Static and Dynamic Metrics for Productivity and Time Complexity”, International Journal of Computer Applications, Volume 30, No.1.
[11] Amjed Tahir, Stephen G.Mcdonell, (2012) “A Systematic Mapping Study on Dynamic Metrics and Software Quality”, IEEE International Conference on Software Maintenance.
[12] Marshima M. Rosli, Noor Hasimah Ibrahim Teo, Nor Shahida M. Yusop and N. Shahriman Mohamad” Fault Prediction Model for Web Application Using Genetic Algorithm”
[13] Mehmet Kaya, James W. Fawcett, (2012) “A New Cohesion Metric and Restructuring Technique for Object Oriented Paradigm”,IEEE 36th International Conference on Computer Software and Applications Workshops.
[14] Mitsuhiro Nakamura, Tomoki Hamagami, (2012)“A Software Quality Evaluation Method Using The Change Of Source Code Metrics”, IEEE 23rd International Symposium on Software Reliability Engineering Workshops.
[15] P.B. Nirpal, K.V. Kale, (2012) “Genetic Algorithm Based Software Testing Specifically Structural Testing For Software Reliability Enhancement”, International Journal of Computational Intelligence Techniques, Volume 3, Issue 1, pp. 60-64.
[16] C.R. Kothari, Research Methodology: Methods and Techniques, New Age International Publishers, Rajasthan [17] Antonia Bertolino, (2013) “An Orchestrated Survey on Automated Software Test Case Generation”.
[17] Chayanika Sharma, SangeetaSabharwal, RituSibal, (2013) “A Survey on Software Testing Techniques using Genetic Algorithm”, International Journal of Computer Science Issues, Volume 10, Issue 1.