PERFORMANCE STUDY OF IMPROVING SOFTWARE METRICS ON SOFTWARE DEFECTS PREDICTION

***T.ARUN NEHRU, ** DR.V.SANGEETHA**

**Assistant professor, Department of Computer Science,*
Periyar University College of Arts and Science,Pappireddipatti,
Tamilnadu, India.
*** Assistant professor and Head, Department of Computer Science,*
Periyar University College of Arts and Science,Pappireddipatti,
Tamilnadu, India.

ABSTRACT

Software maintainability is the most significant methods that play an essential part during the quality of software metrics. The software metrics is designed to differentiate the sizes of selected feature subsets and estimate their efficiency in the framework of software defect prediction. Software quality is the essential method to describe the performance of software products attributes without any defects. Software defect prediction is the more efficient approach for predicting the possible software defects from the test data. There are several methodologies considered for improving the open source software code in software metrics. Open source software code testing and software reliability are developed to generate the larger portion of population working under direct or indirect influence of software. Though, the failure of software reliability does not take place during the specified period of time under specified constraints. The study helps to predict the software reliability faults for particular period of time under specified constraints. This in turn improves the software metrics accuracy that helps to reduce the defect involved in software metrics.

Key words: Software Metrics, Software Quality, Software Reliability, Software Maintainability, Open Source Software Code.

1. INTRODUCTION

Software metrics is one of the essential parts during the process of software development. Software development is subjected to cost and also the software testing is presumed as a time consuming process, where the cost and time spent on software development is increasing frequently. Software Requirement is the most important stage in software development as user requirements for collecting the large project in most significant way. Software specification is defined as the method of predicting the software reliability that facilitates to achieve the specified organizational objective with lesser faults.

The main goal of software metric is a typical measure of software system or process that is essential to obtain the quality of attributes. A Software defect prediction scheme contains the software metrics in which they are very useful to accumulate and calculate during the improvement of software system life cycle and dependent variable. Several methods are designed for improving the open source software code within software metrics. Therefore, the software metrics is very efficient for identifying the defects, predicting the imperfect code and risk of project. Also, it provides the accurate software products to the end users at the time of essential validation phase are collected during elicitation is necessary.

Software maintainability is employed to initiate the new features for identifying the reliability and enhance the precision by using software metrics value. Lower the sequence of software maintenance, higher the software's quality level is addressed. The software reliability is essential to improve the better part of the population working under the influence of software. Furthermore, the software metrics has various methods to plan and develop the open source software code for designing the quality and reliability of software product due to the applications of software in different areas.

This paper is organized as follows: Section II discusses defects prediction on software metrics. Section III describes the existing defects prediction on software metrics technique, Section IV identifies the possible comparison between them, Section V explains the limitations as well as the related work and Section VI concludes the paper, research work is given as to predict the software reliability for enhancing the exact precision and guarantee the optimized selection of software metrics in an efficient manner.

2. LITERATURE TABLE

Software Development Life Cycle (SDLC) in [3] planned huge amount of money and effort to spend for attaining the software quality effectively. Software defect prediction (SDP) in [14] evaluates the performance of learning-to-rank technique to build the defect prediction models for ranking the task accurately. However, the several metric selection approaches are reduced to examine the efficiency of metrics during a SDP for ranking task. Software defects prediction by using basic model test in [10], but they reduced to evaluate the number of defects that provides efficient method for calculating the faults. Software reliability modeling in [8] designed to employ max-margin values for observing the function mapping software metrics to identify the counts of bugs. However, these model needs to execute the inference for several variables that is significantly achieved less scalable and more computational problem.

Object-Oriented Software Maintainability (OOSM) in [7] developed for measuring the large number of metrices and identifies the obstacles. Improvement of defect prediction model in [11] planned for maintaining the ascertaining software quality attributes and mainly focused on

constraint resources. However, evaluate the results on software, metaheuristic techniques and hybridized algorithms remains unaddressed. Open-source software projects in [1] described to estimate and compare these metrics to other code and process metrics. Though, even the accuracy is not validated. Software Metric Fluctuation (SMF) in [5] allows guidance to practitioners for choosing appropriate combination of metric and aggregation function. Defect prediction system in [9] developed for focusing the high data causality among the source code metrics and the occurrence of defects. But, these approaches do not employ to predict the amount of faults of a class involved in every time frame.

Package-modularization metrics in [16] predict the defects efficiently while employing the traditional metrics each other. However, the replicating high software systems improve the other programming languages and closed source software systems. Association mining based method in [12] designed to employ the defect prediction strategy to learn *D* modules during unnecessary datasets that enhances the prediction of *D* modules. Software metrics and fuzzy logic for software reliability prediction considered in [2] ensuring better reliability and overall reduction of future maintenance costs. Though, the better reliability leads to improve the maintenance costs. Multiple kernels Ensemble Learning (MKEL) in [13] develop the attributes of data and improve the performance that provides better results for SDP task. Doubly stochastic Poisson technique in [15] introduced very easier and efficient update for its complex conditional distributions. Though, the lack of simple and efficient algorithms for following computation has limited its application to predict the software defect.

Threshold derivation model in [4] establish the generalized thresholds that are effectively useful in various software systems. Though, the derived thresholds are not tested and evaluate in wider datasets including software systems. Network-based approach in [18] widely examines the change distributions and correlation among the centrality measures and range of change propagation. Though, frequent subgraph mining tools cannot determine to change propagation patterns in software change propagation network that decrease the software maintenance efforts. Metrics threshold values in [6] employed to against the bad smell using risk analysis at five different levels. However, the bad smell has no ability to support the testing team for identifying faults. Software Testing Defect Corrective Model (STDCM) in [17] evaluates the amount of remaining defects in software product that significantly increase the value of software projects. However, the STDCM technique is more complicated and therefore improves the failure rate.

In this paper, in order to overcome the above mentioned limitation, a defects prediction on software metrics is designed. That aims to guarantee the optimal selection of software metrics for detecting the faults and extensively decreasing the software metrics selection time. Hence, the defects prediction is essential to improve the accuracy for achieving the high performance by using software metrics.

3. DEFECTS PREDICTION ON SOFTWARE METRICS

Software metrics is essential to detect the software defects prediction for improving the software quality. Software quality is the most important to measure the performance of software products with lacking of identifying the faults. Also, the software quality metrics are a subset of software metrics that are mainly focus on the quality features of product, process, and project. Product metrics is useful to explain the aspects of product namely size, complexity, design features and quality level. The process of software metrics are employed to achieve the software size that provides better software performance optimization, software debugging and estimation costs. Thus the process of software defect prediction is employed to identify the defects and provides efficient removal and offering the value of software system based on various software metrics. The performance of detects prediction on software metrics system is compared against with the three existing methods such as Object-Oriented Software Maintainability (OOSM) technique, Software Development lifecycle (SDLC) technique and Software Metric Fluctuation (SMF) technique.

3.1 Classifying Metrics for Assessing Object-Oriented Software Maintainability: A Family of Metrics Catalogs

Object-Oriented Software Maintainability (OOSM) is mainly designed for measuring the very large quality of software metrics. These catalogs are very essential to provide the framework of OOSM metrics adoption and categorization of OOSM metrics that depends upon the selection of metrics' classification. Due to the lack of useful information about the OOSM metrics for maintaining the decision-making system implemented in OOSM estimation. The main objective of OOSM technique generates the useful information like metrics' categorization during the decision-making method. Based on the variety of OOSM metrics is designed for measuring every maintainability tasks. Furthermore, the approach offers useful information about the OOSM metrics for developers, project managers and researchers.

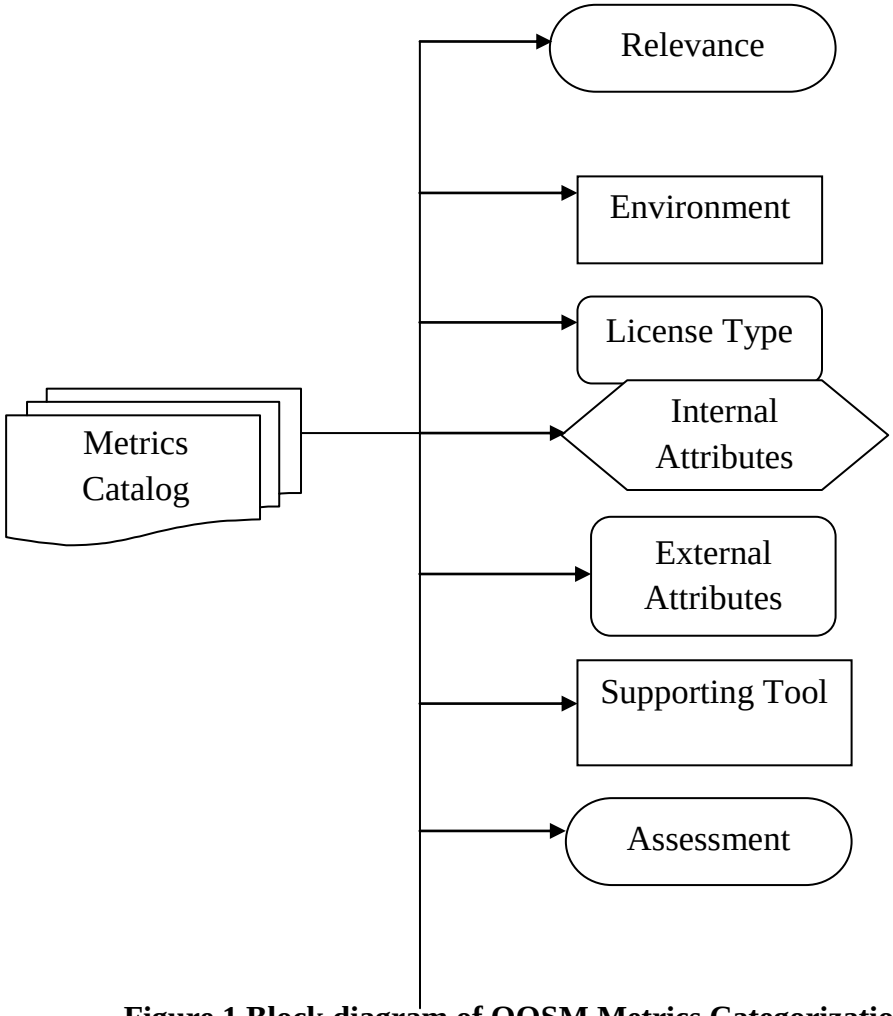

Figure 1 Block diagram of OOSM Metrics Categorization

Figure 1 describes the block diagram of OOSM Metrics Categorization. The metrics catalogs are classified into seven types such as relevance, environment, license type, internal attributes, external attributes, supporting tool and assessment. Initially, the process of relevance includes two varieties of metrics classification like most adopted and relevant metrics. Then the environment has the ability to employ both the academic and industrial process for achieving better results. Academic metrics consists of OOSM metrics that includes software engineering academic research and industrial metrics are implemented in industrial scenarios. License type categorizes the metrics in two set namely, the open source metrics are employed to evaluate open source software maintainability and proprietary metrics are utilized in estimation of proprietary software.

A characteristic of internal attributes consists of five software maintainability such as size, complexity, coupling cohesion and software architecture constraints. The process of supporting

tool is essentially to accumulate automatically by using the tool. Then the external attributes is very useful to clarify the Systematic Mapping Study (SMS) approach in an efficient manner. Finally, the assessment category is classified into two types that are evaluated Metrics and validated Metrics. In addition, the performance analysis of OOSM Metrics categorization is efficiently supports the decision-making process that the classification metrics are exactly implemented in their estimation.

3.2 A fuzzy logic based approach for phase-wise software defects prediction using software metrics

The software defect density indicator calculation in terms of Software Development lifecycle (SDLC) is designed for improving the dependable software product. The software defect prediction is not useful at the end of testing phase since the changes required to implement in the SDLC system for attaining software quality. The fuzzy logic based technique is designed for predicting the software defect prediction to enhance the reliability of appropriate software metrics of all SDLC. Therefore, the predicted defects of different software projects are very close to establish the actual defects identified during testing.

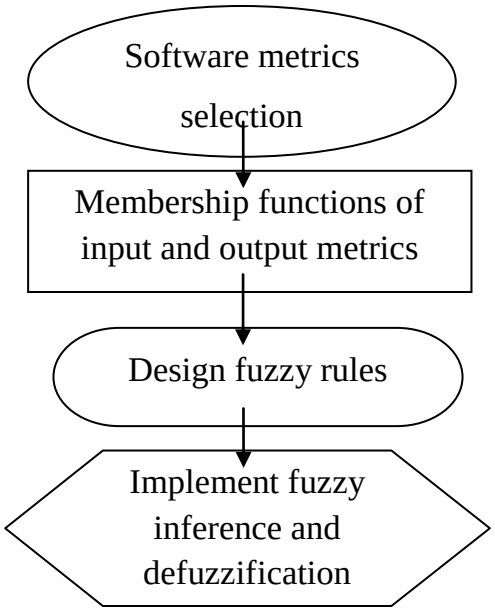

Figure 2 Process of SDLC Model

Figure 2 explains the process involved during of SDLC technique. The defect density indicator in the first four phases of SDLC is to evaluate based on the performance of first four phases of SDLC method. Hence, the model supports the reliability to classify four types such as requirements analysis, design, coding and testing phases of SDLC. Initially, Requirement

analysis Phase Defect Density Indicator (RPDDI) provides efficient software metrics that are measured as input in design phase to predict the defect density indicator at the end of design phase. Next, the Design Phase Defect Density Indicator (DPDDI) are intended as input in coding phase to calculate the defect density indicator at the end of coding phase. Finally, the Coding Phase Defect Density Indicator (CPDDI) is evaluated as input in testing phase to predict the defect density indicator at the end of testing phase. At last, the predicted defect density indicators are very essential to predict the reliability defect in various formation of SDLC of a software project during software metrics selection.

3.3 Software Metrics Fluctuation: A Property for Assisting the Metric Selection Process

Software quality attributes are mainly utilized for evaluating the more suitable metrics efficiently. Though, the selection of such metrics is not always apparent and also it is very complicated task with large number of obtainable metrics. Software Metric Fluctuation (SMF) is designed to measure the quantity in which a metric score differ, due to the changes taking place among consecutive system's versions. While the metrics is useful to access with respect to its fluctuation, the SMF approach provides the extensive range involving metrics that are highly constant. Therefore, the metrics estimation system plays an important role for evaluating the fluctuation. The properties of SMF technique are calculated by using two types. Initially, SMF has the capability to illustrate the various metrics. Then, the process of SMF is employed to evaluate the different metrics of selection method.

The fluctuation property has the ability to distinguish several metrics that successfully measures the same quality attribute like cohesion, coupling and convolution. It is mainly based on metric selection approach for choosing the metric either stable or sensitive according to the requirement of specific value of attributes. Furthermore, the various metrics work at the micro-level while they have capacity to working at a different level. The aggregating metrics is converted from the micro-level to the macro-level that is employed to provide the aggregation function in frequent manner. Thus the performance of SMF is an essential metric property that is applied to increase the precision of metrics selection method. For considering the SMF approach that is essential to require various aggregations functions which is depends upon the degree of fluctuation. Furthermore, the SMF solution has the ability to generate input to researchers for selecting the suitable mixture of metric and aggregation function during the system of metric selection.

The experimental evaluation of defects prediction on software metrics technique is conducted in various factors such as software metrics accuracy, fault detection rate and software metrics selection time.

4. COMPARISON OF DEFECTS PREDICTION ON SOFTWARE METRICS USING DIFFERENT TECHNIQUES AND SUGGESTIONS

In order to compare the defects prediction on software metrics method using different techniques, size of software program is taken to perform this experiment. Various parameters are used for defects prediction on software metrics techniques.

4.1 Software Metrics Accuracy

Software metrics accuracy is defined ratio of number of correctly identified defects without any error to the total number of defects. Software metrics accuracy is measured in terms of percentage (%) and is mathematically formulated as below,

$$Software\ Reliability = \frac{Correctly\ identified\ defects}{Total\ number\ of\ defects}$$

When the software metrics accuracy is higher, the method is said to be more efficient.

Table 4.2 Tabulation of Software Metrics Accuracy

Size of Software Program	Software Metrics Accuracy (%)		
	OOSM	**SDLC**	**SMF**
10	68	50	63
20	75	56	68
30	78	60	69
40	81	62	74
50	84	64	78
60	86	68	79
70	90	72	84

Table 4.1 explains the software metrics accuracy versus different size of software program in the range of 10 to 70. The Software metrics accuracy comparison takes place on existing Object-Oriented Software Maintainability (OOSM), Software Development lifecycle (SDLC) and Software Metric Fluctuation (SMF) approach.

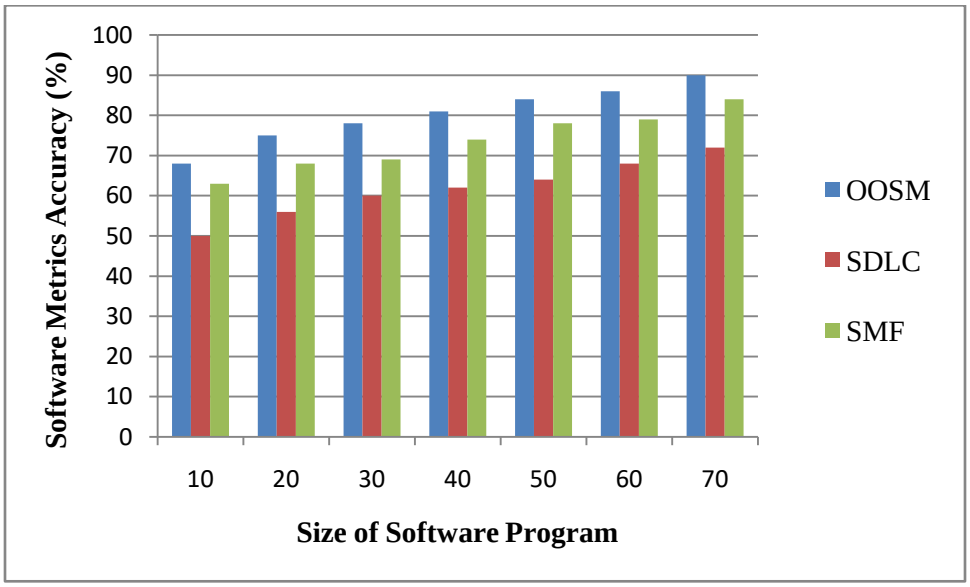

Figure 4.1 Measurement of Software Metrics Accuracy

Figure 4.1 measures the software metrics accuracy of existing techniques. Software metrics accuracy of Object-Oriented Software Maintainability (OOSM) technique is comparatively higher than that of Software Development lifecycle (SDLC) and Software Metric Fluctuation (SMF) method. Research in Object-Oriented Software Maintainability (OOSM) approach consumes 8% higher accuracy than Software Metric Fluctuation (SMF) technique and 23% higher accuracy than Software Development lifecycle (SDLC) technique.

4.2 Fault Detection Rate

Fault detection rate is defined as the measure of testing effectively. It is calculated as a ratio of number of defects found by the testing and total number of defects including found afterwards. Fault detection rate is measured in terms of percentage (%) and is mathematically formulated as below,

$$Fault\ Detection\ Rate = \frac{Number\ \ of\ defects\ found\ by\ testing}{Total\ no\ of\ defects}$$

When the fault detection rate is higher, the method is said to be more efficient

Table 4.2 Tabulation of Fault Detection Rate

Size of Software program	Fault Detection Rate (%)		
	OOSM	SDLC	SMF
10	57	65	75
20	59	69	78
30	60	70	80
40	65	72	82
50	68	75	84
60	70	78	86
70	75	82	90

Table 4.2 explains the fault detection rate versus different size of software program in the range of 10 to 70. The fault detection rate comparison takes place on existing Object-Oriented Software Maintainability (OOSM), Software Development lifecycle (SDLC) and Software Metric Fluctuation (SMF) approach.

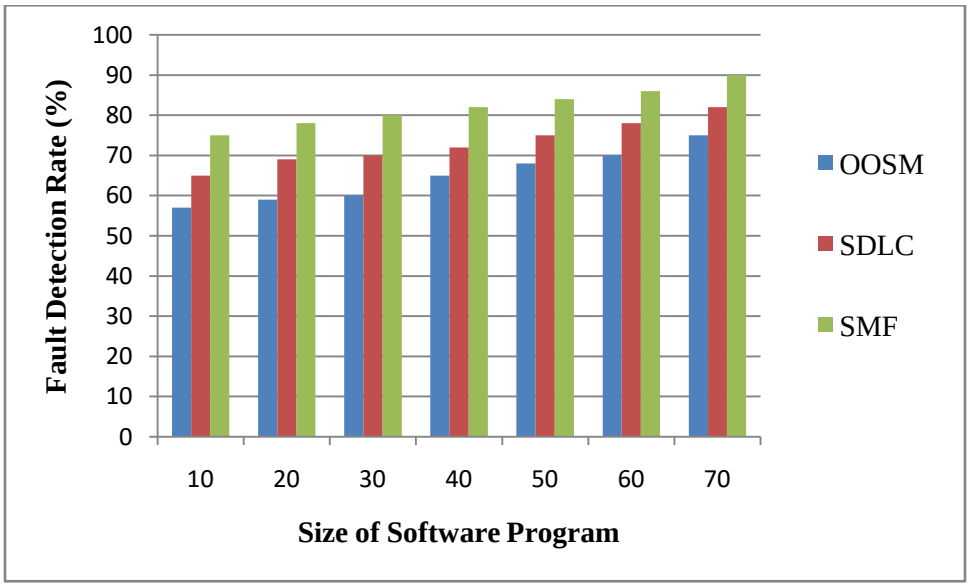

Figure 4.2 Measurement of Fault detection rate

Figure 4.2 measures the fault detection rate of existing techniques. Fault detection rate of Software Metric Fluctuation (SMF) technique is comparatively higher than that of Object-Oriented Software Maintainability (OOSM) and Software Development lifecycle (SDLC) method. Research in Software Metric Fluctuation (SMF) approach consumes 11% higher fault

detection rate than Software Development lifecycle (SDLC) technique and 21% higher rate of detecting faults than Object-Oriented Software Maintainability (OOSM) technique.

4.3 Software Metrics Selection Time

Software metrics selection time is defined as the difference between the ending time and starting time required for software metrics selection. Software metrics selection time is measured in terms of milliseconds (ms) and is mathematically formulated as below,

$$Fault\ Detection\ Time\ (ms) = Ending\ time - Starting\ time\ for\ software\ metrics\ selection$$

When the software metrics selection time is lower, the method is said to be more efficient

Table 4.3 Tabulation of Software Metrics Selection Time

Size of Software Program	Software Metrics Selection Time (ms)		
	OOSM	SDLC	SMF
10	38	45	30
20	40	47	33
30	43	50	35
40	44	53	38
50	46	56	40
60	49	58	41
70	51	60	44

Table 4.3 explains the software metrics selection time versus different size of software program in the range of 10 to 70. The software metrics selection time comparison takes place on existing Object-Oriented Software Maintainability (OOSM), Software Development lifecycle (SDLC) and Software Metric Fluctuation (SMF) approach.

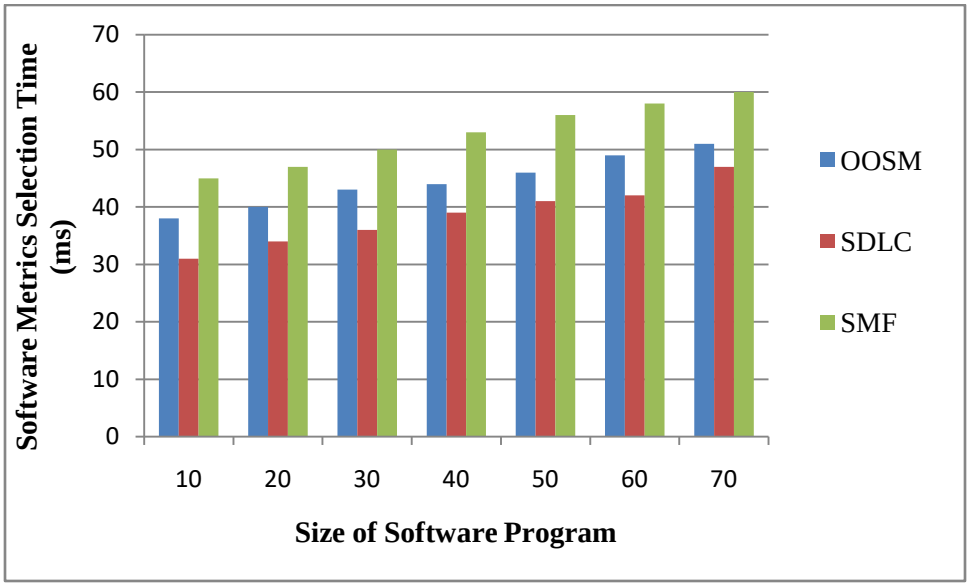

Figure 4.3 Measurement of Software Metrics Selection Time

Figure 4.3 measures the software metrics selection time of existing techniques. Software metrics selection time of Software Development lifecycle (SDLC) technique is comparatively lesser than that of Object-Oriented Software Maintainability (OOSM) and Software Metric Fluctuation (SMF) method. Research in Software Development lifecycle (SDLC) approach consumes 16% lesser selection time than Object-Oriented Software Maintainability (OOSM) technique and 37% lesser selection time than Software Metric Fluctuation (SMF) technique.

5. DISCUSSION AND LIMITATION OF DEFECTS PREDICTION ON SOFTWARE METRICS USING DIFFERENT TECHNIQUES

The characteristic of metrics categorization does not take place during the process of Object-Oriented Software Maintainability (OOSM) technique. The decision making system about the software metrices are difficult to accept in OOSM strategy. For relating the threshold value by using design principles are very difficult to predict the defects in the OOSM method.

Software Metric Fluctuation (SMF) is minimizing the usability of considering metric fluctuation while providing the measurement plans through industrial framework are evaluated. The process of SMF requires more time for executing the software metrics approach**.** Also, the lesser accuracy related to software metric selection is offered and it cannot collect to achieve both the sensitive and more stable metrics.

Predicting software defects is essential to address the higher computational difficulty and processing cost by using Software Development lifecycle (SDLC) system. The uncertainties associated over the assessment of software size metric are not concentrated. Then the smaller important software metrics are occurred to improve the time complication with lesser flexibility level.

5.1 Future Direction

The future direction of defects prediction on software metrics can be employed to enhance the precision and assure the optimized selection for providing better software system quality. In addition, the software metrics can be carried out to predict the deficiency of software reliability for decreasing the faults efficiently.

CONCLUSION

The comparison of different techniques for defects prediction on software metrics technique is carried out. SDLC technique is employed to design more computational complexity and evaluation cost for predicting the software defects. An OOSM method has the lesser ability to choose the optimized software metrics. Thus the process of threshold value is very hard task to predict the fault in the successive manner by using OOSM scheme. SDLC method provides low significant software metrics that maximize the time consuming with lesser flexible are addressed. The software metric selection is very hard task to measure more sensitive and constant metrics accurately. Finally, from the result, the research work can predict the reliability defect of software system and ensure the optimal selection of software metrics. Furthermore, the technique is achieved to improve the software metrics accuracy, rate of fault detection and minimizing the software metrics selection time with better efficient.

REFERENCE

[1] Elvira Maria Arvanitou, Apostolos Ampatzoglou, Alexander Chatzigeorgiou and Paris Avgeriou, “Software Metrics Fluctuation: A Property for Assisting the Metric Selection Process”, Information and Software Technology, Year: April 2016, Volume: 72, Pages: 110-124

[2] Harikesh Bahadur Yadav and Dilip Kumar Yadav, “A fuzzy logic based approach for phase-wise software defects prediction using software metrics”, Information and Software Technology, Year: July 2015, Volume: 63, Pages: 44-57

[3] Juliana de A.G. Saraiva, Micael S. de Francüa, Sergio C.B. Soares, Fernando J.C.L. Filho and Renata M.C.R. de Souza, “Classifying Metrics for Assessing Object-Oriented Software Maintainability: A Family of Metrics Catalogs”, The Journal of Systems and Software, Year: May 2015, Volume: 103, Pages: 85-101

[4] Satwinder Singh and K. S. Kahlon, "Object oriented software metrics threshold values at quantitative acceptable risk level", CSI Transactions on ICT, Year: Nov 2014, Volume: 2, Issue: 3, Pages: 191-205

[5] Omer Faruk Arar and Kursüat Ayan, "Deriving Thresholds of Software Metrics to Predict Faults on Open Source Software: Replicated Case Studies", Expert Systems with Applications, Year: Nov 2016, Volume: 61, Pages: 106-121

[6] S.W.A. Rizvi, V.K. Singh and R.A. Khan, "The State of the Art in Software Reliability Prediction: Software Metrics and Fuzzy Logic Perspective", Information Systems Designs and Intelligent Applications, Year: Feb 2016, Volume: 433, Pages: 629-637

[7] Matthieu Foucault, Cédric Teyton, David Lo, Xavier Blanc and Jean-Rémy Falleri, "On the usefulness of ownership metrics in open-source software projects", ELSEVIER: Information and Software Technology, Year: Aug 2015, Volume: 64, Pages: 102-112

[8] Sotirios P. Chatzis and Andreas S. Andreou, "Maximum Entropy Discrimination Poisson Regression for Software Reliability Modeling", IEEE Transactions on Neural Networks and Learning Systems, Year: Nov 2015, Volume: 26, Issue: 11, Pages: 2689-2701.

[9] Cesar Coutoa, Pedro Piresa, Marco Tulio Valentea, Roberto S. Bigonhaa and Nicolas Anquetilc, "Predicting software defects with causality tests", ELSEVIER: The Journal of Systems and Software, Year: 2014, Volume: 93, Pages: 24-41.

[10] Peng He, Bing Li, Xiao Liu, Jun Chen, Yutao Ma, "An empirical study on software defect prediction with a simplified metric set", ELSEVIER: Information and Software Technology, Year: 2015, Volume: 59, Pages: 170-190.

[11] Ruchika Malhotra, "An empirical framework for defect prediction using machine learning techniques with Android software", ELSEVIER: Applied Soft Computing, Year: May 2016.

[12] Zeeshan Ali Rana, M. Mian Awais and Shafay Shamail, "Improving Recall of software defect prediction models using association mining", Knowledge-Based Systems, Year: Dec 2015, Volume: 90, Pages: 1-13.

[13] TiejianWang, Zhiwu Zhang, Xiaoyuan Jing and Liqiang Zhang, "Multiple kernel ensemble learning for software defect prediction", Automated Software Engineering, Year: Dec 2016, Volume: 23, Issue: 4, Pages: 569-590.

[14] Xiaoxing Yang, Ke Tang and Xin Yao, "A Learning-to-Rank Approach to Software Defect Prediction", IEEE Transactions on Reliability, Year: March 2015, Volume: 64, Issue: 1, Pages: 234-246.

[15] Andreas S. Andreou and Sotirios P. Chatzis, "Software defect prediction using doubly stochastic Poisson processes driven by stochastic belief networks", ELSEVIER: The Journal of Systems and Software, Year: 2016, Volume: 122, Pages: 72-82.

[16] Yangyang Zhao, Yibiao Yang, Hongmin Lu, Yuming Zhou, Qinbao Song and Baowen Xu, "An empirical analysis of package-modularization metrics: Implications for software fault-proneness", ELSEVIER: Information and Software Technology, Year: 2015, Volume: 57, Pages: 186-203.

[17] K. Karnavel and R. Dillibabu, "Development and Application of New Quality Model for Software Projects", The Scientific World Journal, Year: Oct 2014, Volume: 2014.

[18] Rongcun Wang, Rubing Huang and Binbin Qu, "Network-Based Analysis of Software Change Propagation", The Scientific World Journal, Year: March 2014, Volume: 2014.