ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (1120-1124), Month: April - June 2015, Available at: www.researchpublish.com

A Bird's Eye View of Software Quality Metrics

[1]Anusha K, [2]Ankita Sri

Abstract: **Quality is a very important aspect for the software product. Good quality product provides customer satisfaction and provides all benefits to organization and helps the organization to meet their goals. Metrics are used by the software industry to quantify the development of software. Quality increases productivity, which brought software metrics to the forefront. There are various types of software quality metrics to ensure customer satisfaction. The process of applying software metrics to a software process and to a software product is a complex task which requires discipline and knowledge of the status of the process and product of software in regards to the goals to achieve.**

There are specific metrics for different stages of the software development cycle. When a company uses the best software metric during each development phase, the quality of the software will dramatically increase. Therefore, it is highly recommended to use software metrics during all stages of the development process. There are various software quality models available like the quality models as defined by McCall, Richards and Walters in 1977 and the later model of Boëhm published in 1978. Models can be chosen according to the size and the structure of project, type of project or which best suits the organization. Various software quality metrics and the various models present which will be discussed in this paper.

Keywords: **Software Quality, Reliability, Software Metrics, Function point, Object Oriented.**

1. INTRODUCTION

In this era quality is the most important factor in any kind of business. To achieve a respectable position in global market in IT industry [1], a company must have to produce very high quality products. Competition is very high and one cannot afford correcting errors after shipping the products to the customer. A correction after shipping is very costly and it affects the Company credibility and organizations cannot afford losing Customers due to these kinds of problems. To avoid these problems, organizations should follow a proper quality management pla1n to remove errors from the products [2].

Software quality metrics focuses on the quality aspects of process, product and project. The project parameters such as the developers ,knowlege levels, the schedule, the size, complexity and the organization structure certainly affects the quality of the product. The essence of software quality engineering is to work on the relationships between various activities and based on the results provide improvement plans. Moreover, we should view quality from the entire software life-cycle.

2. LITERATURE SURVEY

Software Metrics talks about various ways to estimate the quality of software project and their products. In [1] the principle objective is to break down the existing programming measurements and checking how the source code quality is accessed. Some major program complexity metrics are cyclomatic complexity (McCabe's), Halstead metrics. System complexity metrics like Yin and Winchester, [yin, 1978] made made aggregations of two metrics called: essential measurements and auxiliary measurements. The essential metrics are communicated through definite qualities of the determination of configuration. The measurements are dependent upon two configuration properties: coupling and effortlessness. The auxiliary measurements can give an evidence about the principle framework module or database table. The auxiliary measurements as: fan-in and fan-out. Henry and kafura are used to calculate the system complexity. Metrics Based on the Concepts of Object Oriented are also required as with the growing of the OO technology usage it was necessary to measure coupling, cohesion and all important aspects of the OO technology. The first suites of OO design metrics were proposed by Chidamber and Kemerer [Chidamber, 1994], who proposed 6 class-based design metrics for OO system (CK Metrics).

In [2] there are various metrics described based on the perspective like commercial perspective can have technical metrics, defect metrics, warranty metrics , end user satisfaction metrics etc. On the basis of significance core metric and non-core metric. On the basis of observation primitive and computed metric.On the basis of measurement direct and indirect measurement. Some of the formal approaches of software measurement are Axiomatic Approaches of Software Measurement, Functional Approach of Software Measurement, Halstead's software science, Complexity Metrics like McCabe's , Reliability Metrics: MTBF= MTTF + MTTR where mean time between failures(MTBF).The sum of mean time to failure (MTTF) and mean time to repair (MTTR) gives the measure MTBF. Availability = MTTP / (MTTF +MTTE)* 100% , Readability Metrics etc.

In [3] we understood how the software metrics affects the software quality. A good metric should be simple, precisely definable, objective, easily obtainable, valid and robust. It classifies the software metrics based on process metrics, project metrics and product metrics. With the rapid advancement in software industries, software metrics have also been developed fast.

In [4], many issues related to quality along with responsibilities of management are identified. Management plays an important role in the SQA [11]. So it is the prime responsibility of the team managers to facilitate the team members and provide them the good working environment. Knowledge can be improved through formal training courses and attending seminars by different experts. Some models related to Software Quality assurance are also discussed like PARNAS MODEL, DAVID MODEL, PEER REVIEWS MODEL.

3. IMPORTANCE OF SOFTWARE QUALITY

In recent times the importance of software quality has come to light when random errors on everyday objects like washing machines or on a bank statement were randomly attributed to a bug in the "computer code" or using the ignorant adage of "the computer does things" without making an effort to undermine the cause of the problem or even separating it by hardware or software, which can lead to cataclysmic chain of events. Bearing all this in mind, the importance of enforcing software quality in computer practices has become highly important. Seeing the penetration of computer code into various activities of life, thus the highest levels of software quality is paramount.

How do we assess the quality of something intangible like software quality? The answer varies according to the situation. For example, a small word processing error in a student's assignment will not be a huge issue. But if there is a small error in space shuttle guidance, the mission could be a failure endangering human lives. The definition will differ according to different factors like quality of products and business.

It is crucial to set the goals as well as proactive monitoring of quality factors to make sure that the goals are resolved completed within the given time frame.

Software quality depends on a number of factors. Quality as defined by David and Garwin, is a complex as well as multifaceted concept, which can be viewed according to different points of view as follows

1) User View:

The user viewpoint of software quality is highly subjective and depends on the user. This view evaluates the software product against the user's expectations and also based on various quality attributes like reliability, performance and operational use.

2) Manufacturing View:

This viewpoint looks at the production aspect of the software product. It basically works on delivering a product free of defects instead of making a defective product and spending quality time on repairing or ironing out the defects at a much later stage. Being process based, this viewpoint focuses on conformity to the process, which will eventually lead to a better product.

Quality Models such as ISO 9001 as well as the Capability Maturity Model do encompass this viewpoint that stress on following the process as opposed to going by specification. Both the models CMM as well as the ISO, indirectly do imply by following the principle of "Documenting what you do and doing what you say" helps in improving the product quality.

The critic's viewpoint is that following an optimized and high quality product manufacturing method can also lead to the standardization of a product making it more of a commodity rather than a standout product.

3) Product View:

The product viewpoint looks at the internal features as well as the characteristics of the product. The idea behind this viewpoint is that in case a product is sound in terms of the features and functionality it offers, and then it will also be favourable when viewed from a user viewpoint in terms of software quality. Controlling the internal product quality indicators will influence positively the external product behaviour (user quality) . There are models trying to link both the views of software quality but more work is needed is this area.

4) Value based view:

The value-based view becomes important when there are lots of contrasting views by different departments in an organization. For example, the marketing department generally take a user view and the technical department will generally take a product-based view. The contrasting viewpoints help to develop a satisfactory product with the different viewpoints complementing each other but the later stages of the software product development might have issues.

The issues arise when there might be a set of change proposed to a certain view that can end up throwing a conflict in the other view. A value-based view comes into play helping resolve such conflicts where the software product is not delayed for a longer period This view helps to keep the software product on track and within initial cost and timeline estimates.

4. CLASSIFICATION OF METRICS

There are three types of software metrics: process metrics, project metrics and product metrics. [7]

1) Process Metrics:

Process metrics tells about the process of software development.. To improve the process it is needed to specify some attributes which we can measure and then use these metrics to obtain indicators which will help to derive a strategy for process improvement. Process metrics can also be used to help in software development and maintenance. With the help of these metrics software engineers are able to assess the efficiency of the software process performed. Examples include the efficacy of defect removal during development, the patterning of testing defect arrival, and the response time of the fix process.

2) Project Metrics:

Project metrics are used to monitor project situation and status. It enables the project manager to track risks, identify problem areas, adjust workflow, and evaluate the abilities of the project team. The two main purposes of project metrics are, firstly is to minimize the development schedule which will help to reduce any delay in project, reduce the risks and problems. Secondly is to continuously monitor the product quality throughout its development, modify the technical issues whenever required which in turn help to reduce overall cost of software project. These metrics help a lot during estimation.. Examples include the number of software developers, the staffing pattern over the life cycle of the software, cost, schedule, and productivity. [8]

3) Product Metrics:

Product metrics describe the attributes of the software product at any phase of its development. It includes measure like the size of the program, complexity of the software design, performance, portability, maintainability, and product scale. Product metrics are used to improve the quality of the product. Product metrics can be used to measure the medium or the final product. These metrics helps to determine whether the product is meeting user requirements or not. These metrics help the software engineers to detect and correct potential problems before a serious damage may occur. It contains various metrics like metrics for analysis model, metrics for design model, metrics for source code, metrics for testing, metrics for maintenance.

There are broad range of software metrics available. The software metrics depend on what software's attributes we want to quantify or qualify. There are two main different classes of metrics: metrics for the software process and metrics for the software product. Metrics for the software process are related to the effort it takes to complete the project, the resources to spend on a project, and the methodology we follow for the software development. For example: the time needed for project completion, the number of people needed to develop it, the overall cost of the project, and the method we are following. Many types of attributes of software can be measured. The metric which we can select depends on the nature of the software product. For example, for the requirements, we might want to know how many requirements a project has, make sure it lacks of ambiguity and completeness (i.e all the functions needed are covered). For the software product of a

program, the number of lines of code, its complexity, the functionality it covers, the number of potential defects ('bugs') it will have, and the number of test cases needed to verify all requirements have been implemented.

We also could measure the reliability of the software once delivered. The Software Engineering community still has time to determine the correct metrics to help to measure reliability for the software product and process. We can find more efficient ways of improving software project, product and process management.

5. OVERVIEW OF SOFTWARE METRICS

The software industry does not have standard metric and measurement practices. Most of the software metric has multiple definitions and ambiguous rules for counting. There are also important subject issues that do not have specific metrics, such as quantifying the volume or quality levels of databases, web sites and data warehouses. There is a lack of strong empirical data on software costs, schedules, effort, quality, and other tangible elements, which results in metric problems. [12]

A. Source Code Metrics:

"Source lines of code" or SLOC was the first metric developed for quantifying the outcome of a software project. The divergent "lines of code" or LOC has similar meaning and is also widely acceptable. "Lines of code" could be defined either:

- A physical line of code.
- A logical line of code.

Physical lines of code are sets of coded instructions terminated by hitting the enter key of a keyboard. Physical lines of code and logical lines of code are almost identical for some languages, but for some languages there can be considerable differences. Generally, the difference between physical lines of code and logical lines of code is often excluded from the software metrics literature.

B. Function Point Metrics:

The function point analysis to measure software application is enumerated from analysis of the requirements and logical design of the application. Function Point count can be applied to Development projects, Enhancement projects, and existing applications as well. [10] There are five key elements of Function Point Analysis, which capture the functionality of the application. These are:

- External Inputs (EIs),
- External Outputs (EOs)
- External Inquiries (EQs)
- Internal Logical Files (ILFs) and External
- Interface Files (EIFs).

First three elements are of Transactional Function Types and last two are of Data Function Types. Calculate the adjusted Function Point count. Function Point Analysis has proven to be an accurate technique for sizing, documenting and communicating a system's capabilities. It has been successfully used to evaluate the functionality of real-time and embedded code systems, such as robot based warehouses and avionics, as well as traditional data processing. As computing environments become increasingly complex, it is proving to be a valuable tool that accurately reflects the systems we deliver and maintain.

Function points are clearly superior to lines of code metrics. [9]

C. Object-Oriented Metrics:

In today's software development environment, Object-oriented analysis and design concepts are well known. Object-Oriented Analysis and Design of software provide many advantages such as reusability, decomposition of problem into easily understandable object and the aiding of future modifications. Object-oriented software development requires a diverse approach from more traditional functional decomposition and dataflow development methods. But the OOAD software development life cycle is not easier than the typical procedural approach. Therefore, it is necessary to provide

dependable guidelines that one may follow to help ensure good OO programming practices and write reliable code. Object-Oriented programming metrics is an aspect to be considered. Metrics should be a set of standards against which one can measure the effectiveness of Object-Oriented Analysis techniques in the design of a system. [11]

OO metrics are not linked to all other known software metrics. There are no conversion rules between the OO metrics and any other metrics, so it is complicated to perform alongside comparisons between OO projects and conservative projects using the currently available OO metrics.

6. CONCLUSION AND FUTURE WORK

The overall improvement in the progress rate was observed in quality and productivity with the use of software metrics. The importance will increase multifold when organizations start embracing newer approach and technology to monitor and improve to deliver a product that meets the user's expectations. Various metrics were compared and the importance of software quality was emphasized. Future work includes improvement in the existing metrics based on the magnitude of the problem statement. Automated tools are enhanced to reduce the time and effort.

The metrics field needs to mature even though it is appreciated by people. Adequate international standards for extensively used software metrics should be established.

REFERENCES

[1] Biffl, S.; Halling, M. "Investigating the influence of inspector capability factors with four inspection techniques on inspection performance" Software Metrics, 2002. Proceedings. Eighth IEEE Symposium on 07 August 2002

[2] Zheng Yan , Prehofer, C. "Autonomic Trust Management for a Component-Based Software System" Dependable and Secure Computing, IEEE Transactions on Nov.-Dec. 2011

[3] Prachi Chhabra*, Lalit Bansal, " An Empirical Study of Software Metrics and Associated Aspects", International Journal of Emerging Research in Management &Technology ISSN: 2278-9359 (Volume-3, Issue-4)

[4] Ming-Chang Lee1and To Chang2 , " Software Measurement and Software Metrics in Software Quality" , International Journal of Software Engineering and Its Applications Vol. 7, No. 4, July, 2013.

[5] Mrinal Singh Rawat,Arpita Mittal, Sanjay Kumar Dubey3, " Survey on Impact of Software Metrics on Software Quality"

[6] (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 3, No. 1, 2012.

[7] Ali Javed , Muazzam Maqsood,KhurramAshfaqQazi , Khurram Ali Shah , "HOW TO IMPROVE SOFTWARE QUALITY

[8] ASSURANCE IN DEVELOPING COUNTRIES", Advanced Computing: An International Journal (ACIJ), Vol.3, No.2, March 2012

[9] Hribar, L. , Burilovic, A. , Huljenic, D. "Implementation of the Software Quality Ranks method in the legacy product development environment" Telecommunications, 2009. ConTEL 2009. 10th International Conference on 8-10 June 2009

[10] Tu Honglei1, Sun Wei1, Zhang Yanan1, "The Research on Software Metrics and Software Complexity Metrics", International Forum on Computer Science-Technology and Applications, 2009.

[11] http://www.pearsonhighered.com/samplechapter/0201729156.pdf.

[12] Capers Jones, Chief Scientist Emeritus, "Strengths and Weaknesses of Software Metrics", Version 5, March 22, 2006.

[13] Kurmanadham V.V.G.B. Gollapudi, "Function Points or Lines of Code? – An Insight"

[14] "Software Quality Metrics for Object Oriented System Environments", June 1995, National Aeronautics and Space Administration, Goddard Space Flight Center, Greenbelt Maryland.