RESEARCH ARTICLE OPEN ACCESS

Software Quality Management Measured Based Code Assessments

Salman Zakariya [1], Mohammed Belal [2]
Department of Computer Science [1] & [2]
Helwan university, Cairo, Egypt

ABSTRACT
Software quality has been a major concern for those involved in the area of software engineering, and researchers as well as practitioners of the domain have proposed instruments to measure it. In order to produce a good software product, several measures for software quality attributes need to be taken into account. System complexity measurement plays a vital role in controlling and managing software quality because it generally affects the software quality attributes like software reliability, software testability and software maintainability. This term paper is primarily is concerned on software development process that affect software quality management, Software quality metrics, Software measurement process, and quality models. These aspects of software development and measurement process include software reliability measurement, ISO approach applicable to software quality and some aspects related to software testing improvement. In this paper we develop a tool named as (SWMetrics) using Microsoft Visual Studio-C# to compute a metrics of LOC, SLOC and Complexity based the Cyclomatic of quality measurement for many format languages of source of code.

Keywords:- Software Measurements, Software Testing, Quality Assurance, LOC, Cyclomatic Complexity

I. INTRODUCTION

Improving software quality by using program analysis & measurement tools and SQA (Software Quality Assurance) method is at the appropriate points during the process of development [3]. Although using various program analysis tools and techniques are effective for quality management and measurement.

Periodic analysis and quality measurements of software products throughout the life-cycle are very important to manage and improve software quality [3] [4]. Software development departments are faced with some difficulties when using the tools and techniques for analysis and measurement. Moreover Software measurement has become a key aspect of good software engineering practice. Measurement activities adds value and keeps us actively involved in, and informed of, all phases of the development process[18].

Periodic analysis and quality measurements of software products throughout the life-cycle are very important to manage and improve software quality [3] [4]. Software development departments are faced with some difficulties when using the tools and techniques for analysis and measurement. The main problems are typically as follows:

1) It is often not easy to understand and utilize analyzed results and measured data obtained by measurement tools and techniques.
2) Most development departments often do not have enough time to evaluate the tools and to prepare the environment needed to apply them practically.
3) Continuous activity and repeated experience are required to acquire the know-how for using tools and measurement data effectively and incorporating this in a timely fashion with the software development process [3].

The software industry has performed a significant amount of research on improving software quality. Also development team skill has a significant effect on the quality of a software product [1] [12]. Organizations involved in such research include the National Institute of Standards and Technology (NIST), the National Space and Aeronautics Administration (NASA), the Motor Industry Software Reliability Association (MISRA), and many others.

NASA is one example where getting the software right the first time is critical since they may only get one chance. Applying the best practices of industry including coding standards, software tools, configuration management and other practices will produce better quality code in less time. Good quality code will also be easier to write, understand, maintain and upgrade. This document is a template. An electronic copy can be downloaded from the conference website. For questions on paper guidelines, please contact the conference publications committee as indicated on the conference website. Information about final paper submission is available from the conference website.

II. BACKGROUND AND RELATED WORK

Software complexity is one branch of software metrics that is focused on direct measurement of software attributes, as opposed to indirect software measures such as project milestone status and reported system failures. Eldrandaly (2008) at [17] discussed different quality models and how industries should choose these models according to their software. He introduced a prototype knowledge based advisory system for checking SQA of software industries. In this prototype, different steps were introduced through which industries or their members can choose quality model according to their requirements. Farooq et al. (2011) as [18] explained the importance of software quantitative and qualitative metrics. In this study, software characteristics were explained and discussed how they are tested using software metrics. This study emphasized on software testing process. Iftikhar and Ali (2011) at [19] deliberated the role of quality measurement in software industries of Pakistan. It was discussed how quality assurance is measured in different industries and how they are compared. In this study, a survey has been conducted to differentiate more experience firms from less experience firms on the basis of quality assurance.

On the other hand, there are hundreds of software complexity measures, ranging from the simple, such as source lines of code, to the esoteric, such as the number of variable definition/usage associations.

1) Coding standards for software

Every organization that develops software should use a coding standard document this coding standards document tells developers how they must write their code. Instead of each developer coding in their own preferred style, they will write all code to the standards outlined in the document. This makes sure that a large project is coded in a consistent style, parts are not written differently by different programmers. Not only does this document make the code easier to understand, it also ensures that any developer who looks at the code will know what to expect throughout the entire application [5]

2) ISO Approach to Software Quality Management

According to ISO 9126, quality is defined as a set of features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs [5]. ISO 9000 describes quality assurance elements that can be applied to any business regardless of the products or services offered. The ISO 9000 quality assurance models treat an enterprise as a network of interconnected processes. For a quality system to be an ISO-compliant, these processes must address the areas identified in the standard. ISO 9000 describes the elements of a quality assurance system in general terms. The overall quality of a product can be then expressed by a combination of the a set of six independent high-level quality characteristics: Functionality, Reliability, Usability, Efficiency, Maintainability and Portability which are defined as a set of attributes of a software product by which its quality is described and evaluated[4].

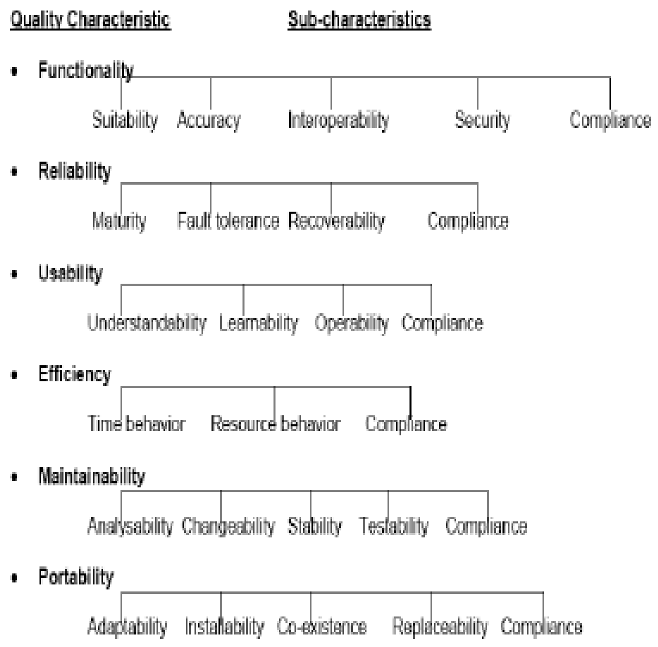


Figure 1. The Quality Characteristic and sub-characteristics

3) IS0 91 26 specify the attributes that insure high quality software [5]:

a) Functionality: Are the required functions available in the software?

b) Reliability: How reliable is the software?

c) Usability: Is the software easy to use?

d) Efficiency: How efficient is the software?

e) Maintainability: How easy'is it to modify the software?

f) Portability: How easy is it to transfer the software to another environment?

High quality software has all of these attributes.

III. SOFTWARE TOOLS

The Software quality evaluation tool, ESQUT, measures the quality metrics which are listed previously. A software tool properly used makes the job easier. Tools may be used to help in understanding the code and find problems quickly. Code browsers, static error analysis and rule enforcement tools are discussed in this section.

A code browser is much more powerful than the typical editor included with a compiler. It enables one to step through the code with just a few mouse clicks. It can also display a tree hierarchy of the function calls as shown in Figure 2using the Understand far C++ tool.

3.1. Code Browser Tools

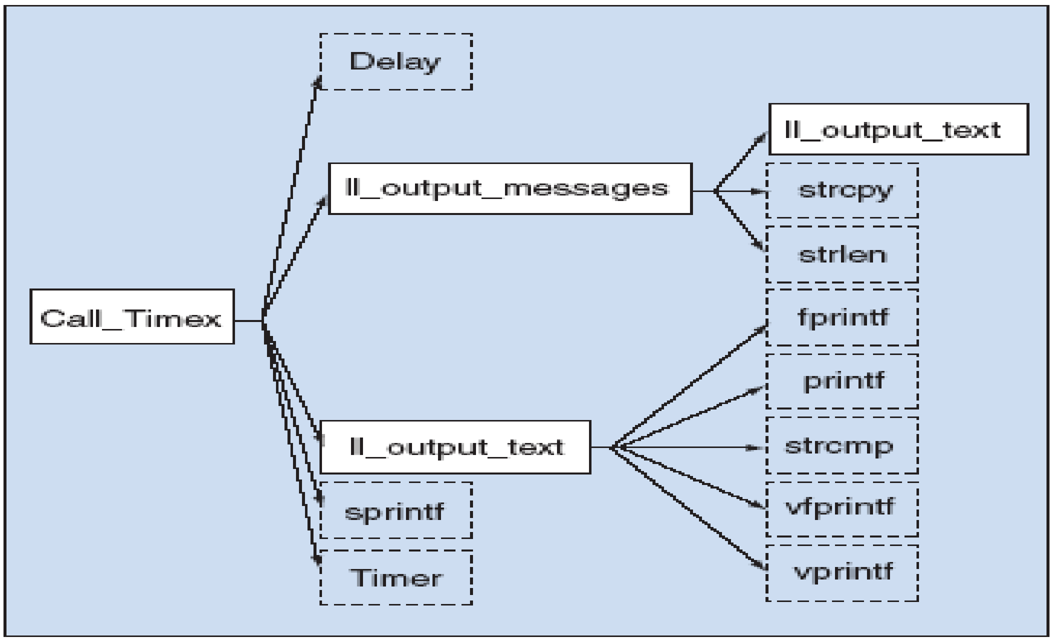


Figure 2. Sample call_Timex Hierarchy Diagram

It will also display unused functions and unused variables. A browser can also display the definition of a variable and all the places it's used in a cross reference table. A code browser may also have metric capability such as Understand for C++.

3.2 Static Error Analysis Tools

A 'C' compiler will produce a list of warnings and error messages catching most syntactic problems. Always set error checking to the highest level of detection and correct all the problems causing errors or warnings. This is a good start but a number of errors still get through. A' static analysis tool such as PC-lint is required to catch the rest of the errors. A lint tool will find errors in the code that compilers miss and warn you of many possible problems caused by common coding mistakes.

3.3. Rule Enforcement Tools

Organizations like MISRA have developed sets of rules for programmers to follow to minimize defects. Examples include: Don't use "goto" statements and don't comment out code. A software tool such as Code Wizard has over 500 rules to compare with code. Some programmers don't like to follow rules because it "cramps their style" or it takes too much time to learn the rules. There is a disparity between personal preference and following rules that lead to better quality code.
For example, the placement of curly is supports after "if," "for," and "while" statements is a personal preference, but leaving them out may lead to programming errors.

The rule enforcement process can be tedious and cumbersome without the use of automated tools. After the code has been compiled and found to be error free, a rule-formatting tool can be used to automatically reformat the source to conform to the rules. Such tools are few and far between.

IV. SOFTWARE QUALITY METRICS

Application of software metrics has proven to be an effective technique for estimating, assessing and improving software quality and productivity [7], there is many types of software metrics [2]. Two general types of software metrics are project management and software quality metrics. Project management metrics are used to track the progress of software development and estimate the target completion date. Software quality metrics provide a risk assessment of the software defects [11]. Project management metrics are quantitative but may also be used in calculating software quality metrics. What is the purpose of this metrics? Examples of the purposes include [11]:

- Facilitating private self-assessment and improvement.
- Evaluating project status (to facilitate management of the project or related projects).
- Evaluating staff performance.
- informing others (e.g. potential customers) about the characteristics (such as development status or behaviour)

5.1. Project Management Metrics

Useful project management metrics count functions and lines of code completed versus the number of outstanding defects remaining. Project management metrics can be used to keep track of costs and detect improvements made to the development process. Software quality management help organizations ensure that all software development activities meet uniform expectations around security, reliability, performance and maintainability.

V. THE PROPOSED APPROACH

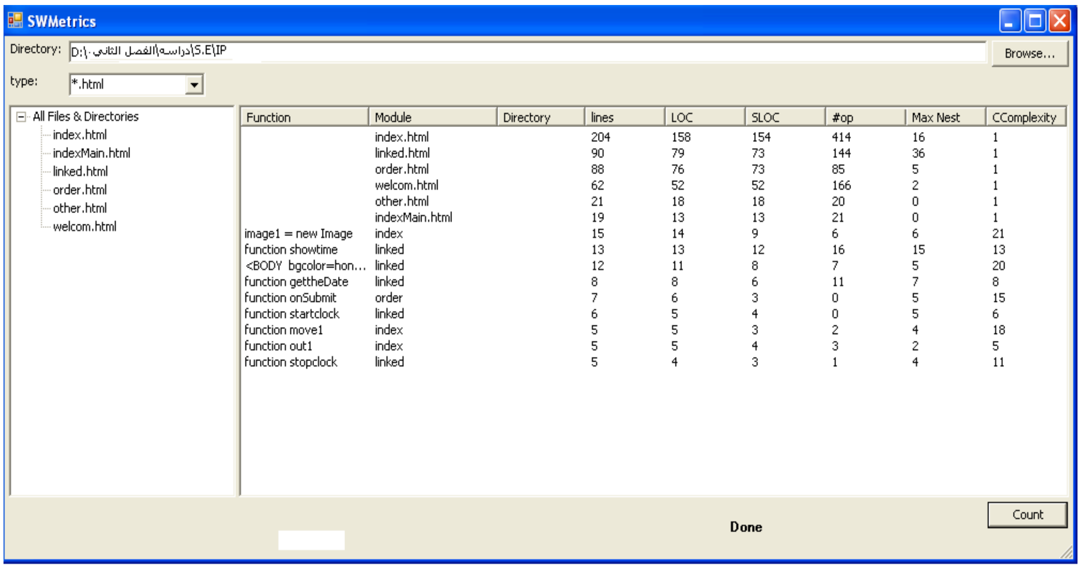
In this research, we develop a tool for the quality measurement of software by using Microsoft Visual Studio-C# language. The tool working for many format file language, therefor we can select from the list in the application, this format file like {.vb, .java, .cs, .aspx, .html, .c;h;cpp, .jsc). Then the calculation of the metrics will be determinate, which are the metrics is Line Of Code (LOC) it will be compute all the line of code in the software including the comment line. And the Source Line Of Code (SLOC) it will be compute the line of source code in software without comment line, Number of operation (# op) it compute all operation in the software like (+ , - , / , ….) , this metric will be needed in the determine the complexity of the system.

The more important s in this tool is to calculate the complexity of the system base the more general metrics, which is the cyclomatic metric of quality measurement of software is one of the most popular metrics in SW development life cycle (SDLC). A broad measure of soundness and confidence for a program, it measures the number of linearly independent paths through a program module.

The McCabe Cyclomatic Complexity is determined by counting the number of basic paths through a function and it is calculated using the equation: If G is the control flow graph of program P, and G has e edges (arcs) and n nodes, then:

$$\text{Cyclomatic number } V(G) = e - n + 2$$

Where e and n are the number of edges and nodes in the control flow graph



Directory: D:\.الفصل الثاني\دراسه\S.E\IP

type: *.html

Function	Module	Directory	lines	LOC	SLOC	#op	Max Nest	CComplexity
	index.html		204	158	154	414	16	1
	linked.html		90	79	73	144	36	1
	order.html		88	76	73	85	5	1
	welcom.html		62	52	52	166	2	1
	other.html		21	18	18	20	0	1
	indexMain.html		19	13	13	21	0	1
image1 = new Image	index		15	14	9	6	6	21
function showtime	linked		13	13	12	16	15	13
<BODY bgcolor=hon...	linked		12	11	8	7	5	20
function gettheDate	linked		8	8	6	11	7	8
function onSubmit	order		7	6	3	0	5	15
function startclock	linked		6	5	4	0	5	6
function move1	index		5	5	3	2	4	18
function out1	index		5	5	4	3	2	5
function stopclock	linked		5	4	3	1	4	11

Figure 3. Snapshot of the Main Menu of the system

Figure 3 shows the snapshot of the prototype of the software with the sample example of the file in format html in web application and it has files 1 php. In the example we see the computing of the all function of the software in separately, and for all the metrics for every specific metrics the column viewed the number this number it describe the function for the name of the column name.

The Cyclomatic complexity metric can be applied in several areas:

1) Code development risk analysis: While code is under development, it can be measured for complexity to assess inherent risk or risk build up.

2) Change risk analysis in maintenance: Code complexity tends to increase as it is maintained over time. By measuring the complexity before and after a proposed change, this build-up can be monitored and used to help decide how to minimize the risk of the change.
3) Test Planning: Mathematical analysis has shown that Cyclomatic complexity gives the exact number of tests needed to test every decision point in a program.
4) Reengineering: Cyclomatic complexity analysis provides the knowledge of the structure of the operational code of a system.

The risk involved in reengineering a piece of code is related to its complexity.

The independent of the complexity and size of software by line code

There is a big difference between complexity and size. There is no consistent relationship between the two. Just as 10 is a common limit for Cyclomatic complexity, 60 is a common limit for the number of lines of code. Many modules with no branching of control flow (and hence the minimum cyclomatic complexity of one) consist of far greater than 60 lines of code, and many modules with complexity greater than ten have far fewer than 60 lines of code.

Thus, although the number of lines of code is an important size measure, it is independent of complexity and should not be used for the same purposes.

VI. CONCLUSION

In this paper, we proposed a prototype to enhancing the effectiveness of testing and to improve the software quality. Software development process must make transitions to higher software culture. Software testing techniques, methodologies, tools and standards can only aid to measure software complexity but it is the management and the people involved who have to plan for and carry out effective testing. In this paper we compute a metrics LOC, SLOC and Complexity based the Cyclomatic of quality measurement for many format languages of source of code. This term paper is primarily is concerned on software development process that affect software quality management, Software quality metrics, Software measurement process, and quality models.

REFERENCES

[1] Nasib S. Gill," Factors Affecting Effective Software Quality Management," Revisited.ACM SIGSOFT Software Engineering, 1 March 2005 Volume 30 Number 2

[2] homas B. Hilburn, Massood Towhidnejad," Software Quality: A Curriculum Postscript?" SIGCSE 2000 3/00 Austin, TX, USA, 2000 ACM.

[3] Takeshi Tanaka, Minoru Aizawa, Hideto Ogasawara, Atsushi Yamada," Software Quality Analysis & Measurement Service Activity in the Company," Software, IEEE,(summer 1998)

[4] G.Zayaraz, Dr. P. Thambidurai, Madhu Srinivasan, Dr. Paul Rodrigues, "Software Quality Assurance through COSMIC FFP," ACM SIGSOFT Software Engineering September 2005 Volume 30 Number 5

[5] J. L. Anderson Jr.," USING SOFTWARE TOOLS AND METRICS TO PRODUCE, BETTER QUALITY TEST SOFTWARE," 2004 IEEE AUTOTESTCON

[6] William A.Ward, Jr.," Some Observations on Software Quality," February 26, 1993 to August 31, 1998.

[7] Shahid Nazir Bhatti J. Kepler University Linz.," Why Quality? ISO 9126 Software Quality Metrics (Functionality) Support by UML Suite ACM SIGSOFT Software Engineering March 2005 Volume 30 Number 2

[8] DROMEY, R. GEOFF, ''Cornering the Chimera,'' IEEE Software, vol. 13, no. 1, pp. 33-43, January 1996.

[9] ARTHUR, L. J., Improving software quality-an insider's guide to TQM, John Wiley & Sons, New York, 1993.

[10] F Georgios Gousios, Vassilios Karakoidas., ''Software Quality Assessment of Open Source Software1,2 ',PCI 2007, volume A, pages 303–315, Athens, May 2007

[11] Cem Kaner, Senior Member, IEEE, and Walter P. Bond," Software Engineering Metrics: What Do They Measure and How Do We Know?", 10TH INTERNATIONAL SOFTWARE METRICS SYMPOSIUM, METRICS 2004

[12] Justin M. Beaver , Guy A. Schiavone ," The Effects of Development Team Skill on Software Product Quality ",ACM SIGSOFT Software Engineering Notes, May 2006 Volume 31 Number3.

[13] Hideto Ogasawara, Atsushi Yamada, Michiko Kojo." Experiences of Software Quality Management Using Metrics through the Life-Cycle", Proceedings of ICSE-18,!EEE 1996.

[14] Suite, "Why Quality? ISO 9126 Software Quality Metrics", ACM SIGSOFT Software Engineering Notes Volume 30 Numbers 2 March 2005.

[15] Wikipedia: http://www.wikipedia.org

[16] Ian Sommerville. Software Engineering, 8th Edition. Addison Wesley, 2006.

[17] Eldrandaly, K., 2008. A knowledge-based advisory system for software quality assurance. Int. Arab J. Inform. Technol., 5(3): 304-310

[18] Farooq, S.U., S. Quadri and N. Ahmad, 2011. Software measurements and metrics: Role in effective software testing. Int. J. Eng. Sci. Technol., 3(1): 671-680.

[19] Iftikhar, A. and S.M. Ali, 2011. Software quality assurance a study based on Pakistan's software industry. Pak. J. Eng. Technol. Sci., 1(2): 65-73.