Volume 5, Issue 10, October-2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Evaluation of Quality of Source Code By Code Readability

[1]Deepa Dhabhai, [2]Dr. A. K. Dua, [3]Anil Saroliya, [4]Dr. Rajesh Purohit
[1, 2, 3] Amity University, Rajasthan, India
[4] MBM University, Jodhpur, Rajasthan, India

Abstract: The Maintainability of software is an important quality, which impacts the cost and availability of a system. One of the important parameter for software to be maintainable is readability of the source code, because the changes required in the system are eventually to be reflected through source code. Many times the code submitted by a programmer to his/her manager is rejected due to lack of readability, and the process is repeated. In some of the organization, to improve readability of a source code, several tools are available. e.g. Uncrustify , Artistic Style.

In this work some of the parameters are identified to evaluate the readability of a given source code, and a tool is developed to evaluate the code readability. By evaluating the code readability, before and after applying the code beautifier, the effectiveness shall be calculated. For this an experiment is carried out for five simple programs, each program is developed in two versions, basic version and improved version. The readability of basic version and improved version is recorded and the improvement index is calculated. The results are shown through graphs and found satisfactory.

Keywords: Software Quality, Code Readability.

I. INTRODUCTION

Readability will be defined as a person's judgment of understanding a text. The essential issue in maintaining the software system quality is readability and the readability of a program is expounded to its maintainability. Wherever the price of a software package in the full life cycle the upkeep can consume around 70%[1]. In maintenance of the software system each the source code readability and documentation readability play a essential role. On alternative hand some researchers have noted that the act of reading code is that the most long part of all maintenance activities[2]. As of the fashionable software system engineering, maintaining software system often means that evolving software system and modifying existing code. Readability is another necessary attributes of software package systems that offers substantial have an effect on software system maintainability. Maintenance of a less legible source code is tougher than a source code that has a lot of legible source code.

1.1 Software Quality-

Software quality comprises six main attributes.

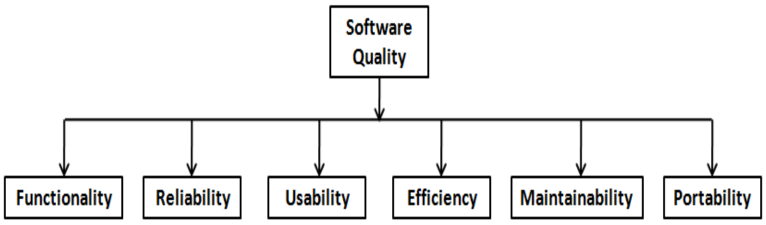


Fig.1.1. Software quality attributes.

These attributes can follows:
-Functionality-it is defined as the capability to provide functions, meet stated and understood needs when the software is used.
-Reliability- -it is defined as to provide failure-free and reliable service.
-Usability -it is defined as the to be understood, learned, & used.
-Efficiency- -it is defined as to provide required representation of relative Amount of resources used.
-Maintainability- -it is defined as to be improve for purposes of making corrections, refinement, or adaptation. Readability term comes under the maintainability.
-Portability-it is defined to be adapted for different particularized environments without applying actions.

1.2 Software Metric-

It is defined as the process of measuring software. Software metric is depends on special kind of analysis based on source code. Software quality measure by many metrics, in this study our goal to collect simple metrics use to identify source code readability and provide results.

The Flesch-Kincaid Grade Level [3], the Gunning-Fog Index [4], the SMOG Index [5], and the Automated Readability Index [6] are few examples of readability metrics for ordinary Text. All of these metrics are based on basic factors such as average syllables per word and average sentence length..We study a descriptive model of software system readability supported easy options that may be extracted mechanically from programs. This model of software system readability correlates powerfully with human annotators and additionally with external (widely available) notions of software system quality, like defect detectors and software system changes.

II. IMPORTANCE AND RELEVANCE OF THE STUDY

Dijkstra,, explain that the readability of a program depends for the most part upon the simple sequences (e.g. goto unnecessarily complicates understanding of program) and employed that notion to assist inspire his top-down approach to system design. According to Raymond P.L. Buse[7], they need chosen to focus on readability directly each as a result of it's an idea that's severally valuable, and conjointly as a result of developers have nice management over it. They show that there's so a significant correlation between readability and quality. The main contributions of that paper are:

- A method for the development of an automatic software package readability metric supported native code options.
- A survey of one hundred twenty human annotators on one hundred code snippets that forms the premise of the metric given and evaluated during this paper. They are unaware of any published software package readability study of comparable size. They directly evaluate the performance of our model on this information set.
- a collection of several experiments that reveal correlations between metric for readability and external notions of computer code package quality in addition as defect density.
- A discussion of the choices involved in our metric and their relevancy software package engineering and programming language style.

This article also includes some previous work[8] that : A further experiment correlating our readability metric with lot of natural notion of software package quality and defect density specific human mentions of bug repair, using software package version management repository data they coarsely separate changes created to deal with bugs from different changes. They found that low readability correlates lots of powerfully with this direct notion of defect density than it'll with the previous approach of victimisation potential bugs according by static analysis tools.

Software quality attributes made public by ISO-9126, maintainability is recognized by many researchers as having the most important impact on software package quality (Troy, 1995). At the 1992 software package Engineering Productivity conference, a Hewlett- Packard government express that sixty – eightieth of their analysis and development worker were resulted maintaining forty – fifty million SLOC (Troy, 1995). Glass explained that software package maintenance consumes from forty – eightieth of the overall software package price, with a mean of hr. Jakob Boehm and Basili (2001) report a mean of seventieth.

2.1 Readability and complexity

It is necessary to notice that readability isn't a similar as complexness that some existing metrics are through empirical observation shown helpful. Whereas software package complexness metrics generally take into consideration the scale of categories and strategies, and also the extent of their interactions, the readability of code is based totally on native, line-by-line factors and complexness factors, on the opposite hand, could have very little relevancy what makes code comprehendible to humans.

N. Nagappan and T. Ball analyzed [9] that artificial code complexity metrics directly to defects is hard, but not impossible. Though each involve native factors such as indentation, readability is also different from coding standards (e.g.,[10],[11],[12]),conventions primarily supposed to facilitate collaboration by maintaining regularity between code written by completely different developers.

III. PROBLEM STATEMENT

In this study, we have target to code readability directly because this concept is independently valuable for developers and they also have a great control over it.

Software readability could be a property that describes how easily a given piece of code are often read and understood. Since readability will act maintainability, quality, etc., programmers are terribly involved concerning the readability of code. If automatic readability checkers may be engineered, they may be integrated into development tool-chains, and therefore continually inform developers concerning the readability level of the code. Unfortunately, readability could be a subjective code property, and not amenable to direct machine-driven computation. in a very recently printed study, Buseet al. asked a hundred participants to rate code snippets by readability, yielding arguably reliable mean readability several every snippet; they then engineered a reasonably advanced prognostic model for these mean scores employing a giant, various set of directly measurable source code properties.

There is additionally a significant downside that the design of committal to writing of code user} are completely different from different programmer. some technologist are create their code correct readable by creating use of comments, indentation etc, while other is also prepare writing the code. We need to measure the effectiveness of the code produce by the beautifier tool. To which extent the code is to be beautified is also a issue.

IV. PROPOSED SOLUTION

It is important to notice that readability and complexity are not equivalent, that some existing metrics are through empirical observation shown helpful. Whereas code complexness metrics usually take under consideration the scale of categories and ways, and therefore the extent of their interactions, the readability of code is predicated totally on native, line-by-line factors. On the opposite hand, might have very little regard to what makes code perceivable to humans. We avoid to examine complex features like complexity and intend to focus on “low level” details of readability of code, we mean only to show that local features have strong impact on readability by extension on software quality.

For calculating readability metric we need some short program examples called “snippets”. We have human annotators who select these snippets. We also have some limitations for snippet selection like snippet should be short so that it they may obscure important readability considerations.

This paper aims at methodology used to define the new metric is employed here [13].First of all some rules which will be used to live readability are obtained from senior code engineers. It mean to asked to few senior code engineers to produce criteria for readability metric and picked up their responses, and we also modify these metric according to new requirement. When creating an initial list, the list is valid and eventually seven rules are thought-about to be a part of new metric. The seven modified rules and their representative notations are in table one. this system is impressed by

No.	Rule use to calculate readability	Notations
1.	Total Lines of Code	TLOC
2.	Line Length	LL
3.	No. of Comment Line	NOCL
4.	No. of Blank Lines	NOBL
5.	No. of Lines After Semicolon	NASL
6.	No. of Space After Directive Statements	SAD
7.	NO. of Methods	NOM

Fig. 4.1- Table for Rule use to calculate readability

As by given table, the notations for all seven rules are bestowed. The new code readability metric is represented by

CR= TLOC+ LL+ NOCL+ NOBL+ NLAS + BSAD+ NOM

here CR is Code Readability . As a part of the methodology to outline new metric and implement it, a example .net application is made mistreatment Visual Studio that takes a supply file as input and extracts values for TLOC, LL, NOCL, NOBL, NLAS, BSAD and NOM. subsequently those values square measure substituted within the equation four that is employed to cipher the readability of given source code. The result are in proportion of readability. A lot of during this proportion, a lot of the source code readability is. When implementing the new metric, evaluated the Code readability metric by this and then also calculate the code readability improvement index.

Code Readability
Improved Index=(R-R')/R'

For this, we have taken 5 samples categories, containing 2 programs each.

One program is less readable code and another program is the beautified code using a beautifier tool as we call improve program. And both codes are the analyzed on the basis of proposed matrices.

The improve program is given by a software’s which are known as Code beautifier. By evaluating the code readability, before and after applying the code beautifier, the effectiveness shall be calculated.

We also analyze that our final result of code readability improvement index is improve or what range they need in improved program, is only decided by the program manger team. By code readability improvement index we can analyze that our improve program is more readable then basic program we take in input.

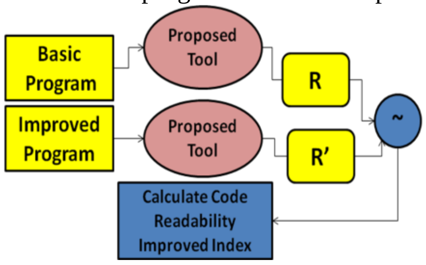


Fig.4.1 - Block diag. of proposed concept.

V. IMPLEMENTATION

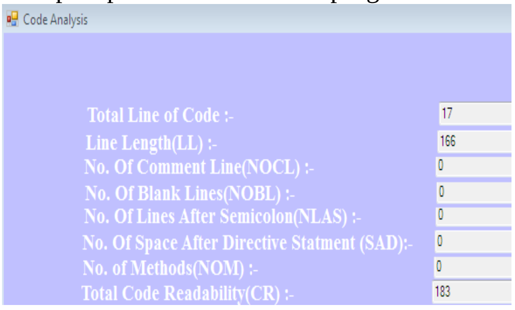
We have implemented in our solution a system which will implement the proposed readability metric .In this form, there are select the operation which we want to perform.

There are two menus in this form,

1. Code analysis: This menu option is used to perform the code analysis.
2. Exit

Steps of followed by program:

Step 1:- First we have taken two programs C/C++ mainly or we also give input program.
Step 2:- Suppose we have taken one programs of C. After it we load that one program in loading box, execution will start automatically and calculate results according to readability.
Step 3: We will take the same code and beautify it using the beautifier .Now load program, again calculate readability metrics.
Step 4:-After completing both program execution we compare results related to that programs mainly and take output in graphical form.
Step 5:- we also analyzed code readability improvement index by using proposed formula.
Step 6: - Evaluate the final results, and Repeat process for different programs and record results.

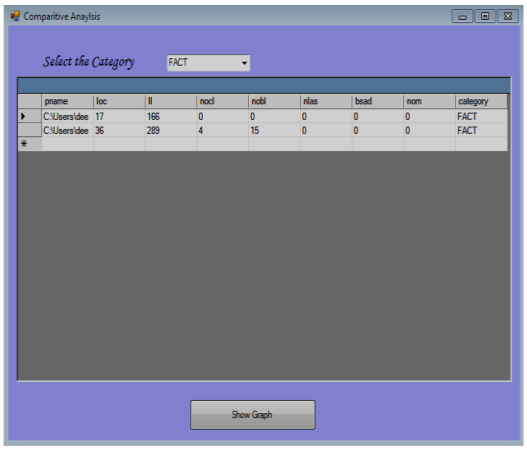


Total Line of Code :-	17
Line Length(LL) :-	166
No. Of Comment Line(NOCL) :-	0
No. Of Blank Lines(NOBL) :-	0
No. Of Lines After Semicolon(NLAS) :-	0
No. Of Space After Directive Statment (SAD):-	0
No. of Methods(NOM) :-	0
Total Code Readability(CR) :-	183

Fig5.1.- Results of basic Program calculation of Readability metric

This is results of code readability metric presented when we do code analysis of program, similarly Readability of improve Program also calculated by this.

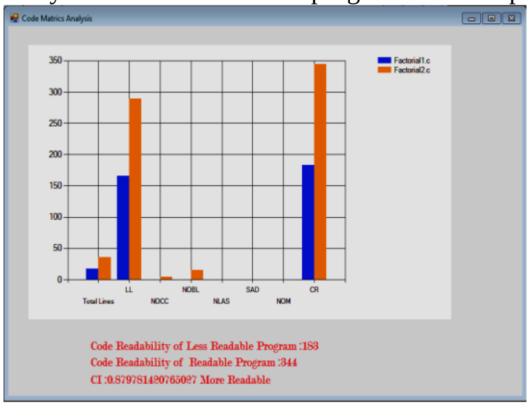
Compare programs



	pname	loc	ll	nocl	nobl	nlas	bsad	nom	category
▶	C:\Users\dee	17	166	0	0	0	0	0	FACT
	C:\Users\dee	36	289	4	15	0	0	0	FACT
*									

Fig.5.2-Results of Compare Two Programs

This form will first present the readability metric values for both programs and compare them graphically.



Code Readability of Less Readable Program :183
Code Readability of Readable Program :344
CI :0.879781420765027 More Readable

Fig.5.3-Results of programs and compare them graphically.

In this section Program find the result of code readability of two different program and analyze the results according to readability metrics and then also calculate code readability improvement index for more readable program or less readable program. In this study, according to this concept analyze five set of sample program each set have two program designed by different developer, and calculate readability and improvement index and evaluate the performance of basic and improved version of program.

VI. CONCLUSION

In this paper we presented a technique for calculating code readability based on judgments of software engineers, and also calculate the improved code readability index.Properly formatted program is always easy to understand so for proper development of new code and maintenance of the existing code it is required that our program should be proper indented and beautified code . In this proposed work a tool evaluate the improvement in readability of two versions of program. Software engineer use proposed metric in development life cycle to compute readability and readability index, readability can improves the quality of software product.

REFERENCES

[1] B. Boehm and V. R. Basili, "Software defect reduction top 10 list" Computer, vol. 34, no. 1, pp. 135–137, 2001.

[2] K. Aggarwal, Y. Singh, and J. K. Chhabra, "An integrated measure of software maintainability, " Reliability and Maintainability Sympo-sium, pp. 235–241, Sep. 2002.

[3] R. F. Flesch, "A new readability yardstick," Journal of Applied Psychology, vol. 32, pp. 221–233, 1948.

[4] R. Gunning, The Technique of Clear Writing. New York: McGraw

[5] G. H. McLaughlin, "Smog grading – a new readability," Journal of Reading, May 1969.

[6] J. P. Kinciad and E. A. Smith, "Derivation and validation of the automated readability index for use with technical materials," Human Factors, vol. 12, pp. 457–464, 1970.

[7] Raymond P.L. Buse,Westley weimer, "Learning a Matric for code Readability". TSE special issue on the ISSTA 2008 Best paper.

[8] R.P.L Buse and W.R. Weimer "A metric for software readability" in international symposium on software Testing and analysis,2008.

[9] N. Nagappan and T. Ball "Use of relative code churn measures to predict system defect density" conference on Software engineering, 2005, pp. 284–292.

[10] S. Ambler, "Java coding standards," Softw. Dev., vol. 5, no. 8, pp.67–71, 1997.

[11] L. W. Cannon, R. A. Elliott, L. W. Kirchhoff, J. H. Miller, J. M. Milner, R. W. Mitze, E. P. Schan, N. O. Whittington, H. Spencer,D. Keppel, , and M. Brader, Recommended C Style and Coding Standards: Revision 6.0. Seattle, Washington: Specialized Systems Consultants, Inc., June 1990.

[12] H. Sutter and A. Alexandrescu, C++ Coding Standards: 101 Rules, Guidelines, and Best Practices. Addison-Wesley Professional, 2004.

[13] Rajendra namani ,Kumar J,"A new metric for code readability". IOSR Journal of computer Engineering(IOSRJCE).