International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 10, October 2012)

A Comparative Study of Maintainability Index of Open Source Software

Anita Ganpati[1], Dr. Arvind Kalia[2], Dr. Hardeep Singh[3]
[1,2] *Computer Science Dept., Himachal Pradesh University, Shimla, India.*
[3]*Guru Nanak Dev University, Amritsar, India.*

Abstract - The development of Open Source Software (OSS) is fundamentally different from the proprietary software. In the OSS development scenario a single developer or group of developers writes the source code for the first version of the software and make it freely available over the internet. Then other developers are invited to contribute to the existing code for its next release. Making the source code of the software available on the internet allows developers around the world to contribute code, add new functionality, improvement of the existing source code and submitting bug fixes to the current release. In such a software development scenario the maintenance of the open source software is a perpetual task. Developing an OSS system implies a series of frequent maintenance efforts for debugging existing functionality and adding new functionality to the software system. The process of making the modifications to software systems after their first release is known as maintenance process. The term maintainability is closely related to the software maintenance because maintainability means the easiness to perform maintenance of the system. The most widely used software metric to quantify the maintainability is known as Maintainability Index (MI). In this study the MI of four most popular OSS namely Apache, Mozilla Firefox, MySql and FileZilla for fifty successive releases was empirically investigated. The MI in terms of software metrics namely Lines of Code (LOC), Cyclomatic Complexity (CC), and Halstead Volume (V) was computed for all the fifty successive versions of four OSS. The software metrics were calculated using Resource Standard Metrics (RSM) tool and Crystal Flow tool. It was observed from the results that the MI value was the highest in case of Mozilla Firefox and was the lowest in the case of Apache OSS.

Keywords **- Open Source Software, Maintenance, Maintainability Index, Lines of Code, Halstead Volume, Cyclomatic Complexity.**

I. INTRODUCTION

In a changing environment, software is also prone to change. Software maintenance is one of the important processes in the software life cycle. The purpose of the software maintenance is to keep software operational, to prevent and correct faults in the software and enhance the functionality of the software. Maintenance refers to the modifications made to software systems after its first release. It is not possible to develop a software system that does not need maintenance because change is an inherent characteristic of software systems [4].

Maintenance is defined by the IEEE as "the process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment" [5]. The maintainability is closely related to the software maintenance because the easiness with which the maintenance of the system is performed is called maintainability. There have been several attempts to quantify the maintainability of a software system. The most widely used software metric which quantifies the maintainability is known as Maintainability Index (MI). The Maintainability Index was given by Oman and Hagemeister and it consists of a number of software metrics which predict software maintainability [8]. It consists of a polynomial expression and results in a number indicating the overall system maintainability. Khan et.al. in their book defined MI as a combination of software metrics namely McCabe's Cyclomatic Complexity (CC), Halstead's Volume (V) and Lines of Code (LOC) that affect maintainability of the software [6]. MI may change because of new code added to the existing source code due to bug fixing or other corrective actions. According to Coleman a MI value above 85 indicates that the software is highly maintainable, a value between 85 and 65 suggests moderate maintainability, and a value below 65 indicates that the system is difficult to maintain [1].

II. LITERATURE SURVEY

According to Coleman et al. the maintainability index is represented as a function of Average Halstead Volume (V) per Module, Average Cyclomatic Complexity per Module, Average Lines of Code per Module and Average Percent of Lines of Code per Module [2].

According to Misra, the MI can be used as an indicator of maintainability for Object Oriented Systems [7].

Tahvildari *et al*. [11] discussed that for calculating the maintainability index other models namely, a three-metric model based on (Fan-Out, Data Complexity and McCabe Cyclomatic Complexity), a single-metric model based on Halstead's effort and a four-metric model based on (Halstead's Effort, McCabe Cyclomatic Complexity, Lines of Code and Percent of Comments) can also be used.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 10, October 2012)

Hayes *et al.* [3] proposed a maintainability model that categorized software modules as "easy to maintain" and "not easy to maintain". Such categorization can help to identify the modules, which are not easy to maintain.

Oman *et al.* [9] proposed a software maintainability hierarchy, in terms of some maintainability indicators. According to hierarchy, Halstead Complexity and Cyclomatic Complexity are the indicators of maintainability.

According to [10], maintainability is an internal quality attribute which can be expressed as a function of Modularity, Portability, Readability, Testability, Reusability, Flexibility and Conformity.

According to Welker the object oriented systems have a fairly high MI due to the typical smaller module size. In his view smaller modules contain less operators and operands, less executable paths, and fewer lines of comments and code therefore the MI tends to be higher [12].

III. Objective Of The Study

The objective of this study is to comparatively analyze maintainability index for four open source software.

IV. Research Methodology

The source code of four open source software i.e. Apache, Mozilla Firefox, MySql and FileZilla was used in this study as the data source. The Apache HTTP Server is an open-source HTTP server for modern operating systems including UNIX and Windows NT. Mozilla Firefox is a free web browser developed by Mozilla Foundation for Windows, Linux, Mac operating system and many more. MySQL is the world's most popular open source database management system. MySQL is a database management system for the relational databases and hence called as relational database management system (RDBMS). FileZilla is free and cross-platform FTP software, consisting of FileZilla Client and FileZilla Server. Binaries are available for Windows, Linux, and Mac OS X. It supports FTP, SFTP, and FTPS. The various software metrics namely LOC, CC, Halstead Volume and MI of fifty versions of the open source software PHP required for studying the maintainability index change were calculated using software tools namely Resource Standard Metrics (RSM) and Crystal Flow.

V. Analysis

The Maintainability Index of the Apache, Mozilla Firefox, MySql and FileZilla software was computed over fifty successive versions. MI was computed by using the formula below[6]:

$$MI = 171 - 5.2 * \ln(aveV) - 0.23 * aveV(g) - 16.2 * \ln(aveLOC)$$

where

aveV= average Halstead Volume

aveV(g) = average Cyclomatic Complexity per module

aveLOC = average Lines of Code per module

The results for the MI for fifty versions of Apache, Mozilla Firefox, MySql and FileZilla are graphically depicted in the FIGURE 1. It is apparent from the values of the maintainability index that the MI of Mozilla Firefox has the highest value among all other software. The MI values ranges between 94 and 96. The value of MI above 85 means the Mozilla Firefox is highly maintainable. A high MI in case of Mozilla Firefox indicates that the software does not show any sign of code decay. The Filezilla OSS has higher values of the maintainability index than the Apache and MySql software but a lower maintainability index than the Mozilla Firefox software. This means that the Mozilla Firefox is more maintainable than the Filezilla, Apache and Mysql Software. The maintainability index is the lowest in case of the Apache software. The MI values in case of Apache are less than 85 which imply that the versions of Apache are moderately maintainable. The decrease in the MI value means the increase in the maintenance effort.

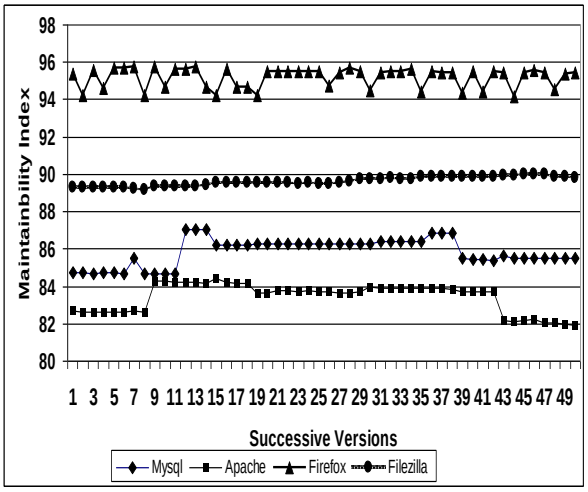

Figure. 1 Mi values of Mysql, Apache, Firefox, Filezilla

The increase in the maintenance effort is the sign of code decay i.e. if the efforts are not done to decrease the maintenance effort for these versions then in the successive versions the code will decay and ultimately that will require a complete reengineering of these versions.

However in case of FileZilla the values of MI are above 89 and are very consistent. The consistent values of MI indicate that there is not much change in case of maintenance efforts and the software versions are highly maintainable.

VI. CONCLUSION

The Maintainability Index of the Apache, Mozilla Firefox, MySql and FileZilla software was observed over fifty successive versions. From the results it was observed that the Mozilla Firefox has the highest maintainability index value and the Apache has the lowest maintainability index. The higher maintainability index value in case of Mozilla Firefox indicates that the versions of this software are highly maintainable. The lower values of MI in case of Apache web server is an indication of increased maintenance effort. The increased maintenance efforts in case of Apache web server is the sign of code decay which implies that if efforts are not done to decrease it then the software versions may require a reengineering in the successive versions.

REFERENCES

[1] Coleman, D. 1992. Assessing maintainability. Proceedings of the 1992 Software Engineering Productivity Conference, pp. 525-532, San Jose, CA.

[2] Coleman, D., Ash, D., Lowther B. and Oman, P. 1994. Using metrics to evaluate software system maintainability, Computer 1994, Vol. 27(8), pp. 44–49.

[3] Hayes, J. H. and Zaho, L. 2005. Maintainability prediction a regression analysis of measures of evolving systems. Proceedings of 21st IEEE International Conference on Software Maintenance, pp.601- 604.

[4] Hsia , P., Gupta , A., Kung C., Peng, J. and Liu, S. 1995. a study on the effect of architecture on maintainability of object-oriented systems. IEEE Conference on Software Maintenance, Opio, France, pp. 4-11.

[5] IEEE Standard. Glossary of software engineering terminology, IEEE Std 610.12-1990.

[6] Khan, R. A., Mustafa, K. and Ahson, S. I. Software quality: concepts and practice. Narosa Publishing House.

[7] Misra, S. C. 2005. Modeling design/coding factors that drive maintainability of software system. Software Quality Journal, Vol. 13(3), pp. 297- 320.

[8] Oman, P. and Hagemeister, J. 1994. Construction and testing of polynomials predicting software maintainability. Journal of System Software, Vol. 24, pp. 251 – 266.

[9] Oman, P. and Hagemeister, J. 1992. Metrics for assessing a software system's maintainability. Proceedings of the Conference on Software Maintenance, IEEE Computer Society Press, pp. 337–344.

[10] Sneed, H. M. 2004. Analysis measurement and evaluation of existing legacy system. Tool Demonstrations of the 8th European Conference on Software Maintenance and Reengineering, IEEE Computer Society Press, Silver Spring MD, pp. 2–3.

[11] Tahvildari, L., Gregory, R. and Kontogiannis, K.. 1999. An approach for measuring software evolution using source code features. Proceedings of the Sixth Asia Pacific Software Engineering Conference, IEEE Computer Society Press, Silver Spring MD, pp.10–17.

[12] Welker, K. D. 2001. The software maintainability index revisited. In Crosstalk, Journal of Defense Software Engineering.