Life Cycle of Software Development Maintainability Attributes Model using Fuzzification

Poonam[1], Rekha[2]

[1]M. Tech. Scholar, [2]Assistant Professor
CBS Group of Institutions, Jhajjar, Haryana
Maharishi Dayanand University, Rohtak.

***Abstract*: Software Development goes through a number of phases. These phases together make a Life Cycle of Software Development. It is estimated that more that there are more than 100 billion lines of code in production in the world. As much as 80% of it is unstructured and not well documented. Maintenance can lessen these problems. Maintainability is the ability to keep the system up to date after deploy to the customer site. We studied a number of software maintainability measurement metrics and also new proposed techniques. In our research we focused on how to measure the software. After considering these factors we can conclude that how much software is maintainable. This means that how the maintenance cost can be reduced and how much efforts will be required to reduce the cost. So we will use a fuzzy logic to implement these factors. This found that fuzzy logic can be used to model uncertainty for these factors. Fuzzy logic is a way to deal with reasoning that is approximate rather than precise. Then by fuzzy logic we measured the maintainability. In our research, we considered experimental data. First we applied these factors on data and then by fuzzy logic we measured the maintainability. This work based on Rule Base consists of number of rules. Rule structure is like "If this and/or If this then this.**

***Keywords*: Software Development, Maintainability Attributes, Fuzzification**

Introduction

Maintenance is the concept most commonly correlated with more robust infrastructure and considerably lower long-term costs. Description of maintenance commonly find, for example: "The effort required to fix errors, boost efficiency or other attributes after delivery of a software program or component, The climate has been modified. Or "A program may be sustained if only minimal efforts are made to fix minor bugs."

Perhaps too easy, naturally. The second is particularly misleading, since in reality it is a tautology rather than a concept. The response would be "its correction needs little effort if you wondered what a minor bug is." In addition to this very native description, Specific metric methods aim to characterize main tenability as conformity with a collection of rules that suit the observable characteristics of the system, such as high consistency, minimal coupling, etc

The management of repair tools is made up of a small number of nations. Within a fixed time frame allocated to the repair engineers, the program should be transferred from one condition to another throughout the maintenance phase. Depending upon the type of maintenance issues, the program struggles or achieves optimum performance. Maintenance relies very much on the software form, as is commonly known. When the amount of object-oriented software programs increasing, managing such structures efficiently becomes increasingly essential for organizations. However, only a limited number of predictive models for object-oriented systems are currently available. The maintenance of information depends on quality and quantitative data. Existing management models organize data into a hierarchical structure with various dependent functions. However, the details used to evaluate the function may be unclear or may be completely unknown. It should also be measured with respect to volatility and incompleteness of the aggregate outcome, i.e. maintainability. Moreover, actual cases need an integration model that is capable of determining the effect on the main tenability of shifts in the interactions between hierarchical characteristics. In app creation, servicing always took a back seat. However, once the software is provided, it will be retained for its entire lifespan. Although application development tends to grow, more and more money is spent on infrastructure. A closer analysis is also required to strengthen our maintenance strategy. The standard of software can be described as the entirety of functionality and functions of a system that will fulfill a particular set of requirements. Computer consistency includes a range of considerations such as efficiency, reusability, portability and maintenance. Such indicators of consistency are eventually broken down to standard parameters that function as device attributes. Contrary to an assumption that much has been spoken regarding maintenance attributes, a detailed and standardized model of attributes doesn't provide it. Maintenance of software is described as a method to change current software while its primary functions remain intact. During its life cycle, every program must be updated during order to meet the consumer demand. Computer maintenances require the repair of bugs, capability improvement and replacement and optimization of redundant technologies. Through fulfilling additional criteria, software may maximize its worth through promoting its use, productivity and the usage of newer technology. While this is an essential job, the implementation is weak. This is also necessary to quantify maintainability that you are powerless to manage, but you cannot calculate. Maintenance is a calculation of program characteristics such as readability of source code, data consistency and coherence of source code and records. They should discuss how the "maintenance" of a machine component varies over the course of time and how calculations are carried out on manufacturing structures. Our notion of "conservancy," theories and solution is introduced.

In the 1970s and 1980s, numerous scholars researched maintenance in order to determine the explanation behind the demands for improvement and its related patterns and costs. These studies have culminated in the identification of many types of maintenance activities; these maintenance types and their impact on the expense and the efficiency of the program in operation. This was also first demonstrated that the classification of repair activities into divisions were more than mere correcting mistakes.

The functionality and function of this device would preferably not be compromised by repair operations; otherwise, any improvements will become extremely complicated and expensive to carry out. This is not the case with management in actual life, which also entails the ageing of the topic system; as Lehman's second rule states: "When a plan progresses, its layout continues to grow more complicated. That is not the case. Furthermore, a few writers take the fourth type of maintenance named proactive maintenance into consideration, and covers any adjustments rendered to one program component in order to render them more main tenable. It implies that there is little requirement for the improvement and simplification of the framework.

• Corrective Maintenance: corrective modification upon distribution of the electronic product to correct any faults that have been discovered. This discusses the removal of application errors.

• Adaptive maintenance: Modification of the computer software carried out during distribution in an evolving setting to maintain the software system operational. The program is tailored to a new world.

• Good repair: adjustment to efficiency or servicing of a software product after production. This includes program maintenance dependent on device needs changes.

• Proactive maintenance: software system alteration after distribution in order to detect and fix latent software product defects until they are successful. It manages the editing of records and allows the device more stable.

All machine improvements may be defined by these four maintenance styles. Repair correction is-: "Standard repair" while certain forms are called "Evolution of apps"

Need of Maintenance

- The program must be managed to insure that the consumer specifications are consistently fulfilled.
- Removed faults
- Technology upgrade
- Updates to execution
- Certain device control
- Technology for deployment residue
- Computer withdrawal.

Key Issues in Software Maintenance

If done on applications, the word maintenance takes on a sense which is somewhat different from the significance it takes in most other technological area. Most technical departments are also preparing to conduct upkeep in order to hold things in order. The key principle is the degradation of a mechanical artefact due to its usage and the passing of time. This also seeks to maintain the quality of the artefact in accordance with that which has been established and documented on publication.

Technical Issues

Limited understanding-: Limited understanding applies to the immediate perception that a software developer may create a technical adjustment or correction.

Testing: The effect of repeated full tests on a critical piece of software in terms of time and resources may be important.
Maintenance-: In order to minimize maintenance costs, maintenance features need to be defined, checked and monitored during software development activities.

Management Issues

Conformity with corporate target- Organizational priorities explain how program development expenditure returns should be illustrated.

Jobs-Staffing relates to how network engineering workers may be recruited and retained. System- Computer system is a collection of operations, procedures, processes, and transformations the computer and the related products are created and maintained by people.

Experimental Study And Results

Maintenance is generally seen to rely heavily on the type of data. We have attempted to assess device maintenance. With reasons and some empirical analysis, we have tried to show that the complexity of the program could not be calculated by commonly used complexity measures like code lines, Halsteade Program Science Metrics and others. Such indicators may not be ideal for calculating

software maintenance costs and workload. We also looked at the five software ventures of students in undergraduate engineering. After the projects have been considered, the various attributes in these projects apply, the values of the individual projects in all four parameters are different.

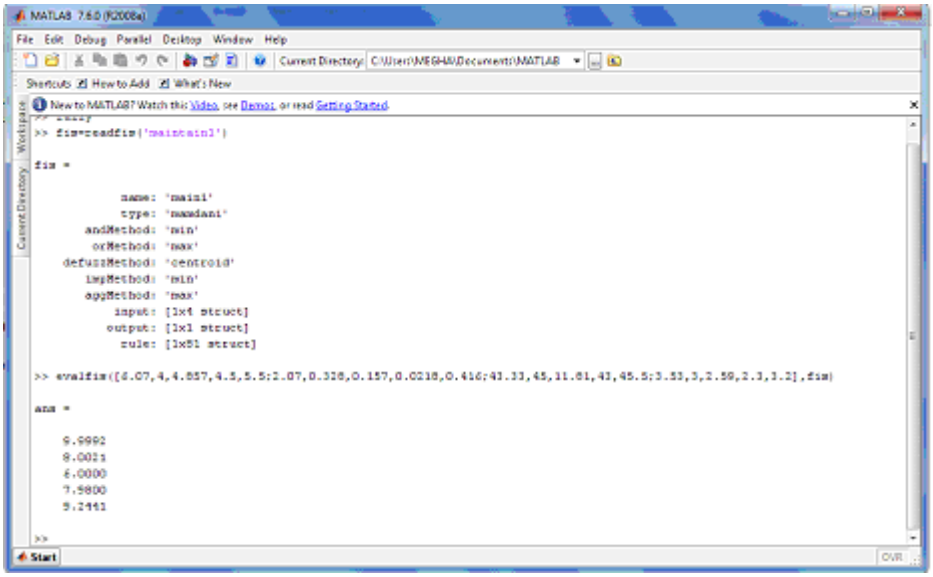

Figure Snapshot of Matlab

Result

There is a variable. This variable is available. The proposed fuzzy model is also used to estimate maintenance. The results are shown below.

P.No	LV	LS	CR	ACC	Maint.
1	6.07	2.07	43.33	3.53	9.9992
2	4	0.328	45	3	8.0021
3	4.857	0.157	11.81	2.59	6.0000
4	4.5	0.0218	43	2.3	7.9800
5	5.5	0.416	45.5	3.2	9.2441

Value of maintainability

The verification diagram is also displayed on various diagrams. In these figures, all four parameters are on one axis, and the items are not on the other axis. The co-relation between four parameters and main tenability is almost impossible. The maintenance of these four parameters is not predictable individually. The verification is done through the fuzzy model, and using the results shown in the table above, the combined values used for maintenance can also provide the best results on each input indicator.

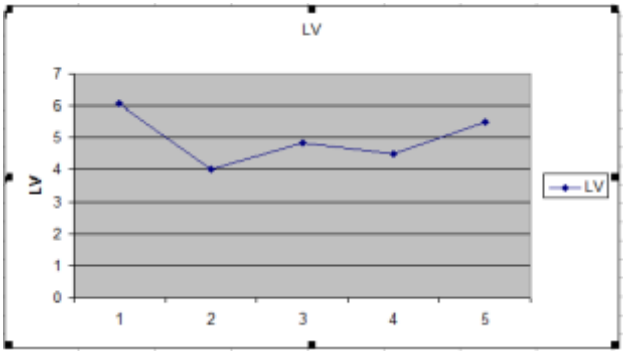

Figure Diagram of Live variables

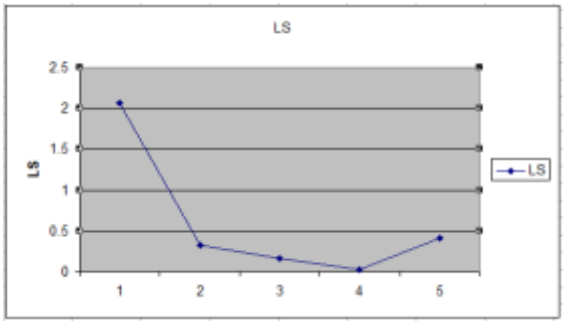

Figure Diagram of Live Span

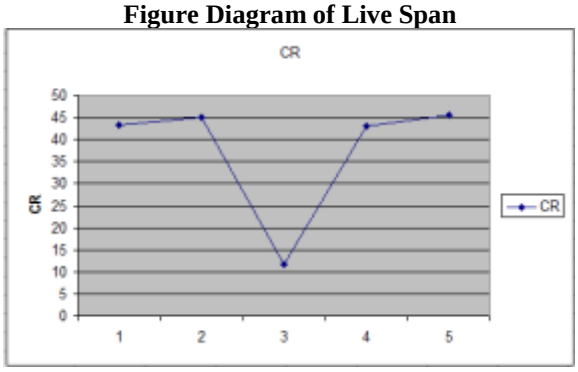

Figure Diagram of Comment Ratio

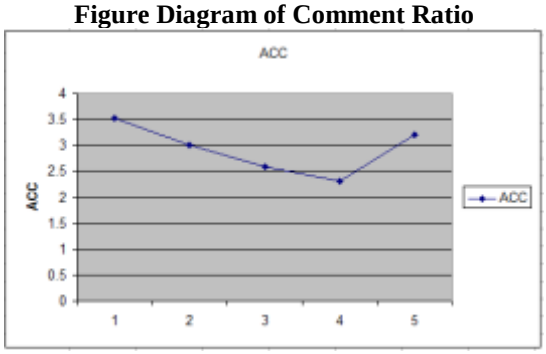

Figure Diagram of Cyclomatic Complexity

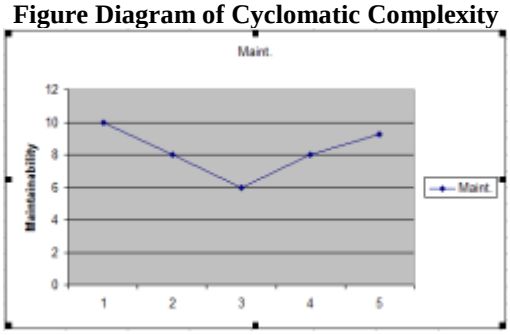

Figure Diagram of Maintainability

Conclusion and Future Work

This experimental provides information on the experimental arrangements of data and complete experimental arrangements. Data that are different factors are described. The segment Experimental results explains the experiment results. The maintenance importance of various projects in a table shape fits the different factors. The maintenance assessment using an excellent graph. There are typical metrics to measure the maintenance of software measures such as code line, the information science metrics of Halstead and the cyclomatic complexity of Mc Cabe. However, these metrics cannot be used to measure software maintenance. Maintenance depends very much on the software type, as is commonly seen. We tried to demonstrate by arguments and empirical study that the software's sophistication might not be calculated by traditional metrics. This report proposes a 4-parameter integrated analysis for software maintenance calculation. The time spent and resources required for the maintenance of software use approximately 40% to 70%. Through these parameters the study will evaluate how maintenance costs and efforts are reduced. We have therefore established a fuzzy model for software maintenance measurements. Further work to increase the accuracy of measurement in this field may be undertaken to build such a framework. We suggest that this model be tested in real time. The time needed to correct this error in the maintenance period is determined when any error is found in the project.

References

[1] Rikard Land Mälardalen "Software Deterioration And Maintainability – A Model Proposal" in 1995 University Department of Computer Engineer

[2] Khairuddin Hashim and Elizabeth Key " A Software Maintainability Attributes Model" Malaysian Journal of Computer Science

[3] C. van Koten 1 and A.R. Gray 'An application of Bayesian network for predicting object-oriented software maintainability' in 2005 Department of Information Science,University of Otago, P.O.Box 56, Dunedin, New Zealand

[4] K.K. Aggarwal et. al. 'Measurement of Software Maintainability Using a Fuzzy Model' Journal of Computer Sciences 1(4):538-542, 2005

[5] P. K. Suri1, Bharat Bhushan2 "Simulator for Software Maintainability" Kurukshetra University, Kurukshetra (Haryana) India IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.11, November 2007

[6] Markus Pizka and Florian Deißenböck ' How to effectively define and measure maintainability' in 2007

[7]Mehwish Riaz, Emilia Mendes, Ewan Tempero 'A Systematic Review of Software Maintainability Prediction and Metrics, New Zealand in2009978-1-4244-4841-8/09/$25.00 ©2009 IEEE

[8]Priyanka Dhankhar1, Harish Mittal2 'Software Maintainability In Object Oriented Software' in 2010 proc.conference 8th may 2010.

[9]Chikako van Koten Andrew Gray An Application of Bayesian Network for Predicting Object-Oriented Software Maintainability in March 2005 ISSN 1172-6024

[10]Berns, G., 1984 "Assessing Software Maintainability ."Communications of the ACM, 27: 14-23.

[11]Baker, A.L.et.al. and R.W.Witty, "A Philosophy for Software Measurement," Journal of Systems and Software, 12, 277-281 (2000).

[12]Wilde, N. and Ross Huitt: "Maintenance Support for Object- Oriented Programs," Proceedings of IEEE Conference on Software Maintenance Wilde, N. and Ross Huitt: "Maintenance Support for Object- Oriented Programs," Proceedings of IEEE Conference on Software Maintenance

[13]Booch, G., "Object Oriented Development," IEEE Transactions on Software Engineering, SE-12, 211-221, 1986.

[14]Halstead,Maurice H. "Elements of Software Science" Elsevier north Holland,New York,1997.

[15]R. K. Bandi et. al. "Predicting Maintenance Performance Using Object-Oriented Design Complexity Metrics", IEEE T Software Eng, 29, 1, Jan. 2003, pp. 77 – 87.

[16] Muthanna, S., K. Kontogiannis and B. Stacey, 2000. 'A maintainability model for industrial software systems using design level metrics.' Proc. Seventh Working Conf.

[17].Land R.,:Measurement of Software Maintainability", In Proceedings of Artes Graduate Student Conference, ARTES, 2002

[18] Chandershekhar Rajaraman Michael R. Lyu "Reliability and Maintainability related software metrics in C++" 2003.

[19]Alain April1 et. al. "Software Maintainence Maturity Model: The software maintainence process model" 2004

[20]McCabe Thomas j. "A Complexity Measure", IEEE Transaction Software Engineering,Vol2, December 1976.

[21]Roger Jang et al.,1995. Fuzzy Logic Toolbox for Matlab. The Math Works,USA.

[22]Shyam R. Chidamber and Chris F. Kemerer "A Metrics Suite for Object Oriented Design", June 1994.

[23]Biggerstaff, T., and C. Richter, "Reusability Framework, Assessment, and Directions," IEEE Software, March 2000,pp.41-44.

[24] Jane Huffman Hayes "A Metrics-Based Software Maintenance Effort Model**"** 2000.