BIOINFO Computational Mathematics
Volume 1, Issue 1, 2011, pp-01-04
Available online at: http://www.bioinfo.in/contents.php?id=321

Software Maintenance: An Analytical Approach

Gautam A. Kudale*,Jagdish A. Sangvikar*

**Vidya Pratishthan's Arts, Science and Commerce College, Baramati,MS,India*
gaukudale@yahoo.co.in, sangvikarjagdish@yahoo.co.in

Abstract:

Software maintenance is defined as modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment.[16] Software maintenance is an unavoidable activity in such changing world. This paper focuses on the different types of maintenance and their relative studies. At what extent and what percentage a particular maintenance is required is also focused.

Software development has become the most valuable productive asset to any business, industrial and government organization in the present time. Ever since there has been software there has been software maintenance [1]. No theory or practice indicates that will change. We examine why software needs "software maintenance". The demands that cause software change are requirements for functional change (enhancement), failures and errors (correction), technology change (adaption), user help and support new developments. The paper presented here also focuses on the need of software maintenance, types of software maintenance, amounts that are consumed for each type of software maintenance, software maintenance problems. Beyond these things, the steps that carry out the process of software maintenance and it's relevant diagram is also exhibited.

Keywords: software development life cycle, software maintenance, maintainability.

INTRODUCTION

Maintenance plays an important role in the life cycle of a software product. It is estimated that there are more than 100 billion lines of code in production in the world. As much as 80% of it is unstructured, patched and not well documented. Maintenance can simplify these problems. As the product becomes aged, it becomes more difficult to keep them updated with new user requirements. Maintenance needs developer's time, effort, and money. This requires that the maintenance phase be as efficient as it can. There are various steps in the software maintenance phase. [7] The majority of a software budget in large companies is devoted to maintaining systems. Near about 50-90 % of software cost is spent on maintenance. Maintenance is of four types.

Corrective maintenance deals with fixing bugs in the code. Adaptive maintenance deals with adapting the software to new environments. Perfective maintenance deals with updating the software according to changes in user requirements. Finally, preventive maintenance deals with updating documentation and making the software more maintainable. All changes to the system can be characterized by these four types of maintenance. Corrective maintenance is 'traditional maintenance' as 'software evolution.'

I. NEED OF SOFTWARE MAINTENANCE

- Changing business emergently needs the change to be reflected in the concerned software.
- Generally every system has a feature of evolution throughout its lifetime through a seamless process.
- The process is a spiral process involving requirements, design and implementation throughout the life time of the system.

II. SOFTWARE MAINTENANCE

Software maintenance is defined 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 [IEEE 1990]

III. TYPES OF SOFTWARE MAINTENANCE

- ***Corrective Maintenance*** is maintenance performed to correct faults in hardware or software [IEEE 1990]. Changes necessitated by actual errors (defects or residual "bugs") in a system are termed corrective maintenance. These defects manifest themselves when the system does not operate as it is designed or advertised to do. A defect or "bug" can result from design errors, logic errors and coding errors. Design errors occur when for example changes made to the software are incorrect, incomplete, wrongly communicated or the change request is misunderstood. Logic errors result from invalid tests and conclusions, incorrect implementations of design specification, faulty logic flow or incomplete test data. Coding errors are caused by incorrect implementation of detailed logic design and incorrect use of the source code logic. Defects are also caused by data processing errors and system performance errors. All these errors, sometimes called "residual errors" or "bugs" prevent the software from conforming to its agreed specifications. ***Corrective maintenance has been estimated to account for 20% of all maintenance activities.***

- ***Adaptive Maintenance*** is software maintenance performed to make a computer program usable in a changed environment [IEEE 1990].

Any effort that is initiated as a result of changes in the environment in which a software system must operate is termed adaptive change.

Adaptive change is a change driven by the need to accommodate modifications in the environment of the software system, without which the system would become increasingly less useful until it became obsolete. The term environment in this context to all the conditions and influences which act from outside upon the system, for example business rules, government policies, work patterns, software and hardware operating platforms. A change to the whole or part of this environment will warrant a corresponding modification of the software. ***This task is estimated to consume about 25% of the total maintenance activity.***

- ***Perfective Maintenance*** is software maintenance performed to improve the performance, maintainability, or other attributes of a computer program. [IEEE 1990]. This is the most common type of maintenance encompassing enhancements both to the function and the efficiency of the code and includes all changes, insertions, deletions, modifications,extensions,andenhancements made to a system to meet the evolving and/or expanding needs of the user. A successful piece of software tends to be subjected to a succession of changes resulting in an increase in its requirements. This is based on the premise that as the software becomes useful, the users tend to experiment with new cases beyond the scope for which it was initially developed. Expansion in requirements can take the form of enhancements of existing system functionality or improvement in computational efficiency. ***Perfective maintenance is by far largest consumer of maintenance resources estimates of around 50% are not uncommon.***

- ***Preventative Maintenance*** is maintenance preformed for the purpose of preventing problems before they occur [IEEE 1990]. The long-termed effect of corrective, adaptive and perfective change is expressed in Lehman's law of increasing entropy:

As a large program is continuously changes its complexity, which reflects deteriorating structure, increases unless work is done to maintain or reduce it (Lehman 1985).

The IEEE defined preventative maintenance as *"maintenance performed for the purpose of preventing problems before they occur"* (IEEE 1219 1993). This is the process of changing software to improve its future maintainability or to provide a better basis for future enhancements.

The preventive change is usually initiated from within the maintenance organization with the intention of making programs easier to understand and hence facilitate future maintenance work. Preventive change does not usually give rise to a substantial increase in the baseline functionality.

Preventive maintenance is rare (only about 5%) the reason being that other pressure tend to push it to the end of the queue.

IV. SOFTWARE MAINTENANCE PROBLEMS

The Complexity is the root problem of software maintenance. Sometimes complexity arises because a system is migrated form hardware to software in order to gain the additional functionality found in software. The combination of scale and application complexity means it is infeasible for one person alone to understand the complete software system. Most computer systems are difficult and expensive to maintain. Software changes are poorly designed and implemented. The repair and enhancement of software often injects new bugs that must later be repaired.

V. PROBLEMS RELATED TO SOFTWARE MAINTENANCE

Software Maintenance Process

Seven –step approach:

Step 1 – Change Management

Step 2 – Impact Analysis

Step 3 – System Release Planning

Step 4 – Design the changes

Step 5 - Code the changes

Step 6 - Test the changes

Step 7 – System release

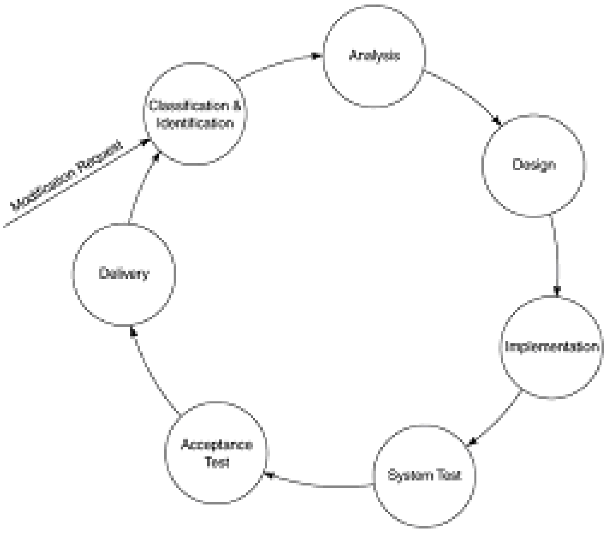


Figure-1: The IEEE Maintenance Process Activities diagram

The nature of software change :

We examine why software needs "software maintenance". What this phrase means. Why it is misleading. The demands that cause software change are:

- Requirements for functional change ("enhancement")
- Failures and errors ("correction")
- Technology changes ("adaptation")
- User help and support ("support")
- New developments.

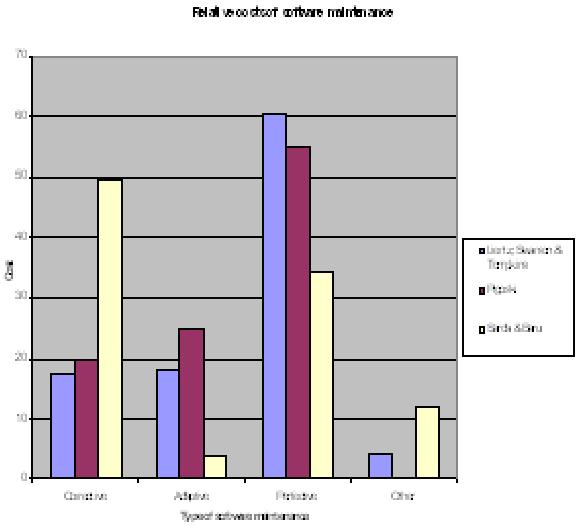
We examine the respective contribution of these demands to the total software work.

I. RELATIVE COSTS OF SOFTWARE MAINTENANCE

According to the analysis proceeded by interpretation by different experts…

Type of maintenance	**Lientz, Swanson & Tompkins**	**Pigoski**	**Sarda & Baru**
Corrective	17.4	20	49.7
Adaptive	18.2	25	4
Perfective	60.3	55	34.4
Other (like preventive)	4.1	0	11.9

Table – 1 Showing the cost consumed by each type of maintenance.



Graph 1 – Showing the relative cost of software maintenance.

The said survey was made on the software's of 120 organizations in terms of the above mentioned types.

I. CONCLUSION

Traditionally, 50 to 90% cost of the total software development cost are spent on the software maintenance phase.

Form the graph and the table, and the survey made by the three scientists shown above, it is clear that most of the maintenance expenses are spent in Perfective Maintenance. To sustain the maintainability in such competitive world one has to spend the money as stated for the Perfective maintenance.

ACKNOWLEDGMENT

The author thanks to **Dr. Arun Adsool**, Principal, Vidya Pratishthan's Arts, Science and Commerce College, Vidyanagari, Baramati for his valuable guidance and encouragement for the research work.

II. REFERENCE

1] Nicholas Zvegintzov, Girish Parikh, "*Sixty Years of Software Maintenance: Lessons Learned*," icsm, pp.726-727, 21st IEEE International Conference on Software Maintenance (ICSM'05), 2005

2] *Software Maintenance Research in the Mire?* Frank Niessink Division of Mathematics and Computer Science, Faculty of Sciences, Vrije Universiteit. Annual Workshop on Empirical Studies of Software Maintenance (WESS'99), September 3-4, 1999, Oxford, UK.

3] "Software Maintenance", Gerardo Canfora and Aniello Cimitile , University of Sannio, Faculty of Engineering at Benevento Palazzo Bosco Lucarelli, Piazza Roma 82100, Benevento Italy. 29 nov, 2000

4] Maintenability : Theory and Practice Gary O' Neill Georgia Tech Research Institute Logistics and Maintenance Applied Research Center.

5] Software maintenance from a Service Perspective, FRANK NIESSINK and HANS VAN VLIET, Faculty of Science, division of Mathematics and Computer Science, Vrije Universiteit, De Boeleallan 1081a,1081HV Amsterdam, the Netherlands.

6] Software maintenance - an overview, Carl Allen CEng MBCS CITP.

7] Software maintenance As Part of the Software Life Cycle, Comp 180: Software Engineering Prof. Stafford, Dept. of Computer Science Tufts University, December 16, 2003.

8] Neural Based Model to Predict Corrective Maintenance Time, Yogesh Singh, Bindu Goel, Pravin Chandra, Journal of CSI, Vol. 37, No. 2, April – June 2007.

9] A step towards software preventive maintenance, Yogesh Singh, Bindu Goel, ACM SIGSOFT Software Engineering Notes, Volume 32, Num,ber 4, July 2007.

10] Evaluation of Various training algorithms in a neural network model for software engineering application, K.K. Aggrawal et all. ACM SIGSOFT Software Engineering Notes, Volume 30, Num,ber 4, July 2005.

11] Software Maintenance As Part of the Software Life Cycle Comp180: Software Engineering Prof. Stafford Department of Computer Science Tufts University Kagan Erdil December 16, 2003.

12] Types of software evolution and software maintenance Ned Chapin1,_,†, Joanne E. Hale2, Khaled Md. Khan3, Juan F. Ramil4 and Wui-Gee Tan5

13] Roger S. Pressman, Software Engineering: A Practitioner's Approach, McGraw-Hill Higher Education

14] Software Maintenance, Effective Practices for Geographically Distributed Environments, Gopalaswamy Ramesh , Ramesh Bhatti Prolu, TATA MC GRAW HILL.

15] System Analysis, Design, An Introduction to Software Engineering (SADSE), Prof. S. Parthasarathy, Prof. B.W. Khalkar, Everest Publishing house.

16] IEEE Standard for Software Maintenance, IEEE Std 1219-1998 (Revision of IEEE Std 1219-1992).