International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 03 Issue: 08 | Aug-2016 www.irjet.net p-ISSN: 2395-0072

Implementation Of Software Management and Maintenance

Sona V

Lecturer in Computer Engineering,
Swami Nithyananda Polytechnic college, Kanhangad, Kasaragod Dt., Kerala

---

Abstract

Software development and maintenance require a great deal of expertise. The application area, the problem to be solved, the software process employed, the technical aspects of the programming language(s), the system's architecture and how the many elements work together, how the system interacts with its environment, etc. are all important. Getting your hands on all of this information is time consuming and expensive. Although difficult to document and archive, it is usually confined to the minds of the developers who worked on a particular software project. Even in the case of new software development, this might be a difficulty when it comes to finding lost information of an abstract type from legacy source code amongst a swarm of unrelated data. The most difficult and expensive aspect of software development is software maintenance. In contrast to the fun and challenging work of software creation, software maintenance chores are significantly less fulfilling for software developers. Knowledge about software maintenance appears to be significantly less common in academic literature than that on software development.

Keyword

Program comprehension, software maintenance, software evolution.

Introduction.

In order to remain relevant in the market, software systems must be adaptable. Successful software maintenance results in a new release that is an improved version of the prior one, which is the only way to implement change. There's a term for this: software evolution. It is essential for software maintainers to become familiar with the specific aspects of the software, such as source code and documentation, in order to improve the quality of the product. The lack of documentation forces maintainers to rely on source code. [1] In order to successfully make changes, it is vital to have a deep understanding of the source code.

The overall maintenance measures for important information will now be taken when computer software development approaches, and additional maintenance measures will be done for product information and development versions as the development stage is entering. In order to successfully control software development costs and to reduce software development effort, software designers must integrate the essential development and management practises with the original characteristics. [2] For decades, computer software developers have focused on how to create an effective information management system for their projects.

In order to get the best results, productivity needs to be tied to a maintenance management system. A company's productivity is determined by how far the inputs can get into the system, as determined by the organisation [3]. An obvious correlation exists between productivity improvement and maintenance management. Increased productivity can be achieved by proper maintenance management, which improves the performance and availability of machines. Quality and efficiency are no longer confined to the availability of a product, but are also important variables in determining its efficacy. Due to its role in ensuring and improving machine availability, performance efficiency, product quality and speedy delivery, environmental and safety requirements, maintenance management has had a significant impact on business performance metrics such as productivity and profitability in recent years.

Implementation of System Management Flow

Business requirements information is managed by creating standard functions and status fields in order to implement the system's information management flow. The prior data information has been locked and cannot be amended based on an examination of the specific dependence relationship between business requirements and the effectiveness treatment of business requirements information. [4] When defining technical requirements in the implementation stage of technical requirements information management flow, the function shall be activated, and the status of matching business needs

shall be updated. The corresponding business needs will be automatically detected and the order of priority modified during the loading stage of the technical requirements.

Importance of software maintenance

A new piece of software is always thrilling for a corporation when it's released into the world for the first time. This includes the actual coding and constructing, as well as licencing options, marketing, and more. In order to remain relevant in an ever software landscape, excellent software must be flexible. This entails keeping close tabs on things and doing routine maintenance on them. Software must keep up with market changes and needs as technology evolves at the speed of light. [5]

Review of Literature

Kitchenham et al. [6] had a similar focus, although they didn't use the Experience Factory. They created an ontology to organise their maintenance-related research. Despite their pioneer effort, they do not intend to solve any maintenance issues; rather, they hope to contribute to maintenance research.

A third method is that of Deridder [7], which seeks to record and retrieve knowledge through the use of specific tools. Maintainability would be improved by a tool that preserves explicit knowledge about the application domain (in the form of concepts and relations between them) and links between these concepts and their implementation. His method focuses solely on the tool and lacks an underlying ontology to back it up. It focuses on application domain knowledge from the start (although the tool would probably allow representing any concept in other domains).

Based on questions about the impact of maintenance efforts on software products, Chapin [8] developed an evidence-based taxonomy that extends Swanson's original typology. Training, consulting, evaluative, reformative, up-to-the-minute grooming and preventative maintenance are among the 12 categories of software maintenance included in this extended maintenance typology. It was acknowledged in the software maintenance ontology proposed by Kitchenham et al. [9] that support activities (supporting maintenance) such as non-technical help desks, support for speedy solutions to technical problems, and training are also part of software maintenance. An organization's technical and organisational situation dictates what type of maintenance duties it should use in industrial practice.

Hasan et al. [10] emphasised the importance of small software businesses in various nations, including the United States, China, Brazil, and Europe. It has been determined that organisations with fewer than 25 people are classified as "Very Small Entities" (VSEs).

To better understand how a small software company that builds and maintains a university information system conducts maintenance, Hasan et al. [11] carried out an interpretive case study. Maintenance procedures were found to be dependent on informal, ad-hoc processes rather than a formal process model, according to the findings of the case study.

Objectives

1. Analyze the current system of maintenance
2. Explore and identify the issues in the maintenance management system
3. To evaluate the inventory control mechanism for maintenance supplies.
4. Create a preventative maintenance plan for high-priority equipment and equipment

Research Methodology

The term "Research Methodology" refers to the study of how a study's methodologies were selected and implemented. Theoretical notions are also included in this discussion, which provide further information concerning the selection and application of procedures. In the current study, secondary data was acquired from a number of sources, including books, educational and development periodicals and government publications as well as print and online reference materials.

Result and Discussion

Maintaining software is a big concern for programmers. Software quality can't be maintained without a well-designed maintenance mechanism. A variety of process models for software maintenance have been proposed by a number of authors. [12]

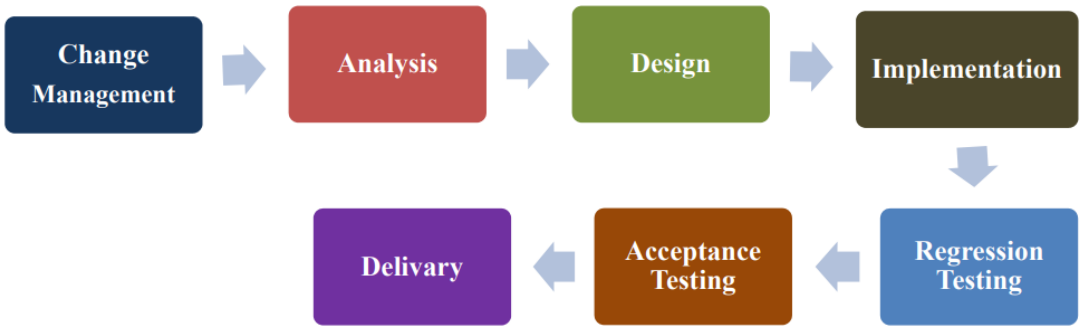

Fig. 1 Maintenance Phase Process

These models help to organise maintenance into a sequence of linked tasks, or phases, and determine the order in which these phases are to be carried out. ” Consequently, there are essentially seven primary steps in the maintenance process, as follows:

Figure 2 depicts a typical workload allocated among the most common forms of software maintenance during the operational usage phase after delivery of the software product. Most bugs are discovered after the software has been released, and they are fixed as quickly as feasible. [13] When the programme is in operation, errors are less likely to be discovered, resulting in a declining curve for corrective maintenance (red curve in fig 2).

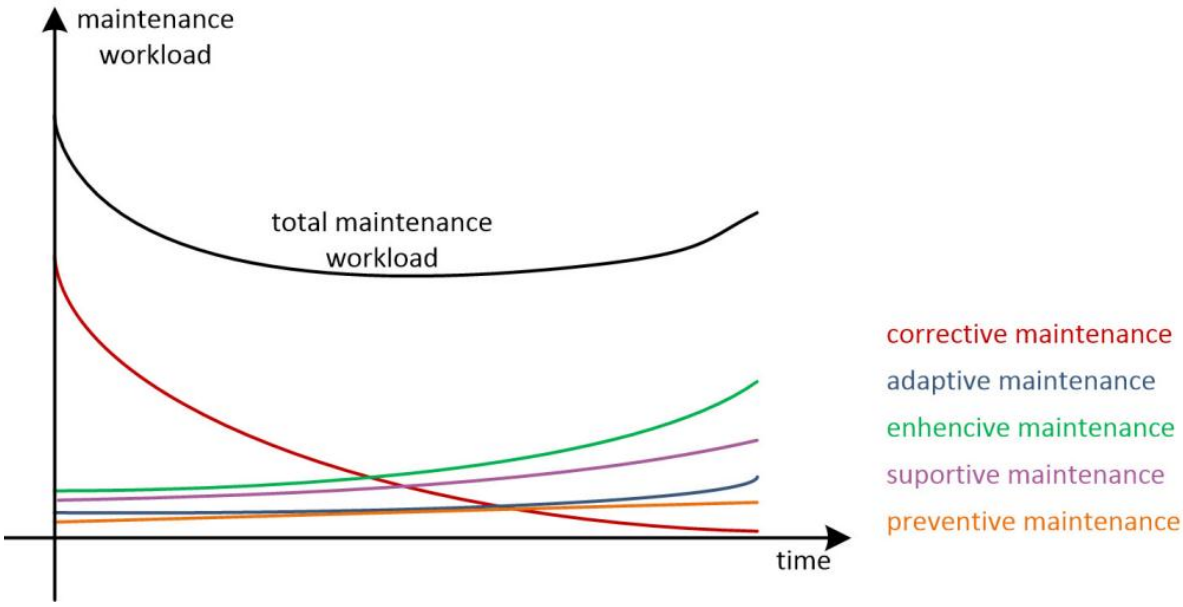

Fig. 2: Typical workload following software delivery for maintenance

Improvements, enhancements, or adaptations to the software that is now being utilised are the result of an increase in other sorts of software maintenance jobs (other coloured lines in fig 2). Program improvement (fig. 2's green curve) and user assistance take centre stage when more and more bugs are found and fixed during the course of software use (purple curve in fig 2). [14]

In order to have a better understanding of the Software Maintenance Lifecycle (SMLC), an abstract representation of each activity is created. Because software maintenance is a relatively new discipline compared to software creation, a limited number of process and lifecycle models have been developed to take maintenance concerns into account [15].

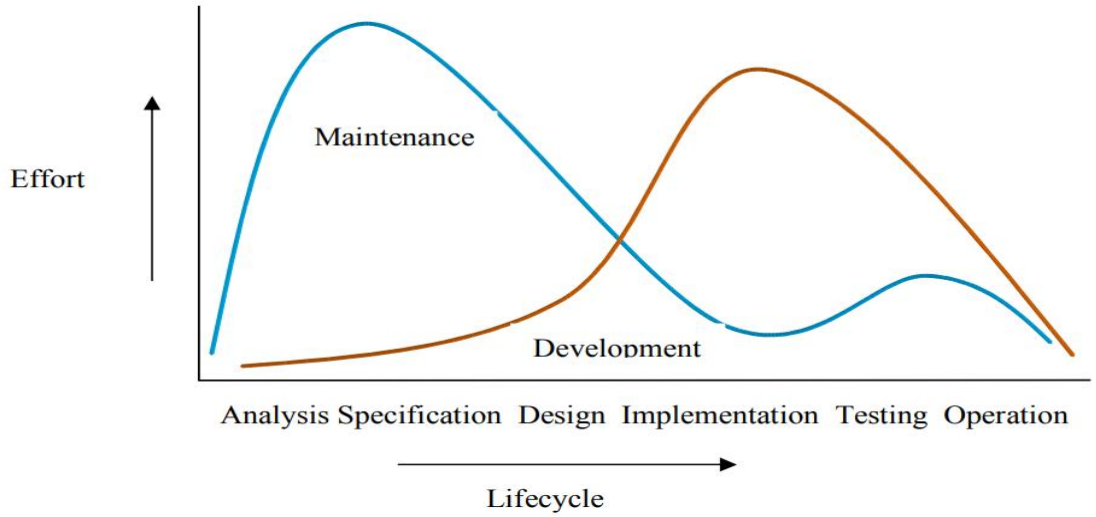

Fig. 3 shows the effort required at various points in the maintenance and development lifecycle.

In contrast to software development lifecycle models, the earliest phases of a maintenance model demand significantly more effort than later ones. There are a lot of differences between traditional development approaches and the maintenance model when it comes to future software evolution. [16]

The workflow system is crucial to the operation of any maintenance organisation. In order to successfully complete maintenance tasks, a well-defined workflow system must be developed and implemented.

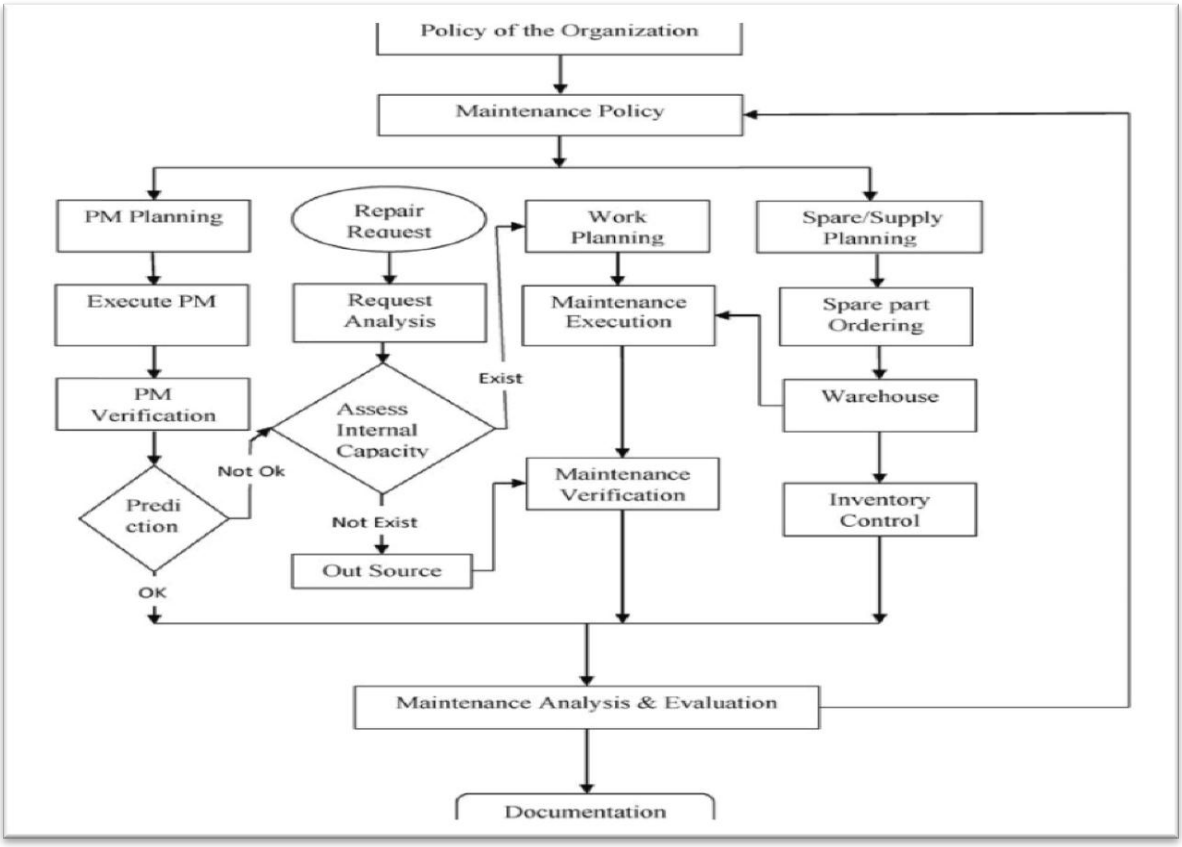

Fig. 4 is a diagram of the recommended maintenance work flow.

Maintenance is made easier, and productivity is increased as a result. As a result, in Fig. 4, the maintenance planner's work flow system for the case company is shown.

Conclusion

Software maintenance is a highly specialised field that requires a great deal of expertise. Knowledge of a legacy software's application domain, the problem it solves (and how it meets those criteria), and how it interacts with its environment are essential to the job of software maintenance. From the experience of the maintainers, the user's knowledge, documentation, source code, and so on. Most of the time, however, gained information is not explicitly documented so that it may be retrieved and reused in the future. Knowledge of the different systems a maintainer has worked on is lost when this person leaves the company.

References

1. Song H Y, Yao H Z. Training Center Management information system development based on maticsoft.NET[J]. Information Technology, 2014.

2. Cheng X, Qiu X X. Design and Implementation of a Software Automation Development Framework for Management Information System[J]. Advanced Materials Research, 2014, 989-994(4):4488-4492.

3. Sun K, Liu J, Yan X. Design and Development Experience of Enhanced Environment Laboratory Information Management System[J]. Journal of Environmental Management College of China, 2015

4. A. Chammas, M. Traore, E. Duviella, M. Sayed-Mouchaweh, S. Lecoeuche Condition monitoring architecture for maintenance of dynamical systems with unknown failure modes IFAC Proceedings Volumes, Volume 45, Issue 31, 2012, pp. 36-41

5. Małgorzata Jasiulewicz-Kaczmarek The role and contribution of maintenance in sustainable manufacturing IFAC Proceedings Volumes, Volume 46, Issue 9, 2013, pp. 1146-1151

6. B.A. Kitchenham, G.H. Travassos, A. von Mayrhauser, F. Niessink, N.F. Schneidewind, J. Singer, S. Takada, R. Vehvilainen, H. Yang,
Towards an ontology of software maintenance, J. Softw. Maint.: Res. Pract. 11 (1999) 365–389

7. Dirk Deridder, Facilitating software maintenance and reuse activities with a concept-oriented approach, Technical report, Programming Technology Lab – Vrije Universiteit Brussel, May 2002

8. Chapin N 2000 Proceedings of the International Conference on Software Maintenance ICSM '00 (San Jose, CA, USA) pp 247– doi: 10.1109/ICSM.2000.883056

9. Kitchenham B A, Travassos G H, von Mayrhauser A, Niessink F, Schneidewind N F, Singer J, Takada S,Vehvilainen R and Yang H 1999 Journal of Software Maintenance: Research and Practice 11 365–389

10. Hasan R, Chakraborty S and Dehlinger J 2012 Software Engineering Research,Management and Applications 2011 (Studies in Computational Intelligence vol 377) ed Lee R (Springer, Berlin, Heidelberg) pp 129–143 doi: 10.1007/978-3-642-23202-2 9

11. April A., Hayes J. H., Abran A., Dumke R.. Software Mainentance Maturity Model (SMmm): the software maintenance process model. Journal of software maintenance and evolution: research and practise, pp.197-233.

12. Coleman, D., Ash, D., Lowther, B. and Oman, P. Using Metrics to Evaluate Software System Maintainability. IEEE Computer, August, pp.44-49.

13. Abran A., Moore J.W. Guide to the software body of knowledge (SWEBOK). Ironman version. IEEE Computer Society Press: Los Alamitos CA,pp. 6-1-6-15.

14. Tiberiu Dobrescu, Nicoleta-Elisabeta Pascu, Gabriel Jiga, Constantin Opran Optimization Criteria of Plane Lapping Machines Procedia Engineering, Volume 100, 2015, pp. 428-434

15. J. Chabriais, B. Gibaud DICOM, le standard pour l'imagerie médicale Traitement De L'image, 2013, pp. 99-150

16. S. Selvi, G. D. Maheshwari, S. Khan, A. K. Gupta, B. N. Anand and A. K. Biswal, "Design, development and implementation of maintenance management software for RDCIS, SAIL," 2013 4th International Conference on Computer and Communication Technology (ICCCT), 2013, pp. 252-257, doi: 10.1109/ICCCT.2013.6749636.