Analysis Of Software Maintenance Cost Affecting Factors And Estimation Models

Chamkaur Singh, Neeraj Sharma, Narender Kumar

Abstract: Software maintenance is a very broad activity that includes improvements in capabilities, error correction, optimization, and removal of obsolete capabilities. Normal operate of basic work and accurate cost estimation is to ensure the normal maintenance of necessary software tools. Due to this various researchers have been attracted towards the research on various factors affective the productivity of software development. The motive behind the identification of those factors that affect productivity helps management to take steps for removing it and saving the maintenance cost of the software. The objective behind this work is to analyze the various activities in accordance with the current software development environment and understanding the key factors that affect the maintenance cost of the software. Along with types of software maintenance estimation, various software maintenance activities are classified into three categories. The other section contains the review on work done by various researchers on the maintenance cost of software and various factors that affects it. These factors are classified into technical and non-technical factors.

Index Terms: Software maintenance, Maintenance cost of software, Software development environment, Maintenance activities, Technical factors, Non-technical factors, Cost estimation Models.

1 INTRODUCTION

Software maintenance is a very broad activity that includes improvements in capabilities, error correction, optimization, and removal of obsolete capabilities. There is the need to develop a mechanism for evaluating, controlling and making changes due to the predictable nature of change [1]. So during usage of software, if work is done to amend it then it is considered as maintenance work. Over the period the new technologies are used to maintain the value of software. In this meeting additional requirements are expanded to make it more efficient. While developing software there is a need to prepare an appropriate plan for maintenance activity that is considered as important aspect of software maintenance, in which modifications are to be done is specified by their plan. Due to a change in any requirement cost should be included to develop the budget of software [2]. This means maintenance cost will increase, not only due to poor design but change in customer environmental and expectations needs in which system has been developed. Further software maintenance is a plan in which scope of maintenance, maintenance team or person and cost estimation of software maintenance is included in it. After delivery software can be modify comes under maintenance in which the faults are corrected and improves the performance or other attributes. Making products to adapt change in the environment also comes under it.

- *Chamkaur Singh , Research scholar of I.K. Gujral Punjab Technical University, Jalandhar, Punjab, India. E-mail: dhillon.chamkur@gmail.com*
- *Dr. Neeraj Sharma ,Professor, Gian Jyoti Institue of Management & Technology , Mohali. E-mail: nrjsharma@yahoo.com*
- *Dr. Narender Kumar, Assistant Professor, HNB Garhwal University, Srinagar Garhwal Uttarakhand, India. E-mail: narenrawal@gmail.com*

There is a very large workload of software maintenance even if there is variation in the costs of maintenance for different application. Even though for large software costs of maintenance is four times larger than the costs of software development. For maintaining the existing software around 60% of the manpower will be used by foreign countries software development organizations. This percentage is still getting an increase day by day in the number of software tools and manpower. Most of the time users and developers face the problem by software maintenance [3]. In understanding the software maintenance, prior arrangement and cost by developers and users played an important role in accurate estimation of software maintenance costs. In this paper, we have given a review on maintenance cost software estimation. In the second section of paper, a brief detail about maintenance cost estimation models is given that contains Phase, release and task level maintenance estimation models. The third section contains the type of software maintenance in which we have given an explanation of five activities related to the maintenance cost of the software. The fourth section contains a review on work done by various researchers in the field of maintenance cost. Before concluded a list of various factors that affect the maintenance cost of software is given by dividing it into technical and non-technical factors.

2 MAINTENANCE COST ESTIMATION

MODELS

As compared to new development less attention is received by area of software maintenance estimation. To estimate the maintenance costs various models are introduced and applied by giving importance of software maintenance [4]. Various sets of software maintenance work are addressed by these models that cover for instance, error rectifications, functional enhancements, technical renovations and re-engineering. On the basis of granularity level of estimation focus it is classified into three types:

- Phase maintenance estimation models
- Release maintenance estimation models
- Task level maintenance estimation models

2.1 PHASE-LEVEL MODELS

The effort of routine maintenance work for either whole phase or certain period of software maintenance is focused by set of maintenance methods. After it is delivered all activities performed during the operation of a software system refers to routine maintenance work. In this, technical improvement, minor functional changes, fault corrections and enhancement are involved in which main purpose is regular operation of the system. In this kind SLIM, SEER-SEM, KnowledgePlan, PRICE-S, and COCOMO is integrated into maintenance models. When developed an estimating cost of a new system then produce a maintenance cost is usually a part of the estimates [5]. To estimate maintenance effort key input is the size of the system and additional cost drivers are used in these models that are specific to software maintenance. The level of the unfamiliarity of the programmer (UNFM) and software understanding (SU) are two drivers used by COCOMO for its sizing the maintenance work. In its maintenance cost calculations, such parameters are used as maintenance rigor and maintenance size growth over time in SEER-SEM. For architectures of making investment and trade-off analysis decisions on the system a maintenance cost for a system is estimating the development cost. Due to number of assumptions made about the system an accurate estimation of cost of system maintenance becomes difficult. In order to compare and evaluate the estimation accuracies of these models there is no empirical study done by any researchers. Another possible reason is that these models, with the exception of COCOMO, are proprietary and their details have not been fully published, making them difficult to be investigated in the research context.Methods are provided by SLIM, SEER-SEM, KnowldgePlan, COCOMO and PROCE-S to resize the work and estimate the reuse and adaptation work to compute the schedule and effort. For estimating the new software development the same models are developed by computing the schedule and effort. The characteristics of new software development and reuse and adaptation work are the same in these models. Unfortunately, this assumption has never been validated empirically.

2.2 RELEASE-LEVEL MODELS

At finger grained level groups of models focuses on the maintenance cost instead of estimating the cost of the maintenance that estimate the effort of a planned release and planned set of maintenance tasks. This approach usually takes data from the past releases and analyzing the changes to estimate the cost for the next release.The simple regression model is described by Basili et al., along with the characterizing the effort distribution of maintenance releases to estimate the effort for different types such as error correction and enhancement. SLOC, single variable is used by model that was measured as sum of deleted, modified and added SLOC that also includes blanks and comments. The prediction accuracy was not reported although the coefficient of determination was relatively high (R2 = 0.75), indicating that SLOC is a good predictor of the maintenance effort.After the initial delivery of the system, a maintenance work is considered as being organized into sequences of operational releases and linear regression models are evaluated and introduced by Lehman and Ramil to estimate the required effort to evolve the system from a release to the next. All necessary maintenance tasks to grow the system are taken into account by their models that include functional enhancements, error corrections, and technical improvements. The size metrics, subsystems, and the number of changes to modules plus all changed modules are measured at coarse granularity levels are predicted by the model.

2.3 TASK-LEVEL MODELS

The cost of implementing each maintenance task is estimated by task level model that comes in the form of change requests and error reports. Small effort estimates are mainly deals with this type of model that usually range from a few hours to a month. SoftCalc toll and seven-step process are introduced by sneed that helps in estimating the required cost and size to implement maintenance tasks [Sneed 1995]. Various size measures are used by this model:

- SLOC: Physical lines of code and statements
- Function Points
- Object Points
- Data points

On the basis of quality, complexity and project influence factors a proportion of the affected software was determined the size of the impact domain. The productivity index and adjusted size are used to compute the maintenance effort. Classification of maintenance change requests is beneficial as compared to generating an exact effort estimate in terms of levels of difficulty or level of required effort. Briand and Basili proposed a modeling approach to building classification models for the maintenance effort of change requests. Four level steps are involved in modeling procedures:

- Identifying predictable metrics
- Identifying significant predictable metrics
- Generating a classification function
- Validating the model

Each predictable effort and the variable range are divided into intervals and difficulty index number is used to represent it. The effort range has five intervals (below one hour, between one hour and one day, between one day and one week, between one week and one month, above one month) which were indexed, from 1 to 5 respectively. Then from four different projects at the NASA Goddard Space flight center, 163 change requests dataset was used by Briand and Basili in order to evaluate the proposed approach. The approach produced the classification models with classification correctness from 74% to 93%.

3 THE TYPES OF SOFTWARE MAINTENANCE

According to International Electro-technical commission activities, there are mainly five categories of software maintenance according to the needs of software maintenance. These are maintenance repair, Preventive maintenance, Integrity of maintenance, adaptability of maintenance and evolution of maintenance. These five software maintenance activities are merge by software maintenance activities [7]. As the integrity of maintenance contain improvement, evolution and correction of maintenance that contains repair of maintenance and preventive maintenance. In this section of the paper, we

have given a brief detail about these three categories in which all the software maintenance activities are divided. The integrity of Maintenance: It expands the functionality, improves performance and does needful expansion and changes to meet the change in needs of users. To improve the performance enhancement and changes done to make the ease of maintenance are the main contents. Adaptability Maintenance: To do changes in software so that it can adapt the changes comes by operating environment. The impact of the system of rules, hardware configuration, structure of text volume, rules and laws, data format, system software are main changes included in it. Correction of maintenance: The errors can't be find out by inspection and testing that comes by the maintenance of the system running in the development process. This includes coding errors, data errors, design errors, and documentation errors, etc. According to stats taken from abroad [2], there is 54 to 72% of integrity maintenance, from total maintenance of activities there is 18 to 25% of adaptive maintenance and only 17 to 21% of correction activity. If errors are not find out by acceptance and testing early in software applications that result in more maintenance workload with fewer errors. On the other hand when the software is running then an error will be less and less need for maintenance workload to correct it. Adaptive maintenance of software applications early in less time with the application of the growth of the workload will increase changes will not be particularly significant. There is rapid growth in workload by integrity to maintain the growth over time.

4 RELATED WORK

Various researchers have attracted towards the research on various factors affects the productivity of software development. The motive behind the identification of those factors that affect productivity helps management to take steps for removing the negative effects. R. D. Banker, et.al, (2000), have used the frontier estimation method to explore the possibility of developing an estimated model of software development [8]. In their approach, they have used the output metrics of project quality metrics and the entire project life cycle. Before moving towards it they did a review on various factors affecting the productivity of software maintenance. From empirical analysis of pilot data it has been concluded that by high project quality it is not necessary that project productivity will get reduced. Various factors that give positive change in productivity are:

- The capability of the project team
- Good system response time

On the other hand, factors that put negative effects on productivity are:

- Lack of team application experience
- High project staff loading

They have seen that short term productivity gets reduced using new structured design methodology and analysis. By analyzing the software architecture critical success factors along with their benefits and cost drivers pitfalls and potential of software architecture investments perspective can be gained. By identifying and exploiting an appropriate positive cost benefit relationships a potential investment of software architecture can be found. According to B. Boehm, et.al, (2007), pitfalls come from neglecting critical success factors or from making unrealistic assumptions about their associated cost drivers [9]. Majority of software total cost can be estimated by considering the software maintenance that is considered as an important software engineering activity. By understanding the factors that influence the cost of software maintenance tasks helps maintainers to take decisions about their work. The student's programmers controlled experiment of performing maintenance tasks on C++ program has been described by V. Nguyen, et.al, (2010) [10]. The main objective of them was to assess the maintenance effort, size, and three different maintenance types of efforts. Another main objective was to predict the programmer's effort spent on maintenance tasks to describe the estimation models. To perform their objectives they have performed an experiment by the participation of around 23 graduate students and senior majoring in computer science. All of them have to perform the tasks of maintenance required for one of the three task groups. Then they have divided the collected data into four estimation models to check the impact of different LOC on maintenance effort. From the experiment, they have predicted that there is very less productivity of corrective maintenance as compared to reductive and enhancive maintenance and there is need of very less around 50% of the total effort in programming comprehensive activities. And the best software effort model is able to estimate the time of around 79% of the programmers with 30% or less error. Normal operation of basic work is ensured by software maintenance that gives accurate cost estimation and ensures the normal maintenance of the necessary software tools. In Y. Ren, et.al., (2011), the article they have described the various types of software maintenance and analyse the impact of technical and non-technical factors on software maintenance costs [11]. They have also given the basic cost estimation model for software maintenance and improvement of cost estimation models and from it; they have listed the cost factors that affect the workload of maintaining weight. To estimate the cost of basic maintenance the annual changes in the development project costs are multiplied with the basic cost estimation model. The accuracy of software maintenance cost gets improved by appropriate adjustment of software maintenance workload weight factors. For software maintenance cost estimation users and software developers, a method was proposed in their article In software engineering effort estimation of software development is an important sub-discipline that has to gain the interest of various researchers. By the introduction of component-based software development (CBSD) a software development turned into engineering. In the place of traditional software development paradigms use of CBSD gives better results along with that it comes with various challenges to software effort estimation. The effort estimation of CBSD has become an important area of research due an increase in its use. So, T. Wijayasiriwardhane, et.al, (2011), did a review on work done by various researchers on predicting the effort of CBSD [5]. They have surveyed the approaches by considering various things like the types of estimation provided by researchers, level of acceptability, type of data required for their use and life cycle activities covered with regards to any validation. The main motive by them to give

a better understanding of schedule and cost of CBSD estimation approaches. Then P. Marounek, (2012), has introduced a software maintenance and support from the perspective of effort estimation [12]. The set of activities need for cost effective support of IT solution is software maintenance and mainly focus is given to best practices and delivery strategies and delivery process that cover in effort estimation. One of the researchers has concluded that in most of the cases estimating models are not used for estimation work due to its complexity and instead of it a pure expert estimate is used. So, P. Marounek, has introduced an extending PERT formula-based approach for measuring the effort estimation in software maintenance. The PERT formula is about the quality of historical experience and quality of estimator. To verify both supporting mortgage IS formulas a sub-competence center is used that gives better results as compared to PERT estimate. The results achieved are 98.8 and 91.8% against pure PERT 90.1%. From the last few years, great success is achieved by the software industry and production and maintenance are two phases of the entire life of the software There very much increase in the cost of software maintenance and around 90% of software life cost is related to its maintenance phase. The extraction and consideration of factors affecting the software maintenance cost help in estimating the cost and its reduction by controlling those factors. S. M. H. Dehaghani, et.al, (2013), have determined the various factors affecting the software maintenance cost and then ranked them on the basis of their priority and also presented the ways to reduce the maintenance cost [13]. For experiment purpose, they have selected 40 members from medical software maintenance team and determined various factors affecting maintenance cost by taking their interview. From it, they have obtained the 32 factors that are further classified into six groups from the projects that have the highest priority in maintenance cost. They have concluded that longevity of software can be increased and reduce the maintenance cost by considering the major elements like full documentation, careful feasibility of IT projects and accompany the designers in the maintenance phase. In order to predict the reliability, project planning, adaptability, productivity improvement and controlling of the software the cost estimation phase is important. And Functional point, SLIM, COCOMO, etc are various traditional software models to estimate the maintenance cost but no work is done on estimating the maintenance cost using fourth-generation language environment. So, M. Islam, et.al, (2014), have proposed a systematic approach and development for software maintenance cost estimation in which they have used fourth-generation language environment on the basis of COCOMO II [14]. Annual change traffic, SMCE with fourth-generation language environment, technical and non-technical factors are three parameters based on the model that affect the maintenance cost. By implementation of model, favourable results can be achieved.

5 VARIOUS FACTORS AFFECTING MAINTENANCE COST OF SOFTWARE

The maintenance cost of software can be affected by various factors. In this section, we have given some of the technical and non-technical factors that affect it [15].

5.1 Technical factors:

- Software complexity: There is a need for more time to modify or read a program if there is a more complicated structure of software that results in a significant increase in maintenance workload.
- Development of Human capacity: There is a reduction in maintenance workload by full analysis done by system analyst. If programmers have a good style of programming then an easy to understand or read software can be program by them. A significant reduction is existed in software errors when testing is done using advanced methods.
- Document Quality: Make documents for software maintenance support in which a detail description of the software system is given by maintenance personnel that gives help to other people to get familiar with the system. This also helps further in identification and correction of errors and improving the system that meets the changes in requirements of users or adapts changes in the system environment. There is a reduction in the cost of maintenance if clear, complete and concise documentation is available for support.
- Configuration management technology: This includes configuration identification, change control, status reports, the configuration of the program, version management, configuration auditing. The maintenance costs get reduced by effective control of software configuration management technology that strengthens and maintain the management of the maintenance work.
- Modern Programming Specifications: Follow the methods of software engineering principles and system engineering [16]. And use the modern programming practices that include variable naming, preparation of single entry and exit modules, documentation and use structured development methods to reduce coupling and improve cohesion. The cost of maintenance workload will get significantly increased if the software is a unified programming specification.
- Database Size: To maintain a database there is lot of work done by software maintenance and maintenance workload as it get affected by database size. There is a significant increase in maintenance workload by an increase in data or re-organizing the database and reconstruction of it.
- Component Reusability: This means the amount of developed code can be used in the system and defines the extent of code reusability. Different weight values are given to it according to its usability in work.
- Component Performance: In the case of component-based software their performance depends on the performance of individual software. There should be a need of minimal performance when there are some integration and adaptation in a component or when it is composed into a system.

5.2 Non-Technical Factors:

Application Experience: There is a need for less maintenance by application areas of software development when a definition of system requirements can be completely accurate [17]. On the other hand, there is very difficult in understanding the demand when the software is in new areas and there will be a huge change in demand for adaptive maintenance workload. Staff Stability: If for maintaining any part of the development a developer is responsible and no need to spend time by any other software engineers to know the code then there will be a reduction in maintenance costs. Application of time: Early in the software applications, coding errors comes when there is concentration of the emergence of a larger correction of maintenance workload [18]. There is more difficulty in maintaining the structure when older procedures are used to run the software and can't handle the integrity and adaptability of the heavy workload that result in increase of maintenance cost. External Environment: Workflow, reports, business rules, etc are some of the soft environment or some of the external environment. There is a need to do appropriate changes in the external environment overreliance software if there is a change in the environment. Support Environment: To accommodate new support environment there is a need to do changes in the software if the support environment is running by software [19]. The changes need to be done in network, system software and hardware to accommodate the new support environment and adaptation to maintain the workload. User needs: Users of the software is the gradual understanding of hair. The performance and functionality of software need to be deeper when the software is running and for perfection, there is need of higher requirements for performance and functionality that results in an increase of maintenance workload [20].

6 CONCLUSION

As compared to new development less attention is received by area of software maintenance estimation. Due to importance of software maintenance various models are introduced and applied to estimate the maintenance costs. On the basis of the granularity level of estimation focus, it is classified into Phase, Release and Task level maintenance estimation models. Majority of software total cost can be estimated by considering the software maintenance that is considered as an important software engineering activity. That's why we have given a brief detail about various activities associated with the current software development environment. By understanding the factors that influence the cost of software maintenance tasks helps maintainers to take decisions about their work. So, various factors associated with it also given in detail. In this work, we have classified those factors into the technical and non-technical part and both contain the number of factors that influences the maintenance cost of the software. Review on work done by various researchers indicates that prior work on estimating the maintenance cost using a better model and considering various factors that affect it can result in a reduction of cost. In the future, we can implement it using various models.

REFERENCES

[1]. P. Abrahamsson, R. Moser, W. Pedrycz, A. Sillitti, G. Succi, "Effort Prediction in Iterative Software Development Processes -- Incremental Versus Global Prediction Models", Proceedings of the First International Symposium on Empirical Software Engineering and Measurement (ESEM), 2007

[2]. F. Fioravanti, P. Nesi, "Estimation and prediction metrics for adaptative maintenance effort of object-oriented systems", IEEE Transactions on Software Engineering, vol. 27 (12), 1062–1084, 2001.

[3]. J. Choudhari, U. Suman, "Phase wise Effort Estimation for Software Maintenance: An Extended SMEEM Model", Proceedings of the CUBE International Information Technology Conference, pp. 397-402, 2012.

[4]. M. Benaroch, "Understanding Factors Contributing to the Escalation of Software Maintenance Costs", Thirty Fourth International Conference on Information Systems, pp. 1-15, 2013.

[5]. Y. Ren, T. Xing, X. Chen, X. Chai, "Research on Software Maintenance Cost of Influence Factor Analysis and Estimation Method", IEEE, pp. 1-4, 2011.

[6]. Buchmann, S. Frischbier, D. Putz, "Towards an Estimation Model for Software Maintenance Costs", 2011 15th European Conference on Software Maintenance and Reengineering, 313-316, 2011

[7]. M. Jorgensen, "A review of studies on expert estimation of software development effort", The Journal of Systems and Software, vol. 70, pp. 37-60, 2004.

[8]. R. D. Banker, S. M. Datar, C. F. Kemerer, "Factors Affecting Software Maintenance Productivity: An Exploratory Study", pp. 160-165, 2000.

[9]. B. Boehm. Software Architectures: "Critical Success Factors and Cost Drivers", IEEE transactions on Software Engineering, 2007: 965-971.

[10]. Nguyen, B. Boehm, P. Danphitsanuphan, "A Controlled Experiment in Assessing and Estimating Software Maintenance Tasks", APSEC Special Issue, Information and Software Technology Journal, pp. 681-692, 2010.

[11]. T. Wijayasiriwardhane, R. Lai, K.C. Kang, "Effort Estimation of Component based software development", a survey IET Software, vol. 5, pp. 216-228, 2011.

[12]. P. Marounek, "Simplified approach to effort estimation in software maintenance", Journal of Systems Integration, pp. 51-63, 2012.

[13]. S. M. H. Dehaghani, N. Hajrahimi, "Which Factors Affect Software Projects Maintenance Cost More?", Acta Informatica Medica, pp. 63-66, 2013.

[14]. M. Islam, Dr. V. Katiyar, "Development of A Software Maintenance Cost Estimation Model: 4 Th Gl Perspective", International Journal of Technical Research and Applications, pp. 65-68, 2014.

[15]. Y. C. Ren, "Research on Software Cost Estimation and Its Expert System," Doctor's degree of Liaoning Technical University, 2008.

[16]. Ravi, "Nonequilibrium Simulated Annealing-Algorithm Applied to Reliability Optimization of Complex Systems, " IEEE Transactions on Reliability, vol. 46, no. 22, pp. 233-239, 2007.

[17]. Q. Tang, “ Approach to Improve the Software Cost Estimation Accuracy,” Science Technology and Engineering, vol. 6, no. 4, pp. 455-461, 2006.
[18]. J. D. Zhu, C. H. Qi, X. M. Hou, “Estimating software cost based on grey theory,” Computer Engineering and Applications, vol. 46, no. 16, pp. 71-73, 2010.
[19]. J. Li, G. Ruhe, A. AlEmran, “S A Flexible Method for Effort Estimation by Analogy,” Empirical Software Engineering, vol. 12, no. 1, pp. 65-10, 2006.
[20]. S. M. Li, M. He, D. Yang, "Software Cost Estimation Method and Application," Journal of Software, vol. 18, no. 4, pp. 775-795, 2007