www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 4 Issue 7 July 2015, Page No. 13610-13614

Study of Software Quality and Risk Estimation and Quality Cost Analysis using empirical study

Dr. P. K. Suri[1], Pooja[2]

[1] Dean, Research and Development, Chairman,
MCSE/IT/MCA, HCTM Kaithal, Haryana, India
pksurikuk@gmail.com

[2] Student, HCTM Kaithal,
Haryana, India
arorapooja.arora910@gmail.com

Abstract: In this paper we calculate the cost of software product using Empirical study. Early prediction of software cost and quality is important for better software planning and controlling. In early development phases, design complexity metrics are considered as useful indicators of software testing effort and some quality attributes. Although many studies investigate the relationship between design complexity and cost and quality, it is unclear what we have learned from these studies, because no systematic synthesis exists to date. Referring to the famous statement of Tom DeMarco , "**You cannot control what you cannot measure**". Quality Measurements are – as in any other engineering discipline – also in software engineering a cornerstone for both improving the engineering process and software products. Quality Measurement not only helps to visualize the abstraction of software development process and product but also provide an infrastructure to perform comparison, assessment and prediction of software development artifacts. The large part of software measurements is to, in one way or another, measure or estimate software complexity and quality due to its importance in practices and research. The relationship between cognitive complexity and software external quality depends on comprehending ability of software developer, tester or maintainer. This factor is not deterministic and hence, cannot be investigated any other ways than empirically. In order to improve quality an organization must take into account the costs associated with achieving quality since the objective of continuous improvement programs is not only to meet customer requirements, but also to do it at the lowest cost. This can only happen by reducing the costs needed to achieve quality, and the reduction of these costs is only possible if they are identified and measured. Therefore, measuring and reporting the cost of quality (CoQ) should be considered an important issue for managers. **Risk management** is the identification, assessment, and prioritization of risks followed by coordinated and economical application of resources to minimize, monitor, and control the probability and/or impact of unfortunate event or to maximize the realization of opportunities. Risk management's objective is to assure uncertainty does not deviate the endeavour from the business goals.

Keywords: Prevention cost, Appraisal cost, Failure cost, Risk management, Empirical study.

1. Introduction

IEEE standard 1061 gives a definition of software quality: *"Software quality is the degree to which software possesses a desired combination of attributes such as maintainability, testability, reusability,complexity, reliability , interoperability , etc".* Software quality represents for wide range of desired non functional features of a software system. Software quality attributes are classified into two categories, namely external quality and internal quality. External quality characteristics are those parts of a product that face its users, i.e. maintainability, reliability, usability, etc, where internal quality characteristics are those that do not, i.e. understandability, complexity, etc. Ideally, the internal quality itself determine the external qualities as well as external quality determine quality in uses . In our model, dependent variable is design complexity, a type of internal quality. Independent variable is external quality attributes, such as maintainability and reliability.

2. Quality Cost Analysis

Quality cost is the cost associated with preventing, findings, and correcting defective works. These cost are huge. Many of such costs can be significantly reduced or completely avoided. One of the key function of a Quality Engineer is the reduction of the total costs of quality associated with products. Here are six useful definition, as applied to software product.

- **Prevention Costs**: Cost of activities that are specifically designed to prevent poor qualities. Example of "poor quality" include coding error, designs error, mistake in the user manual, as well as badly documented or un-maintain ably complex codes. Note this that most of the prevention costs do not fits within the Testing Group's budgets. Such money is spent by the programming, designs, and marketing staff.

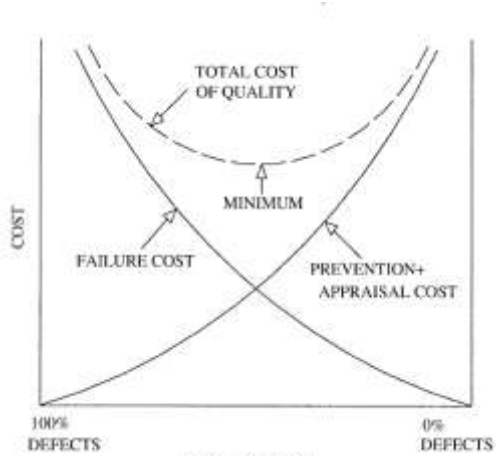


Figure 1.

- **Appraisal Cost(AC)**: Cost of activities designed to find quality problem, such as codes inspection and any types of testing. Design review are part preventions and part appraisal. To the degree that you are looking for ways to strengthen the designs, you are doing preventions.
- **Failures Cost**: Cost that results from poor quality, such as the costs of fixing bug and the cost of dealing with customer's complaint.
- **Internal Failure Cost(IF)**: Failure cost that arise before company supplies its products to the customers. Such costs go beyond the obvious costs of finding and fixing bug. Many of the internal failure cost are borne by groups outside of Product Developments.
- **External Failure Cost(EF):**Failure cost occurring after delivery or shipments of the products — & during or after furnishing of a services — to the customers.

Example are the cost of:

- Processing customer's complaint
- Customer returns.
- Claims.
- Product's recalls

Total Cos t= PC+AC+IF+EF

This represent the differences between the actual costs of products or services and what the reduced costs would be if there were no possibility of substandard service, failures of products or defects in their manufactures.

3. Software Risk and quality Management

There are many risks involved in creating high quality software on time and within budget. A software project may encounter various types of risks:

1. Technical risks include problems with languages, project size, project functionality, platforms, methods, standards, or processes. These risks may result from excessive constraints, lack of experience, poorly defined parameters, or dependencies on organizations outside the direct control of the project team.
2. Management risks include lack of planning, lack of management experience and training, communications problems, organizational issues, lack of authority, and control problems.
3. Financial risks include cash flow, capital and budgetary issues, and return on investment constraints.
4. Contractual and legal risks include changing requirements, market-driven schedules, health & safety issues, government regulation, and product warranty issues.
5. Personnel risks include staffing lags, experience and training problems, ethical and moral issues, staff conflicts, and productivity issues.
6. Other resource risks include unavailability or late delivery of equipment & supplies, inadequate tools, inadequate facilities, distributed locations, unavailability of computer resources, and slow response times.

Figure 2. Risk Management Process

Figure 2 illustrates the risk management process. This process starts with the identification of a list of potential risks.

Each of these risks is then analyzed and prioritized. A risk management plan is created that identifies containment actions that will reduce the probability of the risk occurring and/or reduce the impact if the risk turns into a problem.

3.1 Steps for Risk Management and quality maintenance :

1. Identify possible risks; recognize what can go wrong
2. Analyze each risk to estimate the probability that it will occur and the impact (i.e., damage) that it will do if it does occur
3. Rank the risks by probability and impact
4. Impact may be negligible, marginal, critical, and catastrophic
5. Develop a contingency plan to manage those risks having high probability and high impact.

3.2 Empirical study for quality management

This involves introducing assumptions or hypotheses about observed phenomenon, investigating of the correctness of these assumptions and evolving it into body knowledge . Empirical study is always attached to an environment context in which the study is performed. The formal definition of empirical software study is given as below:

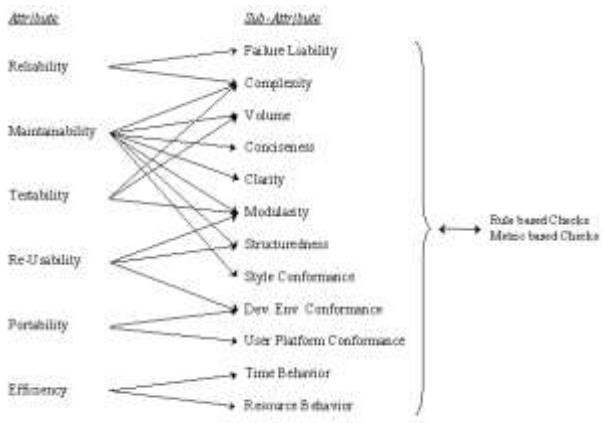
"*Empirical software engineering involves the scientific use of quantitative and qualitative data to understand and improve the software product, software development process and software management"*

The definition differentiates two approaches in empirical studies. Qualitative approach attempts to interpret a phenomenon, problem or object based on explanation that people bring to them. **Quantitative approach** this involves quantifying a relationship or to compare two or more groups . Therefore, quantitative investigations are common in empirical studies about relationship between design complexity and external software quality. In general, any quantitatively empirical study can be mapped to the following main research steps:

• Definition: formulating an hypothesis or question to test
• Planning: designing, selecting suitable sample, population, participants
• Operation: executing the design, collecting data, variables, materials
• Data Analysis & interpretation: abstracting observations into data and analyzing data
• Conclusions: drawing conclusions and significance of the study.
• Presentation: report the study.

The work within the steps differs considerably depending on the type of empirical study. Quantitative empirical studies are differentiated due to the context and control level of experiment variable. Zelkowitz describes three main groups of validation model as below:

• An observational method: collects relevant data as a project develops. There is relatively little control over the development process .
• An historical method: collects data from projects that have already been completed. The data already exists; it is only necessary to analyze what has already been collected.
• A controlled method: provides for multiple instances of an observation in order to provide for statistical validity of the results . In general, the controlled methods provide the more reliable results due to the well-control of experiment variables. However it is only possible to conduct controlled experiment for small case or laboratory situation that does not reflect the real case in industry. In practice, observational method or historical method is more common with the data collected from real projects. The drawback of these methods is the little or no control of experiment variable, which could seriously affect the reliability of the empirical result . Quantitative empirical studies include the use of statistical methods to assess and quantify the relationship between treatment groups. There are two frequently used statistical approaches to investigate the relationship between design complexity and external quality, namely correlation analysis and regression analysis. From now on, we use the term "Correlation and regression analysis" to represents for this study area.

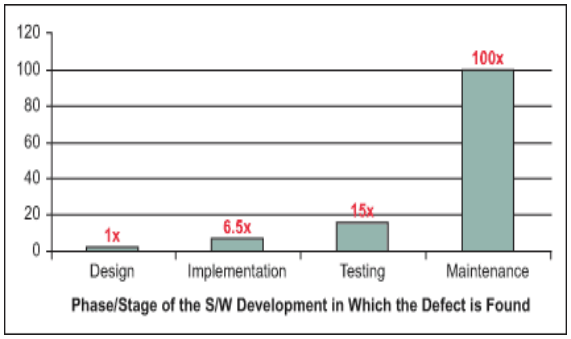


4. Defect Prevention: Reducing Costs and Enhancing Quality

"Preventions are better than cure" applies to defect in the software developments life cycle as well as illness in medical sciences. Defect, as defined by software developers, are variances from a desired attributes. Such attribute includes complete and correct requirement and specification as drawn from the desires of potential customer. Thus, defects cause software to fail to meet requirements and make customer unhappy.

And when defects gets through during the development processes, the earlier it is diagnosed, the easier and cheaper is the rectification of the defects. The end result in prevention or early detections is a product with zero or minimal defect.

How serious is defect in software development? In the United States, up to 60 percent of software developer are involved in fixing/ correcting error, Computer Finance Magazine reported in 1998. These facts alone, without consideration of providing the quality needed to please customers, shows the value of preventing software defect.



4.1 Advantage of Early Defect Detection and Risk Prevention

Data to supports the need for early fixe of software defects is supplied by several reports. The National Institute of Standard Technology (NIST) published a study in 2002 noting that the costs of fixing one bug(s) found in the production stage of software is 15 hours compared to five hour of effort if the same bugs were found in the coding stages.

The Systems Sciences Institute at IBM has reported that the costs to fix an errors found after product release was four to five times as much as one uncovered during designs, and up to 100 times more than one identified in the maintenance phases (Figure 1).

Figure 1: Relative Costs to Fix Software defects (Source: IBM Systems Sciences Institute)

Can software defects be prevented by simply logging them into some "defect tracking tool/system," documenting them and providing fixes for them? The obvious answers is no, though this is the first steps toward defect prevention.

Defect preventions involves a structured problem-solving methodology to identify, analyze and prevents the occurrences of defects. Defect preventions is frameworks and on going processes of collecting the defect data, doing root cause analysis, determining and implementing the corrective actions and sharing the lessons learned to avoid future defects.

5. Conclusion

In order to improve quality an organization must take into account the costs associated with achieving quality since the objective of continuous improvement programs is not only to meet customer requirements, but also to do it at the lowest cost. Total Quality Management (TQM) focuses on process improvement and the elimination of all forms of waste. A realistic estimation of quality costs is an essential element of any TQM initiative. However, in spite of the extensive literature on the importance and principles of quality costing, only a minority of organizations implements CoQ models and uses formal quality costing methods. CoQ reporting is beneficial at both the corporate and operational level. At the corporate level it gets management's attention and provides a benchmark against which financial improvement can be measured over time. At the operational level it helps to identify, prioritize, and select projects; provide financial benefits of process improvement and monitor project improvements.

Refrences :

1. *Krasner, H., Using the Cost of Quality Approach for Software. Crosstalk The Journal of Defence Software Engineering, Volume 11, number 11, November 1998.*
2. *(Laporte et al. 2008a) Laporte, C.Y., Alexandre, S., and Renault, A., The Application of International Software Engineering Standards in Very Small Enterprises, Software Quality Professional, Vol. 10, Number 3, June 2008, p. 4-11.*
3. *(Laporte et al. 2008b) Laporte, C.Y., Alexandre, S., O'Connor, R., A Software Engineering Lifecycle Standard for*
4. *Very Small Enterprises, in R.V. O'Connor et al. (Eds.): EuroSPI 2008, CCIS 16, pp. 129– 141,*
5. *(Mandeville 1990) Mandeville, W., A, Software cost of quality. Proceedings IEEE Journal on Selected Areason Communications, 8(2): 315.318, 1990.*
6. *(Paulk et al. 1993) Paulk M., Curtis B., Chrissis M.B., Weber C., Capability Maturity Model for Software, Version 1.1, CMU/SEI-93-TR-24, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1993.*
7. *(PMI 2008) A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition Project Management Institute, Newton Square, (PA), 2008.*
8. *(Price Waterhouse 1988) Price Waterhouse Report, Software Quality Standards: The Costs and Benefits. UK*
9. *Department of Trade and Industry, 1988.*
10. *(Radice 1985) Radice, R., A Programming Process Architecture, IBM Systems Journal, vol. 24, no. 2,1985.*
11. *(Slaughter et al. 1998) Slaughter, S.A., Harter, D.E., and Krishnan, M.A, Evaluating the cost of software quality.*
12. *Communications of the ACM, 41(8): 10.17, 1998.*
13. *(SEI 2010) SEI, CMMI® for Development Version 1.3. 2010, Carnegie Mellon University, Software*
14. *Engineering Institute: Pittsburgh.*
15. *(Liu 2007) Angel Qi Liu, Motorola Software Group's China Center: Value Added by CMMI, Data &*
16. *Analysis Center for Software, Department of Defense, Software Tech News, STN Vol. 10,*
17. *No. 1 March 2007. pp 19-23*
18. *H.J. Harrington, Poor-Quality Cost, Marcel Dekker, Inc. ASQC Quality Press, New*
19. *York 1987.*
20. *H.J. Harrington, "Quality Cost - A Key to Productivity", in Quality Costs: Ideas &*
21. *Applications, A. F. Grimm, ed.: ASQC Quality Press, American, Society for Quality Control, Milwaukee, Wisconsin, pp. 397-412, 1987.*
22. *H.J. Harrington, "Performance Improvement: a Total Poor-Quality Cost System", The TQM Magazine, Vol. 11 (4), pp. 221-230, 1999.*
23. *J. Campanella, "Quality Costs: Principles and Implementation and Use", in Quality Costs Press, J. Campanella, ed.: American Society for Quality Control, Milwaukee, Wisconsin, 1999.*
24. *J. Campanella, "Quality Costs: Principles and Implementation", in Quality Costs: Ideas & Applications J. Campanella, ed.: ASQC Quality Press, American Society for Quality Control, Milwaukee, Wisconsin, pp. 460-473, 1987.*
25. *A.F. Grimm, and J. G. Fox, "Avoidance/ Failure Costs Reversal: An Action Plan", in Quality Costs: Ideas & Applications, J. Campanella, ed.: ASQC Quality Press, American Society for Quality Control, Milwaukee, Wisconsin, pp. 409-419, 1987.*
26. *A.V. Feigenbaum, Total Quality Control, McGraw-Hill Book Company, third Edition,*
27. *Revised, New York, 1991.*
    J. Gray, "Quality Cost: A Report Card on Business", Quality Progress, Vol. 24 pp.51-54, 1995.
28. *L. P. Carr, and L. A. Ponoemon, "The Behavior of Quality Costs: Classifying the Confusion", Journal of Cost Management Practices, Vol. 12, pp. 26-34, 1994.*
29. *K. Vincent, S. Sakesun, and G. E. Norman, "The relationship between quality and quality cost for a manufacturing company" International Journal of Quality &Reliability Management, Vol. 21 (3), pp. 277-290, 2004.*
30. *J.L. Heskett, W.L. Hart, and W.E. Sasser, Service Break-through: Changing the Rules of the Games, The Free Press, New York, 1990.*
    A.C. Rosander, Applications of Quality Control in Service Industries, Marcel Dekker, ASQC Press, New York, 1985.
31. *W.E. Youngdahl, and D.L. Kellogg, "Customer Costs of Service Quality: A Critical Incident Study", Service Marketing and Management Effectiveness, Vol. 3. JAI Press, Greenwich, CT, pp. 149-173, 197*