Cost Optimization Approaches of Software Quality Assurance

R. Anitha, Dr. M.V. Srinath, Dr. Omar A. AlHeyasat

1Assistant Professor, Department of Computer Science, STET Womens College, Mannargudi, Tamil Nadu, India.
2Director, MCA, STET Womens College, Mannargudi, Tamil Nadu, India.
3Vice Dean Of Academic affairs, Computer Engineering Department Faculty of Engineering AlBalqa Applied University, Salt JORDAN.

ARTICLE INFO

Article history:
Received 25 June 2014
Received in revised form 8 July 2014
Accepted 25 July 2014
Available online 20 August 2014

Keywords:
Defect management, Cost estimation model, Cost-Schedule-Value, Software quality assurance.

ABSTRACT

Efficacious implementation of software projects is a critical strategic and competitive necessity for firms in all industrial sectors today. One of the most challenging issues in the software industry is to provide high quality software to attain total customer satisfaction. The result shows that less than 30% of the IT products are witnessed to be successful. The strategic challenge of an IT industry is to engineer a software product with minimum post deployment defects. To overcome the systematic causes of failure software quality assurance practices are employed. High quality software is provided by defect management system. It provides milestone budgets and schedules to help the project and evaluate the satisfactory progress of a project. It also helps in analyzing cost-schedule-value and focuses on controlling risks. Quality assurance provides a framework for guaranteeing the development of project life cycle. In this paper, Survey on the cost optimization techniques and cost estimation model for Software Quality Assurance (SQA) are presented.

© 2014 AENSI Publisher All rights reserved.


INTRODUCTION

SQA is an essential factor in the development of a software and ensures the quality of the product. It is a degree by which a system, component or process meets customer’s needs or expectations. Software costing is carried out with the goal of accurately predicting the cost of developing the software. The factors which affect software pricing are

- Market opportunity
- Cost estimate uncertainty
- Contractual terms
- Requirements volatility
- Financial health

Cost estimation and scheduling are carried together. The costs of development are primarily the cost involved, so the effort computation is used in both the cost and the schedule estimate. The main goal of SQA is to enhance software quality by appropriate monitoring of both the development process and software. The quality cost is categorized into

- Prevention Costs (PC)
- Appraisal Costs (AC)
- Internal Failure Costs (IFC)
- External Failure Costs (EFC)

Prevention costs are the cost associated with quality planning, designing, implementing quality system and process enhancements. Appraisal cost deals with evaluating the products and product materials to assure conformance with performance requirements. Failure cost is categorized into external and internal failure costs. IFC composed of defects of processes, and products that fail to meet quality requirements. EFCs are generated with defective services and processes during customer usage. It includes repairs, warranties, poor packaging, and customer returns. A defect refers to any inaccuracy in a software process or software work product. The term defect denotes fault, error or failure. The expected development of high quality software is the main problem due to the late defect removal. 99% of the software defects are determined by the defect prevention approach. Defect prevention is the most vital part of software quality assurance. The defect detection and

Corresponding Author: R. Anitha, Assistant Professor, Department of Computer Science, STET Womens College, Mannargudi, Tamil Nadu, India. E-mail: veenanitha@yahoo.co.in
removal is a software inspection that provides technical support for the detection of defects. Inspection is a significant approach for generating ideal software through improved methodologies of unaided inspection schedules. Inspection enhances quality and also saves the defect cost. 25% to 30% of testing and 13% to 15% of inspection out of whole project effort time is needed for 99% of defect elimination.

Quality assurance includes all the activities related to satisfying the relevant quality standards for a software product. Cost estimation models are mathematical algorithms, used to estimate the cost of a product. Cost estimation is done throughout the software life cycle. It needs effective monitoring and control over the software costs. To estimate the cost, set of inputs are given as input to the process. Then, the process uses the inputs to generate the output. An effective cost optimization is based on the deep analysis such as cost driver analysis. With this analysis, management can implement a programme of change to drive out costs through strategic changes.

The paper is organized as follows, section 2 illustrates about cost optimization and software defect detection approach. Section 3 deals with results and discussion. And section 4 describes conclusion and proposed work.

Approaches to Optimize the Cost of Software Quality Assurance:

A) Commercial Off-The-Shelf Model (COTS):

A refined model (Rawashdeh, et al., 2006) for selecting the best commercial off the self-software product among the alternatives for each module. This model minimizes the quality within a budgetary constraint. COTS software is widely used in IT strategy for building and delivering the system. It is based on a black box testing where the user has no access to source code. The use of COTS components is a vital need because of the possibility to construct the system at lower cost. Searching COTS components require to overcome several problems like, the growing size and availability of the COTS market place. GOTHIC method (Ayala and Franch, 2006) provides a methodology framework to construct the components of COTS. It does not depend upon the extent and the characteristics of the taxonomy in the business application. This is the intellectual tool that helps to organize goals in order to implement a structure and locate mechanism. The organization of goals comes from the analysis of post and pre conditions stated for each goal.

B) An Extended Cost-Effectiveness (CE) Model:

This model (Freimut, et al., 2005) defines the cost effectiveness over the potential defect cost. The potential defect cost without inspection parameter is the defect rework cost, which is incurred if no inspection takes place. The cost expended by inspections parameter is the cost spent on performing code inspections. The cost saved by inspection traps the cost saved in later phases due to the defect detection in code inspection. A minimal set of assumptions is made to obtain the cost effectiveness that is operationalized under realistic constraints. A multi-criteria genetic algorithm (Martens, et al., 2010) to software architecture modelled with the Palladio Component Model supports performance, reliability, and cost.

This approach is used to design the space spanned by different design options. It is systematically used to explore the Meta heuristic search technique. This reduces development cost and results better quality in software architecture. Software cost (Kashyap and Misra, 2014) estimation method using different attributes deals with selection, quantification, and comparison of different attributes. A software cost depend on the expenses of the resources and time, which are the attributes of the software. Cost estimation is a tedious job which is compulsorily required in software development activity.

C) Activity-Based Quality Model:

This approach (Deissenboeck, et al., 2007) mainly aims at describing complex quality criteria by breaking them into sub-criteria. These models were designed in a tree-like fashion with abstract quality parameters. The set of relevant activities depends on the particular development and maintenance process of the organization. Prevention-Appraisal-Failure model (PAF) (Sharma, et al., 2007) was used to measures the cost quality activities in software development. According to the PAF model, the quality cost is measured as the sum of conformance and non-conformance costs. Conformance cost, is divided into prevention and appraisal cost. This cost is related to the implementation of practices and techniques to prevent poor quality. Non-conformance cost is the cost of quality failure or the cost of non-conforming to the pioneer requirements of the software. The PAF model reduces the cost of failure and invest for the prevention activities.

D) Four Step Approach Model for Defect Management (FAMI):

This approach (Nair, 2012) provides an effective inspection of defect in software development phase. Each phase of software development contains deliverables. Each deliverable undergoes an inspection for static and dynamic defects. It ensures the implementation of product with insignificant risk. It also improves the quality of the product, increases productivity, and reduces the cost for rework. The four steps of FAMI approach are as follows:

1. Integration of inspection life cycle in software development process.
2. Introduction of Depth of Inspection (DI) as a quality metric.
3. Parameters influencing depth of inspection.
4. Use of Bayesian belief approach to achieve the depth of inspection.

The practice of FAMI leads to the development of zero-defect products. It reflects on the continuous process improvement in software industries through defect management.

E) Quamoco Quality Modelling:
An explicit Meta model (Wagner, et al., 2012) designed to specify operationalized quality models in a flexible manner by assessing the factors. The resulting QM represents the consolidated view on the quality of software source code and is generally applicable to any kind of software. It allows for comprehensive, tool-supported quality assessment without requiring large adoption or configuration support. The Quamoco quality assessment method models the preferences of decision makers for a product’s quality using the concept of quality. It measures the relative satisfaction of the decision maker concerning the quality of the software product.

F) Defect Prevention Approach:
The key point of this approach (Islam, et al.,) is to detect and prevent defect at early stages of software development. Project cost estimation and project scheduling are carried together. The cost of the software development is the cost of the effort involved. Cost estimation is done before the schedules are drawn up. Cost can be estimated through three parameters.
1. Hardware, software and maintenance cost
2. Travel and training cost
3. Effort cost

The estimation methodology (Suma and Nair, 2010) is based upon the use of multiple estimates, data-driven estimates from historical experience, risk and uncertainty impacts. Debugging comprises of finding the defect and fixing it. If a defect resides for a longer time in the product, then it is more expensive to fix it. It is mandatory to reduce defect injection and boost defect removal efficiency. Defect Removal Efficiency (DRE) quantifies the supremacy of the product by computing the number of defects.

\[ \text{DRE} = \frac{\text{np}}{\text{nl}} \]  

Where, np denotes the number of defects removed during the development phase and nl indicates the total number of latent defects. The Defect Prevention (DP) (Lazić, 2010) techniques followed in the company includes proactive, reactive and retrospective DP. Pro-active DP creates an environment for controlling defects rather than just reacting to it. Reactive DP diagnose and conducts Root Cause Analysis (RCA) for defects meeting at trigger points or logical points. Retrospection is performed at the end of the software development to identify the strong points and to explore the areas requiring perfection.

Defects are identified by preplanned activities which, are intended to uncover defects. Orthogonal Defect Classification (ODC) is a technique (Kumaresh and Baskaran, 2010) used for identifying defects, where the defects are grouped into two points. It divides each defect into orthogonal attributes, where some are technical and some are managerial. The defects are classified into first level ODC and second level ODC. Defect prevention practices upgrades the ability of software developers to study from the fault. It also reduces development time and cost, increases customer satisfaction, thereby decreases the cost and improves the product quality.

G) Hybrid Cost Estimation Model:
This model presents (Trendowicz, et al., 2006) an enhanced process to estimate the cost of a software product. It is a hybrid method of combining data and expert based cost estimation approaches. In this model the input data are validated first, called as pre-estimation analysis. Later, the estimation model is built and is known as post-estimation analysis. Pre-estimation analysis concerns about the reliability of the software product. Post-estimation shows an improvement in a model accuracy. It is important to establish a maintenance process for the estimation model in order to re-calibrate the model and re-increase the estimation accuracy.

H) Cost Estimation Technique:
The techniques (Niazi, et al., 2006) are classified into two groups as qualitative and quantitative. Qualitative cost estimation is built on the analysis of a new product with products. These are manufactured in order to pinpoint the correspondences in the new one. The identified similarities help to combine the past data into the new product, which reduces the cost estimation. Qualitative techniques are the detailed analysis of product design and its features corresponding to the manufacturing processes. The intuitive cost estimation
techniques based on the past experience. A domain knowledge generates cost approximations for parts and assemblies. The knowledge is stored in the form of rules, decision tree, judgments.

Case Based Methodology (Yang, *et al.*, 2012) also known as Case Based Reasoning (CBR) makes use of the information from the previous designs. It closely matches the parameters of new designs from a database. This system requires necessary change, which is done according to the new design. All the necessary changes are made to the new design that follows the outlined design specification. The new scheme is later stored in the database. This framework permits the cost estimation for a new product by incorporating the past results. This approach is helpful in making good estimation at the conceptual stages. Software cost estimation is significant for budgeting, risk analysis, project planning, and software improvement analysis. In this, the cost is estimated (Zia, *et al.*, 2011) using fourth generation language environment. It has a better predictive accuracy.

I) Cost Sensitive Technique Using Decision Tree:
The cost based act of a software quality prediction model (Seliya and Khoshgoftaar, 2011) was determined after the model-training process. Boosting is a Meta learning technique (Sun, *et al.*, 2007), which improves the performance of classifiers by iterative building a collection of classifiers. Software quality economic model traces design decisions and possible alternatives. In this model, it is possible to minimize the cost of switching between design alternatives when the current one did not fulfill the quality constraints. Software quality dimension satisfies the degree to which customers or client perceive a software product to meet their requirements.

Cost of Quality (CoQ) (Khan and Beg, 2014) indicates any cost, where the organization incurs from a repeating process in order to complete the work. Cost of Software Accounting (CoSQ) is useful to enable the discerning of the economic trade-offs involved in delivering good-quality software. The goal of the quality cost analysis is not to reduce the cost, but to make sure that the cost are spent at the right kind. It detects, quantifies, and adds all the positive factor. The key consideration in cost analysis is the visibility of the quality. It is the visibility earned from the cost of quality analysis. It enables the QA people involved to focus the attention on the activities which discover, and correct the root cause of the defects. Performing Quality Assurance is the process of auditing the quality requirements and the results from quality control measurement. It assures the quality standards and operational definitions that are used across the development of the software products.

J) Knowledge-Based Advisory system:
A prototype called knowledge-based advisory system (Eldrandaly, 2008) is designed to play the role of a “virtual quality editor”. It helps the individuals, organization, and software companies to implement the quality models. SQA is a planned and organized approach to ensure the software processes and products confirms to the recognized standard. The models for Software Quality Assurance is Capability Maturity model Integration (CMMI). The implementation of this model is a tedious and costly task for some firms. The Gauging Absence of Prerequisites (GAP) is identified between the prerequisites and the existing system, this provides the assessment results.

Component Object Model (COM) (Jones, 2008) is used to design and integrate the diverse components of the prototype system to declare system interoperability between this components. Measuring defect levels and defect removal efficiency levels are the easiest form of software measurement. To measure defect potential, all the records of defect are made accurate. The number of defects found during the reviews, inspection, and testing is a straight forward. To calculate the defect removal efficiency, customer-reported defects were submitted during a fixed period. The final result is compared with the internal defects.

K) Quantitative assessment of software quality:
The software quality estimation models (Azar, *et al.*, 2009) are used to estimate software quality characteristics from the measurable attributes. These take distinct forms like rule-based models, statistical models and decision trees. A search-based software engineering approach is used to enhance the prediction accuracy of software quality estimation models. A novel genetic algorithm-based data sampling approach (Drown, *et al.*, 2009) named evolutionary sampling is used to enhance the software quality modeling for high-assurance systems.

The approach is compared with multiple existing data sampling methods, including one-sided selection, random under sampling, random oversampling, Wilson’s editing, Synthetic Minority Oversampling Technique (SMOT), cluster-based oversampling, and border-SMOTE. The results showed that the evolutionary sampling enhances the performance of software quality models for high-assurance systems. It is proved to be better than most of the other data sampling approaches. Faults-Slip-Through (FST) (Humphrey, 2008) is the measure that determines which faults would be more cost-efficient to identify in earlier phases. The cost of defect removal is computed for each and every quality assurance activity by multiplying the number of defects removed by the relative cost of removing a defect.
RESULTS AND DISCUSSION

The results of the survey are shown in Table 1. Various approaches for software defect detection and prevention and cost optimization approaches of software quality assurance are depicted. Selection of better COTS enhances the software quality within budgetary constraint. A knowledge based advisory system for software quality assurance contains three-tier architecture. The GAP between the prerequisites are identified. A four-step approach model of inspection (FAMI) consists of three steps. Metrics such as depth of inspection and inspection performance metric are formulated. Test optimization and productivity improvement approaches are used to optimize the quality assurance cost.

Table 1: Techniques for cost optimization of software quality assurance.

<table>
<thead>
<tr>
<th>AUTHORS</th>
<th>YEAR AND REFERENCE</th>
<th>TECHNIQUE</th>
<th>PERFORMANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ayala.C and Franch.X</td>
<td>2006</td>
<td>GOThiC method</td>
<td>Provides a methodology to construct the components of COTS.</td>
</tr>
<tr>
<td>Deissenboeck, F., et al</td>
<td>2007</td>
<td>Activity based quality model</td>
<td>Improves the maintainability of the software product.</td>
</tr>
<tr>
<td>Eldrandaly, K</td>
<td>2008</td>
<td>Knowledge based advisory system</td>
<td>Virtual quality editor is used to implement the quality models.</td>
</tr>
<tr>
<td>Freimut, B., et al</td>
<td>2005</td>
<td>Cost Effectiveness (CE) Model</td>
<td>Cost effectiveness is measured by combining the project data and expert opinion.</td>
</tr>
<tr>
<td>Humphrey, W.S</td>
<td>2008</td>
<td>Fault Slip Through method (FST)</td>
<td>Determines the fault at early stage and cost of defect removal is calculated.</td>
</tr>
<tr>
<td>Jones, C.</td>
<td>2008</td>
<td>Component Object Model (COM)</td>
<td>Design and integrates the component of the prototype system.</td>
</tr>
<tr>
<td>Khan, P. and Beg, M</td>
<td>2014</td>
<td>Cost Of Quality method</td>
<td>Indicates any cost where the organization incurs from the repeated process.</td>
</tr>
<tr>
<td>Kumaresh, S and Baskaran, R</td>
<td>2010</td>
<td>Orthogonal Defect Classification (ODC) Technique</td>
<td>Reduces development time and cost, increases customer satisfaction.</td>
</tr>
<tr>
<td>Lazic, L</td>
<td>2010</td>
<td>Quantitative defect management</td>
<td>Reduces defect injection and boost the defect removal efficiency.</td>
</tr>
<tr>
<td>Martens, A., et al</td>
<td>2010</td>
<td>Palladio Component Model (PCM)</td>
<td>Reduces development cost and improves the quality of the software architecture.</td>
</tr>
<tr>
<td>Nair, T.</td>
<td>2012</td>
<td>Four Step Approach Model (FAMI)</td>
<td>Provides an inspection of defect in software development phase.</td>
</tr>
<tr>
<td>Niazi, A., et al</td>
<td>2006</td>
<td>Cost Estimation Technique</td>
<td>Identifies the similarity and combines the past data with the new product.</td>
</tr>
<tr>
<td>Rawashdeh, A. and Matalkah, B</td>
<td>2006</td>
<td>Commercial Off The Shelf (COTS)</td>
<td>The quality is enhanced within the budgetary constraint.</td>
</tr>
<tr>
<td>Seliya, N and Khoshgoftaar, T.M.</td>
<td>2011</td>
<td>Cost Sensitive Technique using Decision tree</td>
<td>Predicts the quality of the software using decision tree.</td>
</tr>
<tr>
<td>Soma, V. and Nair, T</td>
<td>2010</td>
<td>Defect Removal Efficiency Model (DRE)</td>
<td>Quantifies the supremacy of the product by evaluating the number of defects.</td>
</tr>
<tr>
<td>Sun, Y. et al</td>
<td>2007</td>
<td>Cost Sensitive Boosting</td>
<td>Improves the performance of the classifiers.</td>
</tr>
<tr>
<td>Trendowicz, A., et al</td>
<td>2006</td>
<td>Hybrid Cost Estimation model</td>
<td>Enhances the process to estimate the cost of the software.</td>
</tr>
<tr>
<td>Wagner, S. et al</td>
<td>2012</td>
<td>Quamoco Quality modeling</td>
<td>Measures the quality of the software product.</td>
</tr>
</tbody>
</table>

Conclusion:

The paper presents the survey resulted in cost optimization approaches of software quality assurance. The survey describes various methods and techniques knowledge based advisory system, four step approach model for inspection (FAMI), test optimization, orthogonal based classification technique and QUACAMO quality modeling were discussed. The paper examined all the categories, in detail with reference to the published literature. The significance of cost estimation in the development process is also discussed. Cost optimal use analytical quality assurance and the analysis is done over the components of the system.
REFERENCES


