*Research Journal of Management Sciences* E-ISSN 2319–1171
Vol. **5(7),** 1-5 July **(2016)** *Res. J. Management Sci.*

# Optimizing Software Development Process through Effective Reviews: Method of Eliciting Defects at the origin using Checklists

**Uma Sankar S.S.[1*] and R. Jubi[2]**
[1]Research and Development Centre, Bharathiar University, Coimbatore, Tamilnadu, India
[2]Mar Thoma Institute of Information Technology, Kollam, Kerala, India
umasankar.aniyoor@gmail.com

**Available online at: www.isca.in, www.isca.me**
Received 1st May 2016, revised 1st June 2016, accepted 15th June 2016

## Abstract

*Software development companies practice defect prevention and defect removal activities to minimize defects in their customer deliverables. Reviews are one of the best and easily adaptable techniques to find the defects in the work product. It facilitate the detection of defect at the time of origin and it in-turn reduce the turnaround time for attaining the work product quality. Reviews are considered as a preventive measure to avoid the injection of defects to subsequent phases of software development cycles and improve work product quality, however it increases Cost of Quality (CoQ), more precisely it increases appraisal/prevention cost. So, to make the review effective and perform it in minimum time is an essential aspect to execute the project within time and budget. This study is an attempt to make the reviews effective by using checklists in reviews. The study revealed that, the use of checklists in reviews improves review quality and reduce turnaround time for the work product to attain the stability within budget and time.*

**Keywords:** Checklist based Review, Defect prevention, Software quality, Cost of Quality, Defect Removal Efficiency.

## Introduction

The quality of the software depends on apt people, process and tools used in Software Development Life Cycle (SDLC) process. Software quality can be achieved by adopting two major activities, viz. Defect Prevention and Defect Removal. Defects can be prevented by adopting good practices, apt tools, well documentation, effective tracking systems and configuration and change control processes. Defect can be detected at the early stages of development through Reviews. The National Institute of Standard Technology (NIST) published a study in 2002 revealed that errors introduced in the coding/unit testing stage was twice as costly to fix the error if it was not found until the integration phase and five times as costly if it was not detected until post-product release[1].

As part of this study a survey has been conducted among the software professionals working in MNC companies in Technopark, Trivandrum to assess the role of reviews and usage of checklists in their work product development. Technopark is one of the largest software technology parks in India operating under Information Technology Department, Government of Kerala. There are around 300 software companies with approximately 46,000 IT professionals working in various projects[2]. The survey revealed that reviews are part of their development process irrespective of the size of the company and type of the project. However, optimizing the review process by adopting suitable tools are not used commonly. So there is a scope for improvement in reviews by make it effective by adopting suitable processes which should also minimize Cost of Quality (CoQ).

The objective of this study is therefore examines the possibility of using checklist in review process thereby increase 'Defect Removal Efficiency' (Measure of the development team ability to remove defects prior to release[3]) with minimum CoQ.

## Problem Definition

Primary and secondary data revealed that there is a scope for improvement in review process. So, the problem under this study is defined as: 'How do we reduce software development time and achieve software quality by using checklists in reviews with minimum CoQ'.

## Theoretical Framework

Dependent variable is software project development time. Independent variables are Software quality, Development cost, Review efficiency, Cost of Quality (CoQ) etc.

The use of checklist in review process improves review efficiency and facilitates to detect defects at the early stages of development. It eliminates rework and therefore it avoids unwanted utilization of manpower and other resources thereby reduce the software development cost. Checklist based reviews minimizes the personal centric processes. Checklist based review can be easily adaptable irrespective of the type of the company and area of operation. The cost of quality is comparatively very less for checklist based reviews.

## Data Source and Sampling Design

Primary data has been collected through direct observation and administrating the questionnaire. Secondary data has been collected from various sources from internet and books. Sampling size has been determined using Krejcie and Morgan Table[4]. As per Krejcie and Morgan table with accepted margin of 5% error and with 95% of confidence level the calculated sample size is 377 for the population of 20,000 IT professionals (MNC companies). Therefore 400 questionnaires has been distributed randomly and got 380 responses. Refer Table-1, the responses has been tabulated based on the interest of the study.

## Data Analysis

As part of the survey 380 responses were collected from 50 MNC companies. Data has been consolidated company wise to analyze the general practices possessed in their software development process. Excel charts has been used for plotting the values. Individual responses were taken for analyze the effectiveness of checklist in reviews. The reason is, some of the participants were using their own checklist to make their review effective and systematic but there is no common practice in their companies to use the checklist in reviews.

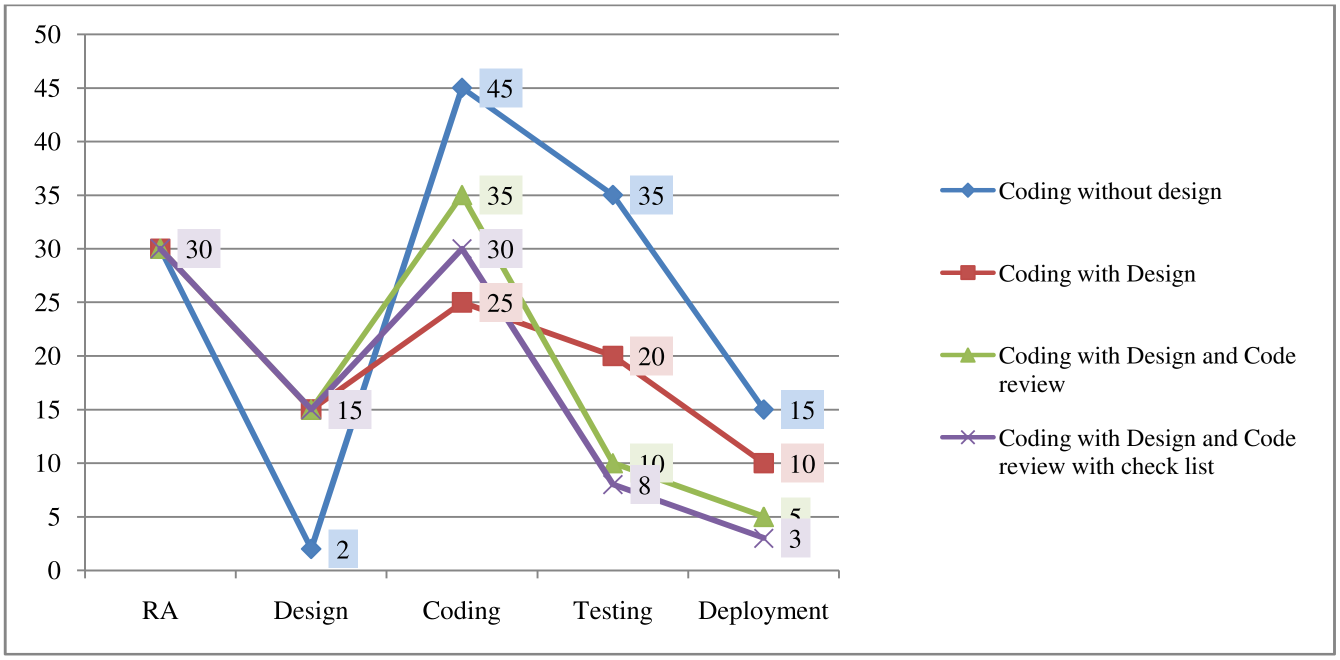
Refer Figure-1, It shows that the time conception for coding, testing and deployment efforts including reviews is very high in the case of absence of design. Coding effort increased approximately 80%, testing effort increased by 75% and deployment effort increased by 50%. So the overall time conception and thereby cost increased approximately by 27%. However, approximately 78% of increase in cost and effort if coding is done without design with respect to the coding with design and using code review checklist.

From the analysis it has been found that less than 10% companies are using checklist in their reviews. However some of the members in those companies are using their own review checklist for review tasks. It is also revealed that, reviews using checklist will increase the productivity and it aids the team to deliver the product on time by capturing the bugs at the time of origin itself, it also reduces the rework effort and supports the findings of NIST's study report published in 2002 (1st reference). Checklist based review facilitate the team to correct their mistakes and they may get chance to avoid such mistakes again. Refer Figure-1, it provides a comparative analysis of review and checklist based reviews in the SDLC process with respect to the absence of those good practices.

## Testing of Hypothesis

We are using ψ2 (Chi-Square) test because it is not possible to make any assumption about the distribution of the population from which the samples are drawn. It is a non-parametric test. The following sections provide the details of ψ2 test and its observed statistical inferences.

The objective of this study is to ascertain whether the checklist based review reduce the development time and achieving the intended software quality with minimum CoQ. Refer Table-1. We have taken dependent variable as development time comparing it with three factors one, absence of review, second presence of review and third, use of checklist in review.



**Figure–1**
**Time consumption**

**Table-1**
**Contingency Table for ψ2 test**

| | Without Review | With Review | Review with Checklist | Row Total |
|---|---|---|---|---|
| Software development time is less | 70 | 65 | 100 | 235 |
| Software development time is high | 40 | 40 | 65 | 145 |
| Column total | 110 | 105 | 165 | 380 |

From the above data it has been found that the development time can be less if we use the checklist in review. 26% of participates agrees that the software development cost will decrease if we use checklist in reviews. Therefore the NULL hypothesis H0 is:

H0: Checklist based review will reduce the software development time.

Table-2 provides the observed and expected frequencies for calculating the ψ2 by using the formula

$\psi 2 = \Sigma ((O-E)^2 / E)$

**Table-2**
**Observed and Expected frequency tabulation**

| O | E | $(O-E)^2$ | $(O-E)^2 / E$ |
|---|---|---|---|
| 70 | 68.03 | 3.9 | 0.06 |
| 40 | 41.97 | 3.8 | 0.09 |
| 65 | 64.93 | 4.9 | 0.08 |
| 40 | 40.06 | 3.6 | 0.09 |
| 100 | 102.04 | 4.16 | 0.04 |
| 65 | 62.96 | 4.16 | 0.07 |

$\Sigma ((O-E)^{2/} E) = 0.43$

$\psi 2 = \Sigma ((O-E)^2 / E) = 0.43$

Degree of freedom = (r-1) (c-1) = (2-1) (3-1) = 2.

Level of significant = 5% = 0.05.

Therefore the table value (Points of ψ2 distribution) for degree of freedom 2 and level of significance 5% = 5.99147.

So, the calculated value is less than the table value. Therefore H0 has been accepted. I.e., Checklist based review will reduce the software development time.

## Checklist Based Review

From the present system study it has been found that around 9 to 10% of development time and there by the development cost reduced by adopting the checklist based review as part of the SDLC process (Figure-1). Checklist based review facilities the development team to improve their skills and it helps to build a system which is process centric rather than personal centric. It means that to make the review effective the skilled resources may not be required if the apt checklist has been used. This will make the company to achieve the quality of work product with in their budget. Checklist also helps the team to conduct the review more effectively with shorter in time compare to the review without checklist. It has to be noted that appropriate revisions has to be made continually in checklist to meet the review system suitable to make the expected quality. The review efficiency may not be ensured by simply using these checklists. It means that the software service companies have to design the checklists suitable to their scope and the domain in which they are working.

## Objective of the Proposed System

The objective of the proposed system is to provide the details of how to improve review process more effectively within the time and budget. As a defect detection process, reviews are must in SDLC processes. It helps the team to make the quality work product within time and budget and helps the company to be competitive. Effective reviews will also increase the team collective skills. So checklist facilitates an effective review process which uplifts the company from a personal centric system to the process centric system.

## Values and Benefits

The following are the major benefits i. It increases Defect Detection Efficiency, ii. It facilitate effective reviews and thereby improves work product quality, iii. It increase the team members skill set, iv. It reduce the turnaround time of the product to the market, v. It makes the process more process centric rather than the personal centric, vi. It reduces the total cost of the development and minimizing CoQ, vii. It makes the company more competitive thereby increase the market share.

## Implementation

Checklists are generally a document with YES/NO questions. Refer Annexure-1: Guideline for preparing the checklist & Annexure-2: Sample SRS checklists. It has to be prepared by referring the respective procedure or guidelines or by referring the acceptance criteria given by the customer. Preparation of checklist is a critical activity. It is the responsibility of the Technical Architect to directly involve or delegate to the competent team member.

Before baselining the checklist, Project Manager (PM) has to review its correctness and completeness and take the ownership. It has to be noted that this checklist will act as a guideline document to make the review effective. So the completeness and correctness of the checklist decides the quality of the review.

## General Findings

i. One of the core reasons of software development failures are due to poor software requirement engineering process[5]. It is because of lack of critical reviews and absence of client approval. ii. Poor configuration control and change management increases development cost[6]. Companies has to do a proper requirement change management to tackle the changes and find out impacts in advance. iii. Reviews are best tools for defect removal process. It is easy to implement and facilitate the elicitation of defects at early stages of development. iv. Reviews reduces the rework thereby reduce the development time and cost. v. Checklist based reviews facilitate a systematic reviews. It also eliminates the personal dependencies in reviews. vi. Checklist based reviews makes review more effective, it helps junior members also to be participate in reviews. In turn it reduce the involvement of experts in the review process thereby reduce the cost of development. vii. Checklist based reviews makes the work product more quality. In software development it considerably reduces the testing effort. viii. It is difficult to design a generic checklist. It varies depends on the type of work. So, companies have to design their checklist based on their requirement and needs.

## Conclusion

Reviews are very effective tools to detect defects at very early stages of development. It is easy to implement and it improves team quality. As a defect removal tool it helps the team to detect the bugs at the time or origin, therefore it avoids the rework and cost required for fixing it in later stages. Checklists in reviews make review more effective. It provides a proper guideline and facilitates the team to do their reviews with more focus. Consolidation of review output, ie, defect count with respect to defect type will helps the team to take appropriate corrective measures to avoid the similar bugs. The data captured during the review process helps the management to identify the weak areas. Based on this input, management can initiate training to improve the production quality.

It is to note that the checklist cannot be design generically. It should be design with respect to the requirements and scope of the work. It is also noted that checklist has to be revised on a continual basis and it has to be ensure that it is effective by analyzing review and defect removal efficiency parameters statistically. So to conclude with, checklist based reviews is one of the most effective tools the software companies can adopt to reduce the software failures and increase quality.

It facilitates the company to run the project in competitive cost and meet the expected quality within in time and budget.

## Reference

**1.** Gregory Tassey (2002). The Economic Impacts of Inadequate Infrastructure for Software Testing. National Institute of Standards and Technology, http://www.nist.gov/director/planning/upload/report02-3.pdf, 4th Mar 2016.

**2.** Techno Park (2016). IT companies and IT professionals in Technopark, Trivandrum. http://www.technopark.org/, 4th Mar 2016.

**3.** Software testing concepts (2016). Defect Removal Efficiency definition. https://swtestingconcepts.wordpress.com/test-metrics/defect-removal-efficiency/,23rd Mar 2016.

**4.** Uma Sekaran (2008). Research Methods for Managers - A Skill-building Approach. 4/e, Wiley-India Edition, India, pp 293-294. ISBN 978-81-265-0928-7.

**5.** Adu Michael K. and Alese Boniface K. (2014). Inadequate Requirements Engineering Process: A Key Factor for Poor Software Development in Developing Nations: A Case Study. *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 8(9), 2014, http://waset.org/publications/9999244/inadequate-requirements-engineering-process-a-key-factor-for-poor-software-development-in-developing-nations-a-case-study. 23rd Mar 2016.

**6.** In four (2007). The Configuration Management Benchmark Report. Aberdeen Group, www.innofour.com/download/INNOFOUR01453.pdf, 23rd Mar 2016.

**Annexure-1: Guideline for preparing the checklist**
The following are the major points to be consider while preparing the checklist,

- It should be simple and easy to understand
- It will not consume more productive hours to fill
- Preferably question shall be close ended (YES/NO/NA)
- Checklist should meet the objective of the review
- No ambiguous question shall be there in the checklist
- Appropriate explanation shall be given if the question is complex
- Phrase the question in short, lengthy questions will make itself complex
- It has to be prepared by referring the respective procedures, standards and guidelines
- Consider the limitations while preparing the checklist, i.e. It should be realistic to meet the objective within time and budget
- Checklist has to be verified periodically and ensure its compliance and effectiveness to the meet quality objectives
- Checklist has to be baselined only after review
- Appropriate modifications in the checklist has to be made based on the ‘Defect Removal Efficiency’,
- Mandatory fields have to be marked appropriately.
- After filling the checklist by the reviewer, the Project Manager or Project Leader has to verify the checklist comments.

**Annexure-2**
**Software Requirement Specification (SRS) checklist**

Project: ………………………….. Phase: ……………………Reviewer: …………………Date: …………………………
Document under review: ………………………………………… Version: …………..…….
Is the work product fit for release: [YES/NO] Re-review required: [YES/NO]

| **Sl** | **Sample checklist** | **Y/N/NA** | **Comments** |
|---|---|---|---|
| 1 | Is the document comply Standards/guidelines and naming conventions established for the document? | | |
| 2 | Are all the requirements from customer documented in SRS? | | |
| 3 | Is all the changes documented in SRS (maintenance project)? | | |
| 4 | Are all requirements testable? | | |
| 5 | Is Graphical User Interfaces designed? | | |
| 6 | Is there a high-level system overview? | | |
| 7 | Is system functional flow documented? | | |
| 8 | Are all definitions, acronyms, and abbreviations included? | | |
| 9 | Are the software functions described at a high-level? | | |
| 10 | Are the user characteristics defined? | | |
| 11 | Are general design and implementation constraints noted? | | |
| 12 | Are general assumptions that affect implementation been stated? | | |
| 13 | Are general dependencies noted? | | |
| 14 | Are timing requirements provided? | | |
| 15 | Are memory requirements provided? | | |
| 16 | Is functional requirements been stated in terms of inputs, outputs, and processing? | | |
| 17 | Is performance requirements mentioned? | | |
| 18 | Are the functional requirements clear and specific enough to be the basis for detailed design and functional test cases? | | |
| 19 | Are software quality requirements identified (e.g., reliability, portability, reusability, maintainability)? | | |
| 20 | Are all delivery requirements identified? | | |
| 21 | Are functional requirements uniquely numbered? | | |
| 22 | Are requirements stated consistently without contradicting themselves or other requirements? | | |
| 23 | Was the document baselined prior to the Software Requirements Review? | | |
| 24 | Is customer approval got? | | |
| 25 | Is the document prepared in the prescribed format (template) | | |

General comments about the document: -----------------------------------------------------------------------------------------------------------

Note: The above checklist is applied in review process. It is applied prior to baseline the SRS document and used for the design phase. Make sure that the review team has to use this checklist for each and every requirement changes happened in the development system.