

SPM-ICON 2009

4th Annual International Conference on Project Management

October 29-30, 2009 Bangalore

Defect Prevention and Management

by

Franklin Joseph — Team Lead

Ekanath Santharam — Team Lead

Honeywell Technology Solutions.

Copyright: Quality Solutions for Information Technology Pvt. Ltd.

Published with permission for restricted use in 'SPM-ICON 2009' in agreement with full copyrights from owner(s) / author(s) of material. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior consent of the owner(s) / author(s). This edition is manufactured in India and is authorized for distribution only during 'SPM-ICON 2009' as per the applicable conditions.

Hosted By



White Paper Submitted for SPM-ICON 2009

Defect Prevention and Management

Author(s):

Franklin Joseph

Designation: Team Lead

Email: franklin.joseph@honeywell.com

Ekanath Santharam

Designation: Team Lead

Email: ekanath.santharam@honeywell.com

Company Address

Honeywell Technology Solutions,
151/1, Doraisanipalya, Bannerghatta Road,
Bangalore - 560076, INDIA

Abstract:

Introduction:

Defect Prevention is a methodical procedure applied to the software development life cycle that identifies root causes of defects and other process problems and prevents them from recurring. As a step ahead, Defect Prevention techniques can also be applied to investigate patterns of defects that commonly occur in the Software Development Process to take necessary corrective actions to stop the negative trend.

Defect Prevention has its own importance in software industries since the occurrence of defects is the greatest contributor to significant increases in product costs due to rectification and rework time.

This paper elucidates different Defect Prevention techniques that have been adapted by us in our projects which yielded fruitful results.

Audience:

Audiences of this paper are **Program managers** who would like to control the Cost of Poor Quality (COPQ) of their running programs, **Developers** who try to avoid the repetitive coding mistakes to create defect free solution, **Quality engineers** who are keen on establishing effective process and intelligent testing to make robust delivery to customers.

Area of Application:

Defect Prevention is applied in Software Development life cycle to increase the quality of the product and to reduce the cycle time. It is also one of the Key Process Area (KPA) in Level 5 Capability Maturity Model (CMM).

Benefits:

Following are the benefits in adapting the Defect Prevention techniques into Software development and testing,

- Increased quality of the software

- Lower development cost
- Reduced cycle time

Issues and Challenges:

Following are challenges in implementing Defect Prevention techniques in Software development,

- Analyzing the root cause of the defects would be time consuming exercise and often overlooked as a non-value added activity in projects.
- The types of defects are project specific and hence no generic strategy can be adapted in DP analysis.

Considering the long term benefits that Defect Prevention methodologies bring to an Organization, these initial bottlenecks on the mindset of the people can be easily broken.

Table of Contents

ABSTRACT:	2
1.0 INTRODUCTION:	5
2.0 DP TECHNIQUES	5
3.0 DP MANAGEMENT	9
4.0 CONCLUSION	12
5.0 DEFINITIONS, ABBREVIATION AND ACRONYMS	12
6.0 REFERENCES	12
BIOGRAPHY OF THE AUTHORS	13

1.0 Introduction:

Eliminating defects in later stages of the Software Development Life-cycle is more complicated, expensive and time-consuming than correcting them in earlier stages. However, avoidance of defects altogether would be even more beneficial to software programs than early detection. Defect Prevention is therefore the best way to optimize the development process costs and to shorten the development cycle time.

This paper presents the different Defect Prevention techniques that, when introduced at all stages of a Software life cycle can reduce the time and cost necessary to develop high quality systems.

The named DP techniques have been experimented across our projects and proven to be effective. Most of the described techniques are standard techniques which can be adapted by any organization.

2.0 DP Techniques

Defect Prevention is the essence of Total Quality Management (TQM), involves analyzing defects encountered in the past and specifying checkpoints and actions to prevent the occurrence of similar defects in the future. This should address the defects in the software as well as loop-holes in the process.

DP activities can also aid in propagating the knowledge of lessons learned between projects.

The following are the different techniques that we have been adapting in our projects, which were proven effective. These are standard techniques which can be exerted by any software industry,

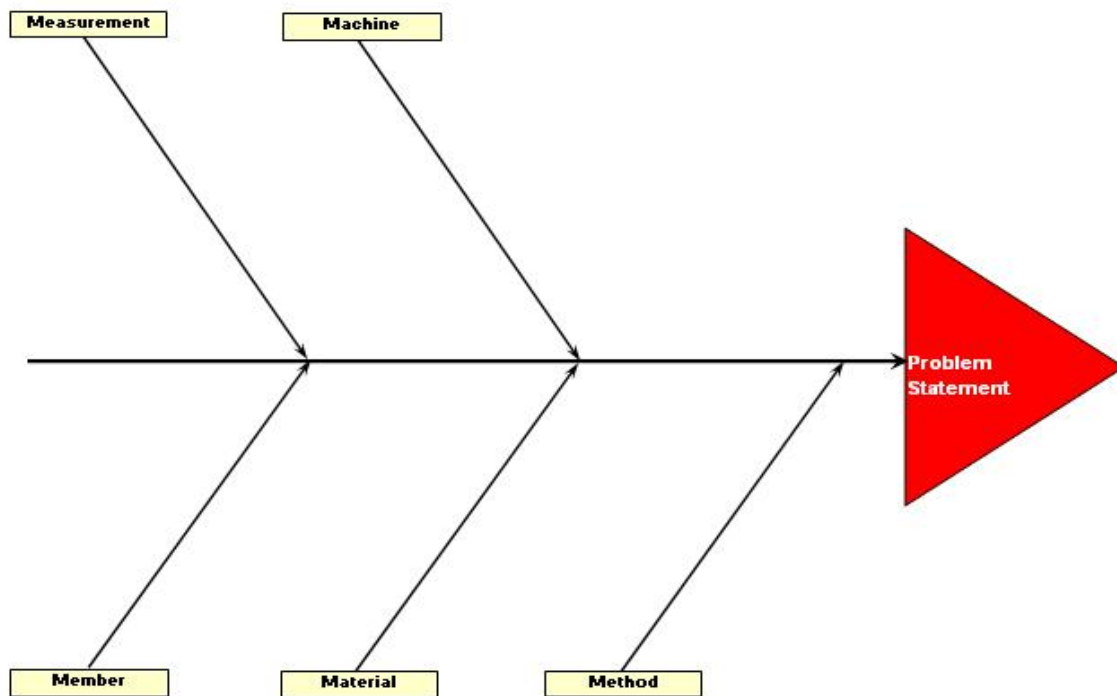
1. Root Cause Analysis (RCA)

Root cause Analysis is the most common and widely used technique to analyze the source of the given problem. In the context of Defect Prediction, RCA can be done for each critical issue that occurs in the software development life cycle thus analyzing various contributors of that particular issue. If RCA points to same set of contributors for multiple issues, then it would be

easy for the project team to address those contributors than trying to fix each defect individually.

These contributors could be competency of the developers, missing/unclear requirements, limitation of the chosen architecture etc. For instance, to fix the architecture issue, it would be better to analyze the alternate designs than to individually spend time on each issue that got introduced due to this. This would guarantee the arrest of the similar defects in future on the program, completely.

Fish Bone diagram is the common way of doing Root Cause Analysis where the given problem is analyzed in terms of Man (people related issues), Machine (infrastructure related issues), Method (Process related issues), Material (Reference artifacts related issues) and Measurement (Review related issues). This also widens the thought process in analyzing the root cause of an issue in all possible means.



At the end of this exercise, the important causes are encircled and definite action items will be derived to address each of them. This would guarantee to eliminate the root causes of the problem and hence minimize the reoccurrence/persistence of the same.

2. Data Mining

When an organization has linear product lines, during the design stage of a new product, project team can perform the data mining exercise on various **Defects database** on Post release defects, **support calls from customers** and **internal defects** to make sure that those get converted as **test cases** and **use cases** in development cycle of the current product under development. This would prevent the re-occurrence of such defects in future products of the organization.

Similarly, within an organization, a repository of test cases of similar products can be maintained in a common server and every time a new project comes, a data mining exercise can be performed to analyze the reuse possibilities of existing test cases, especially the break scenarios. Once these break scenarios are inhaled by development and testing teams, the reoccurrence of such defects can be totally prevented.

3. Orthogonal Defects Classification (ODC)

Orthogonal Defect Classification (ODC) is a concept that enables the quick feedback to developers by extracting the effectiveness of the development process from defects.

Each defect is categorized as any of the following basic types as part of this analysis,

- Function,
- Logic
- Interface
- Checking
- Assignment
- Timing/Serialization
- Build/Package/Merge

It is then mapped to any of the standard triggers based on the phase at which the defect is injected, for instance, a coding defect can be mapped to Simple path, coverage, sequencing, interaction etc.

Through ODC analysis, developers would have an opportunity to look back and see whether he/she has done any repetitive mistakes which he/she needs to put extra effort in the next time from reoccurrence. For example, if a developer does careless mistakes in data initialization, his code may break in boundary conditions. If he/she has the habit of making this

mistake, it could be easily ascertained through this analysis, and can trigger developer in creating self checklist in coding to avoid this mistake again in future.

4. System Modeling and Fast Prototyping

Once the architecture of the system is finalized, the same can be evaluated through any of the modeling tool to observe the performance bottlenecks and capacity limits of the system. Early feedback on the System performance would help the development team in doing the necessary refinement on the basic design, thus preventing the defects from propagating to subsequent phases.

Similarly, fast prototyping would help in getting the feedback about the system from all the stakeholders before the actual development of business logics. This would eliminate any requirement defects that may possibly be present in the project.

5. Use case based development and testing

If the project team does a brainstorming on all the use cases of the system based on the requirements before they do the detailed designing, the defects that occur at the later stage of the project or after the release (post release defects found by customer) can be eliminated substantially, since each of the possible valid workflow can be virtually evaluated during the design stage itself.

Further, System Integrated test cases can also be formed based on these use cases to ensure that System satisfies each of the identified user scenarios. This exercise would also help the project team to question the requirements if they are ambiguity or assumptions in workflow.

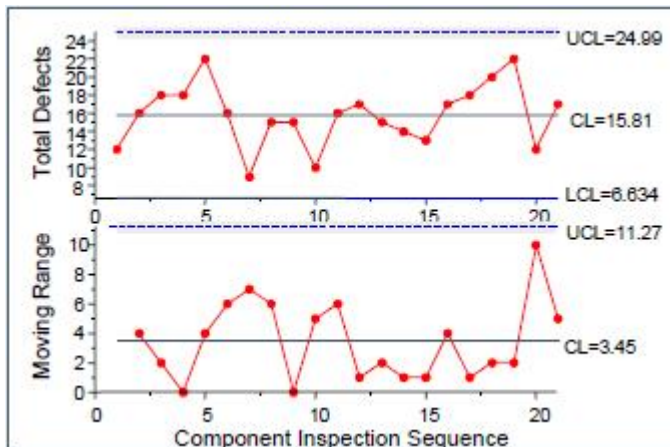
6. Model based development approach

Model based development is appropriate for the organizations that run multiple projects of similar nature. In this approach, the basic functional units which are common across projects are separated out from project portfolio and nurtured as core assets. These core assets are made as generic as possible and tested exhaustively.

Once the core assets are completely ready, each project that requires these assets can just reuse them as they are. This approach would not only reduce the development and testing efforts but also prevent the defects that may occur, if they are developed afresh every time.

6. Statistical Process Control Technique

Usage of control chart would help the project team in deciding whether the performance of the running project is aligned to organization goals or not. This evaluation can be done at any stage of the project, once the required data points are available. These data points can be collected from the projects which succeeded in the past so that the deviations can be easily found and corrected.



Based on the trend, it would also be possible to predict the defects that may occur at the latent stages and appropriate actions can be taken to prevent the same.

The process loopholes can be arrested easily through this technique.

3.0 DP Management

The effectiveness of Defect Prevention techniques can be actualized, only if the right technique that is best suited for the project is chosen and used meticulously. The following are the essential process steps to be carried out decisively to get benefits from Defect Prevention analysis,

- **Data Collection:** Defect data and other problems are collected and the most serious/repetitive defects are targeted.
- **Data Classification and Defect Classification:** Defects and other problems are classified based on Orthogonal Classification of Defect type.
- **Data Analysis:** Defects and other problems are analyzed and root cause for the defects are identified.

- **DP Actions:** Corresponding to the root causes identified, DP actions are formulated. They are then prioritized based on a number of factors such as urgency, criticality, need for resources, changes in hardware/software, cost benefit analysis and so on.
- **Implementation Reporting and follow-up:** The implementation plan is drawn up and the plan is communicated and tracked.
- **Institutionalization:** DP is not complete, unless the lessons learnt are assimilated into organizational learning. Using different mechanisms, the lessons learnt, and benefits are shared across the different projects in the organization.
- **Evaluate Effects of Changes:** Evaluate the effects of changes on process performance and record the data from the DP actions and its institutionalization.

In general, the following activities can be followed in sequence towards Defect Prevention implementation,

Activity	Description
1.	Change Request/Problem Report Generation <ul style="list-style-type: none"> • Independent test team/Customer identifies a defect or a need for a change in a work product. • The change request is documented in the form of a defect and maintained.
2.	Identify lessons learnt from previous phases/Similar Projects <ul style="list-style-type: none"> • As part of Project Initiation/Kick-off, collect the best practices, lessons learnt and actions items for DP planning from the following sources: <ul style="list-style-type: none"> ○ Previous Project Inquiry Reports (of similar projects). ○ Project repositories ○ Previous project Audit reports. ○ Best Practices Sessions.
3.	Prepare DP Plan <ul style="list-style-type: none"> • Every project plan may include the following DP activities: <ul style="list-style-type: none"> ○ Identify and prioritize the Defect/Problem categories to be addressed by the DP activities. ○ Ensure alignment of identified DP actions with the project's

	<p>Scope and CTQs.</p> <ul style="list-style-type: none"> ○ Identify DP actions to be implemented in each phase. ○ Identify Mechanism for tracking defects/DP actions. ○ Communicate the responsibilities and expectations from DP activities to all stakeholders. ○ Phase End analysis/Phase kick-off. ○ Quantitative Measures and overall project/program measures, goals and plans. ○ Audit results analysis meeting with QA. ○ Phase wise lessons learnt and Project Inquiry meetings. ○ Phase kick-off meetings (Closure meeting of the previous phase). ○ Inspection analysis. ○ Root Cause Analysis for defects/Non Conformances. ○ Test Report Analysis. ○ The overall DP plan can be documented and reviewed as part of the project plan. <ul style="list-style-type: none"> ● Project teams identify various problems, analyze the test reports and the defects generated during testing and find the root causes for the defects. Corrective and Preventive actions are initiated and tracked. These are done during team internal reviews. ● The corrective actions and preventive goals for defects found at the personal level too form part of DP Plan for the project. These are tracked I during the team internal reviews. ● Collect measures and analyze them. This is part of score card or phase-kick off meetings or Project Inquiry reporting.
4.	<p>Conduct formal review of Project plan</p> <ul style="list-style-type: none"> ● Conduct formal review of DP Plan (as part of the Project Plan).
5.	<p>Track & Monitor DP actions</p> <ul style="list-style-type: none"> ● The DP activities are tracked as per the DP plan. ● Regular audits to escalate any deviations.
6.	<p>Evaluate effectiveness of DP activities</p> <ul style="list-style-type: none"> ● Evaluate effectiveness of DP activities at planned intervals. ● Changes (if any) must be documented in the DP plan. The same must be

	communicated to all the stake holders.
7.	Final Project Inquiry <ul style="list-style-type: none"> • Perform Project Inquiry as per the guidelines. • Raise Process Change Request, if necessary.
8.	Perform Closure Audit <ul style="list-style-type: none"> • Conduct project closure audit as per the Audit checklist.

4.0 Conclusion

Defect Prevention techniques when used diligently will certainly produce good results for the organization in terms of both Quality and Cost improvements. It needn't to be considered as a process overhead but rather as part of the integral process.

Though the benefits may not be realized instantly, Defect Prevention initiative can help organization in the longer run.

5.0 Definitions, Abbreviation and Acronyms

Acronym	Description
DP	Defect Prediction
RCA	Root cause Analysis
ODC	Orthogonal Defect Classification
TQM	Total Quality Management
KPA	Key Process Area
CMM	Capability Maturity Model
COPQ	Cost of Poor Quality

6.0 References

Author/Type	Source/title
DP in Six Sigma	http://software.isixsigma.com
DP Techniques	http://www.iscn.at
ODC	http://www.research.ibm.com
Mays, R.G., Jone, Holloway	Experiences with defect prevention

Biography of the authors

Author 1:

Franklin Joseph joined Honeywell in the year 2002, and currently working as a Team Leader. He has 6.5 Years of experience in Testing and Software Quality Assurance. In his career, he has been involved in Integration and System level testing of various engineering tools of Honeywell building Solutions domain. He has also been involved in Process Improvement initiatives of Honeywell at different stages. He has experience in System level Performance and Capacity testing as well.

Franklin is also being involved in Training and mentoring initiatives of Honeywell and has trained Honeywell employees in Six Sigma Process and Honeywell Open systems.

Franklin is a Computer Science Engineering graduate from Bharathidasan University and currently pursuing his Masters in Business Administration from ICFAI.

Author 2:

Ekanath Santharam joined Honeywell in the year 2002, and currently working as a Team Leader. He has 6.5 Years of experience in Testing and Software Quality Assurance. In his career, he has been involved in Acceptance and System testing of Video Surveillance product lines of Honeywell building Solutions domain. He is also being involved in test automation initiatives Honeywell Video domain.

Ekanath trained Honeywell employees in Video domain and technologies and Agile Methodologies of Testing.

Ekanath is an Electronics and Communication Engineering graduate from Madurai Kamaraj University and currently pursuing his Masters in Software Systems from BITS, Pilani.