Shanmugakumar Ramasamy
Rohil Ahmed Aliyar Mohammed
Cognizant Technology Solutions,
Chennai, TamilNadu, India.

COMMON DEFECTS IN WEB APPLICATIONS

Abstract

Quality of product/ service is the area where no customer is prepared for a compromise. Effective and efficient test execution, thus, holds the key to earn and maintain customer good will for continued association and manage customer loyalty.

Defect Prone Zones are the areas in the web applications that are prone to the common defects. Security, Miscellaneous data and Unconventional usage of keys etc. are the types of Defect Prone Zones. The functionalities like Security, Miscellaneous data and Unconventional usage of keys should be supported by all the applications irrespective of the domain, criticality and user levels. This may not be listed in the Business Requirements, Use Cases or User Interface Design Documents, but it should be ensured that the web applications are supporting these functionalities. But considering the project pressure, it is difficult to incorporate all the business requirements within the given schedule. Hence incorporating these additional functionalities in the given time is next to impossible. It is the responsibility of the testers to validate these functionalities and ensure that these are available in all the web applications (common defects are not present).

The basics of the Test Maturity Model states that, there is always a scope for improvement. Being an ISO 9001: 2000 certified and SEI CMM Level 5 company, Cognizant Technology Solutions has experimented proactively in evaluating the ways for maturity / improvement. One of the key initiatives taken for improving the test efficiency is the identification of the common defects and testing the same in all the web applications. This has caused a considerable reduction in defect leakage and in turn customer satisfaction.

This paper details the common defects that are present in web applications and also mention the improvement in the test efficiency, with minimum increase in effort.

Author's Biography

Shanmugakumar Ramasamy is a B.E Mechanical Engineering from the College of Engineering Guindy, Anna University, Chennai. He is a Programmer Analyst and has been with Cognizant Technology Solutions since December 2002. As a process owner, he has actively involved in the process definition, implementation and improvement initiatives.

Rohil Ahmed Aliyar Mohammed is a MCA from Karnataka University, Dharwad. He is a Programmer Analyst and has been with Cognizant Technology Solutions since December 2003. As a Defect Prevention Coordinator, he has designed and implemented the Defect Prevention Plan for various projects.

1. Introduction

Testing as performed in most of the organizations is a process designed to compensate for an ineffective software development process. It is risky to develop software and not test it. As long as developmental and maintenance processes that organization utilize continue to introduce defects into software, testing will be very important component of the developmental process.

Testing Cost depend heavily on when in the project life cycle testing occurs. The later in the life cycle testing occurs, the higher the cost. As defects would be identified in the later stage of testing cycle and the cost of a defect is twofold; you pay to identify and to correct it. The cost of defect identification and correction increases exponentially as the project phase progresses. Figure illustrates the cost incurred for eliminating a defect.

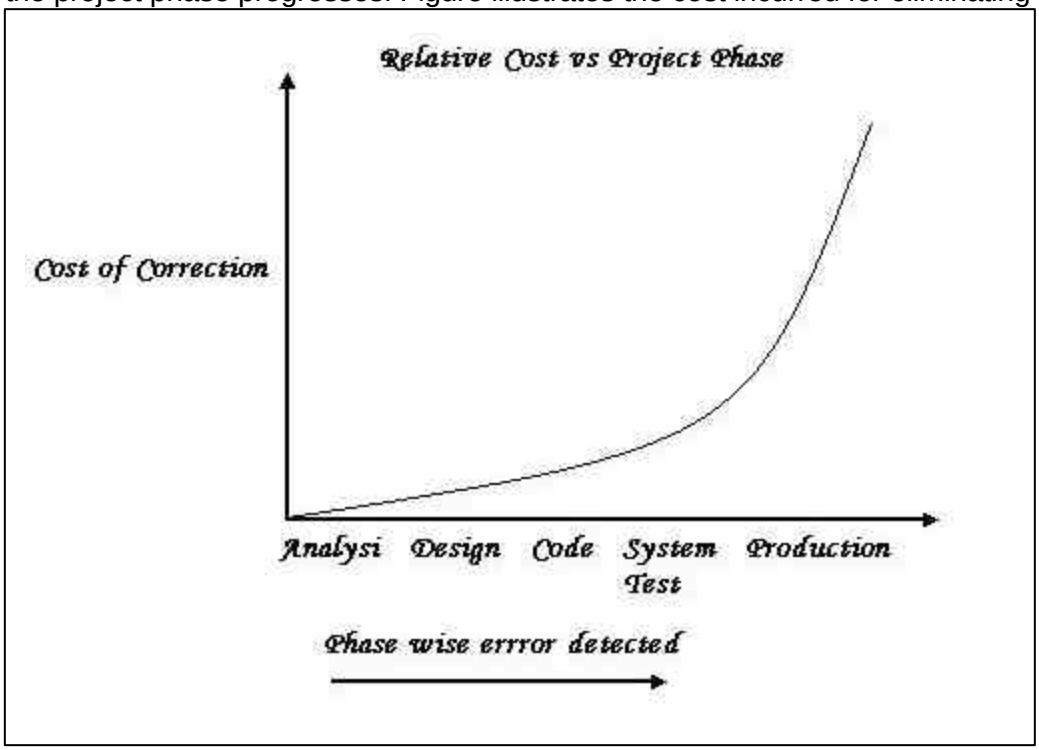

Figure: Shanmugamkumar_1

Apart from eliminating defects which are variance from specification, there are certain defects which we overlook at times, this might not be listed in Business Rules/ Use Case / UIDD, still it should be ensured that our application are supporting these features. But considering the project pressure, the developers are finding it difficult to incorporate all the common standards/ Procedures within the given schedule. Hence they may not be able to incorporate these additional features in the given time. So, we testers should take the initiative of validating them and ensuring the availability of these features (absence of common defects). This document details these common defects that are present in web applications. With the advent of common defect, test efficiency can be improved with minimal effort.

2. Defect Prone Zones

These are the areas in your web application that are prone to common defects.

2.1 Security

This zone in your application is prone to defects related to the “Access Restrictions”.

2.2 Miscellaneous Data

This zone in your application is prone to defects due to the usage of miscellaneous data such as “single quote” in the search criteria field, “enter key”.

2.3 Unconventional usage of keys

This zone in your application is prone to defects due to the usage of keys in an unconventional way.

2.4 Other

This zone in your application is prone to other defects that do not fall under any of the above-mentioned categories.

All the above categorized defects fall under common defects which remain undetected during the design, Coding phase. Below pie chart illustrates categorization of defects and shows that common defects does remain in all application.

Defects	No. Of Defects	% Increase Defects
Detected(Valid)	40	17.5%
Undetected	5	
Common	7	

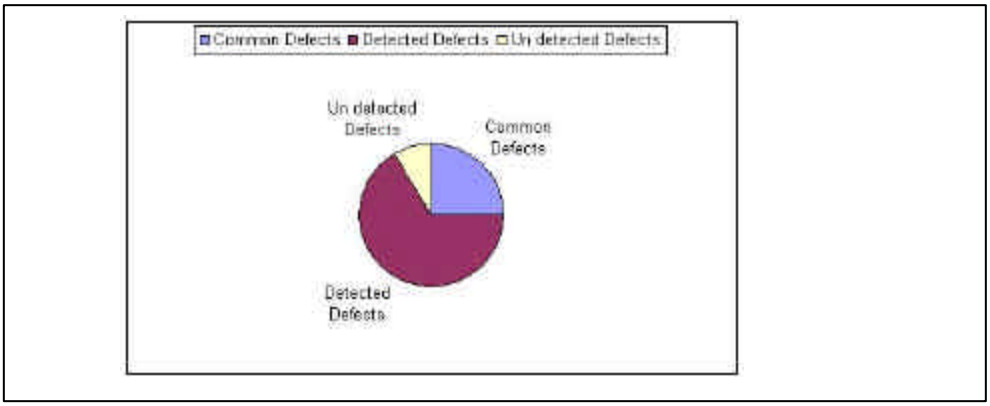

Figure: Shanmugamkumar_2

3. Defects

3.1 Access Restrictions to Screens

Defect Prone Zone - Security

Description :-

Assume a web-based application accessed by two users, one a super user (Have access to "Create User" menu), and the other an ordinary user (Does not have access to "Create User" menu).

Execution Sequence

1. Log onto the application as Super User
2. Click on 'Create User" menu
3. "Create User" screen is displayed with "http://www.test_application.com/ as the URL
4. Copy the URL
5. Log out of the application
6. Log onto the application as Ordinary User in a new browser
7. Home Page is displayed
8. Paste the URL
9. Click on Go button (or) Press Enter key

Comments:-

Your application might display the "Create User" screen. Instead, the message "The user does not have access to this menu" should be displayed in a new screen.

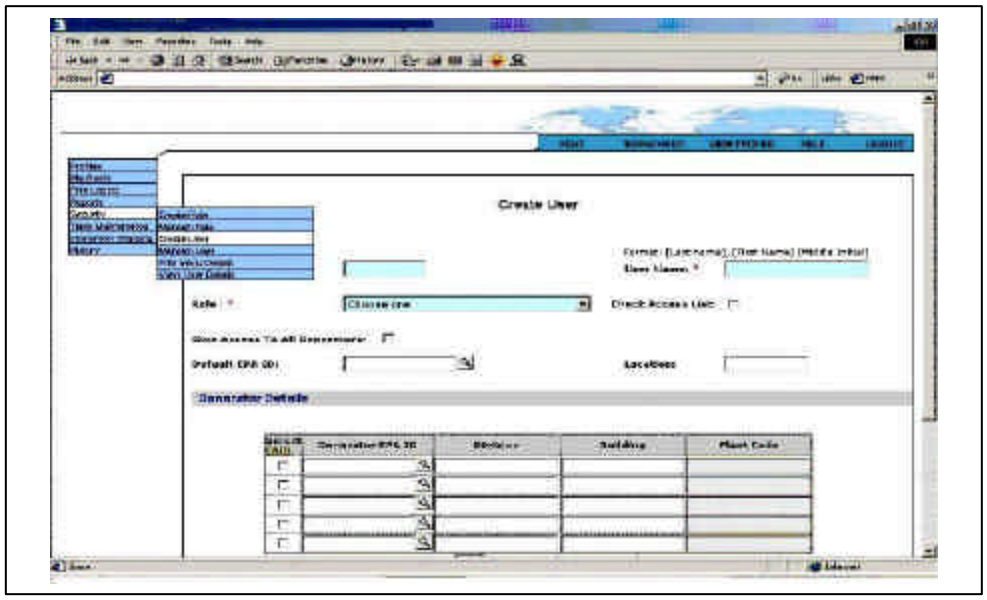

Figure: Shanmugamkumar_3

3.2 Access Restrictions to entities

Defect Prone Zone – Security

Description :-

Assume a web-based application used for tracking projects. All the users can create projects, but the users will be able to view only their projects (created by them). User_1 creates Project_1 & User_2 creates Project_2.

Execution Sequence

1. Log onto the application as User_1
2. Click on "View Project" menu
3. A new screen with a user entry field is displayed
4. If the user enters Project_1 and Clicks on 'Submit' button, he will be allowed to view the project. But if the user enters Project_2 (to which he has no access), an error message will be displayed saying that the user does not have access to view the project.
5. Enter Project_1 and Click on 'Submit' button
6. A report will be displayed with the details of the project.
7. The address bar in the report will be of the format http://test_application.com/display_reports.do? Project_ID=Project_1
8. Edit the Project_ID in the URL to Project_2. (i.e. URL is changed to http://test_application.com/display_reports.do? Project_ID=Project_2)
9. Click on Go button (or) Press Enter key

Comments :-

Your application might display the details of Project_2. Instead, the message "The user does not have access to view this project" should be displayed in a new screen.

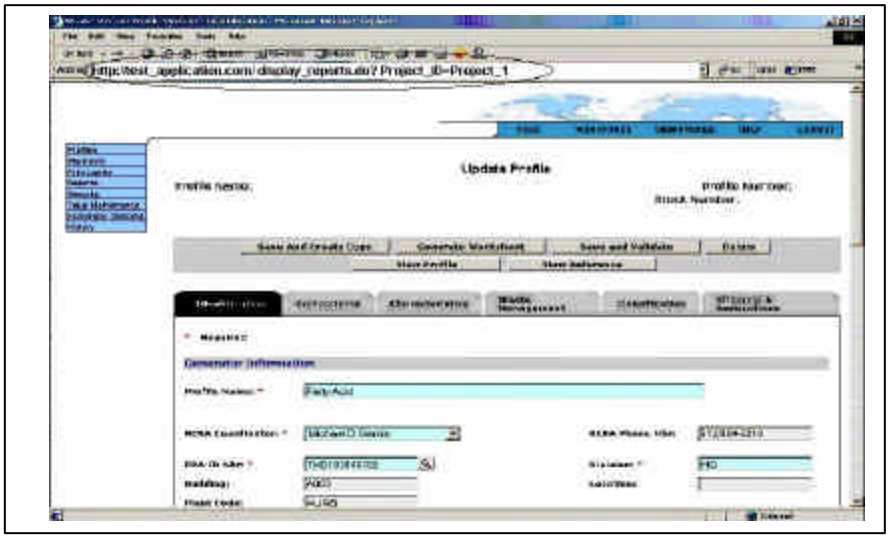

Figure: Shanmugamkumar_4

3.3 Single Quote as Search Criteria

Defect Prone Zone – Miscellaneous Data

Description :-

Assume a web-based application used for viewing user details. A search screen is available allowing the users to search based on User ID and/or User Name.

Execution Sequence :-

1. Log onto the application
2. Click on "Search User" menu
3. A new screen with two user entry fields is displayed
4. Enter single quote as the value of User Name
5. Click on Search button

Comments :-

Your application might go to error page. Instead, the application should either throw an error message stating, “Search Criteria is Invalid” (Client Side validation) or the application should search the DB and then displays a message “No search results found”.

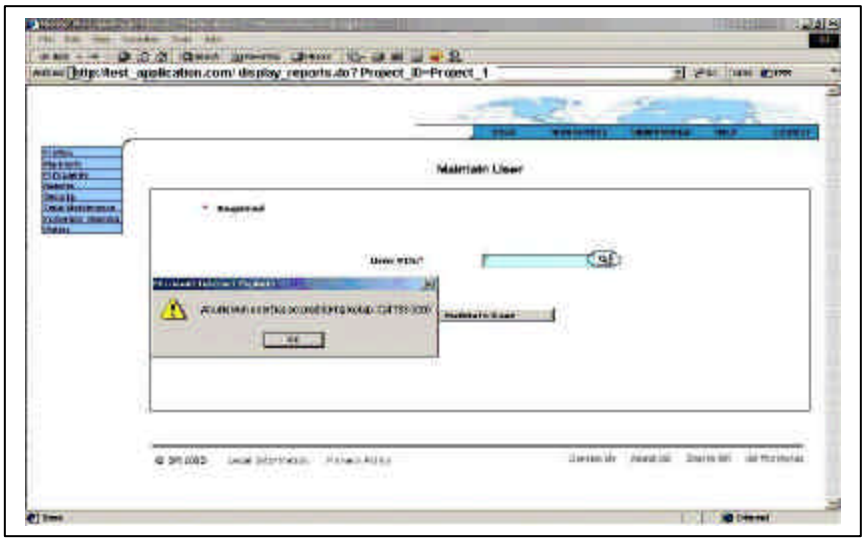

Figure: Shanmugamkumar_5

3.4 Enter Key as a value for Text Area field

Defect Prone Zone – Miscellaneous Data

Description

Assume a web-based application used for managing users. A screen for creating a user is available containing text area as one of the mandatory fields. While creating a user, if the text area is left blank, the application throws an error message.

Execution Sequence

1. Log onto the application
2. Click on "Create User" menu
3. Enter values in all the fields except for the text area field
4. Click on Create User button
5. Application throws an error message asking the user to enter value in the text area field
6. Click on text area

7. Press Enter Key so that the cursor moves to the second line of the text area
8. Click on Create User button

Comments
Your application might create the user with “null” as the value for the text area field. Instead it should throw an error message asking the user to enter value in the text area field

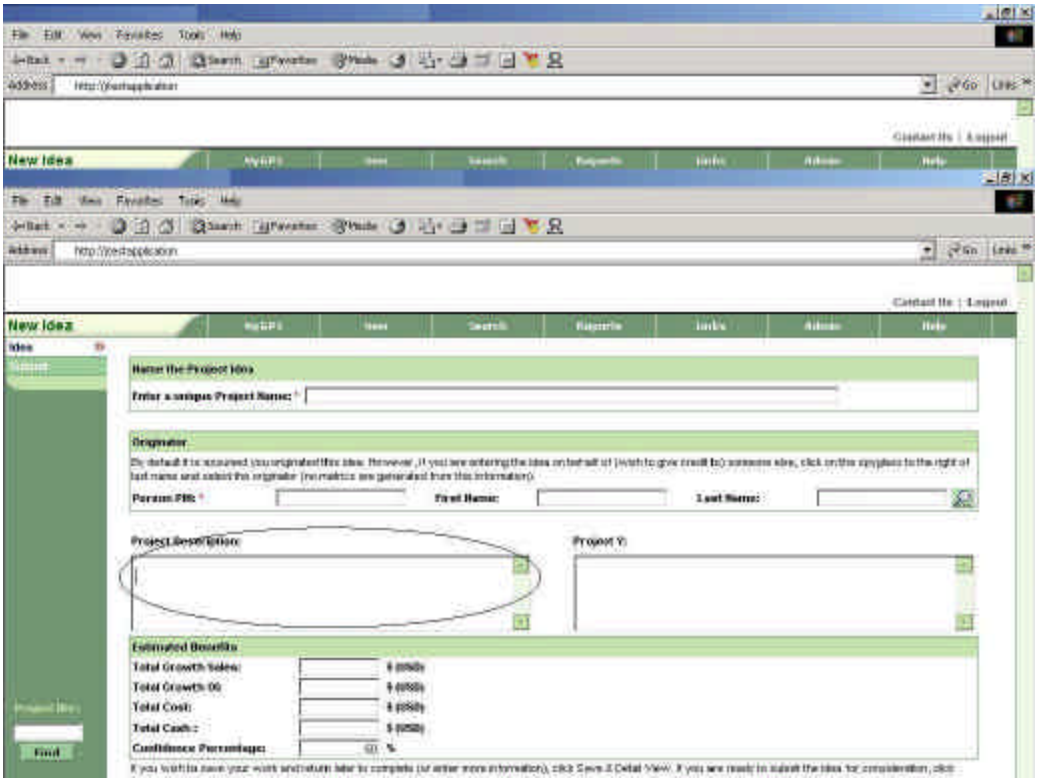

Figure : Shanmugamkumar_6

3.5 Pasting invalid data in a text box

Defect Prone Zone – Unconventional usage of keys
Description
Assume a web-based application used for managing products. A screen for searching products is available with Product Name and Product Cost as the two search criteria fields. Product Cost field has a restriction of allowing the user to enter only numbers and a single decimal. Even if the user types any other character in the text box, it will not be entered / displayed in the text box.
Execution Sequence

1. Log onto the application
2. Click on "Search User" menu
3. Search user screen is displayed with two fields, Product Cost and Product Name
4. Enter “ABC123.25” in the Product Cost field
5. “123.25” is displayed in the Product Cost field
6. Clear the contents of the Product Cost field
7. Have the text “ABC123.25” in the clipboard by entering the text in a notepad and copying it.

8. Set the focus onto the Product Cost field
9. Press “Ctrl + V” or right click and select the option “Paste”

Comments

Your application might display the value “ABC123.25” in the text box. Instead it should display “123.25”

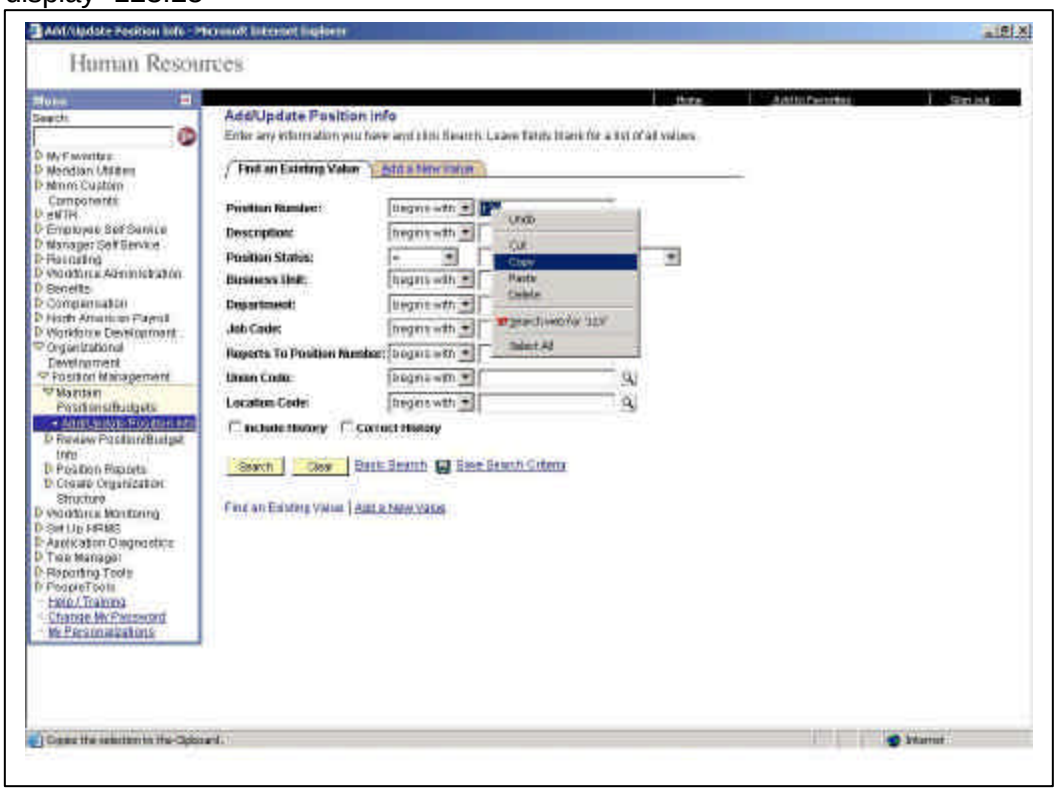

Figure : Shanmugamkumar_7

3.6 Pressing Enter key instead of Submit button

Defect Prone Zone – Unconventional usage of keys

Description

Assume a web-based application used for viewing user details. A search screen is available allowing the users to search based on User ID. User ID field has a restriction of allowing the user to enter only six digit numbers.

Execution Sequence

Log onto the application

1. Click on "Search User" menu
2. A new screen with a single user entry field is displayed
3. Enter “001” as the value for User ID field
4. Click on Search button
5. “Enter a six digit number” message is displayed and the application will not search the DB (Client side validation)
6. Click Ok for the error message
7. Press Enter key (which is mapped onto the Search button)

Comments

Your application might search the DB and display “No search results found” message in the search results frame. Instead it should display the message “Enter a six digit number”.

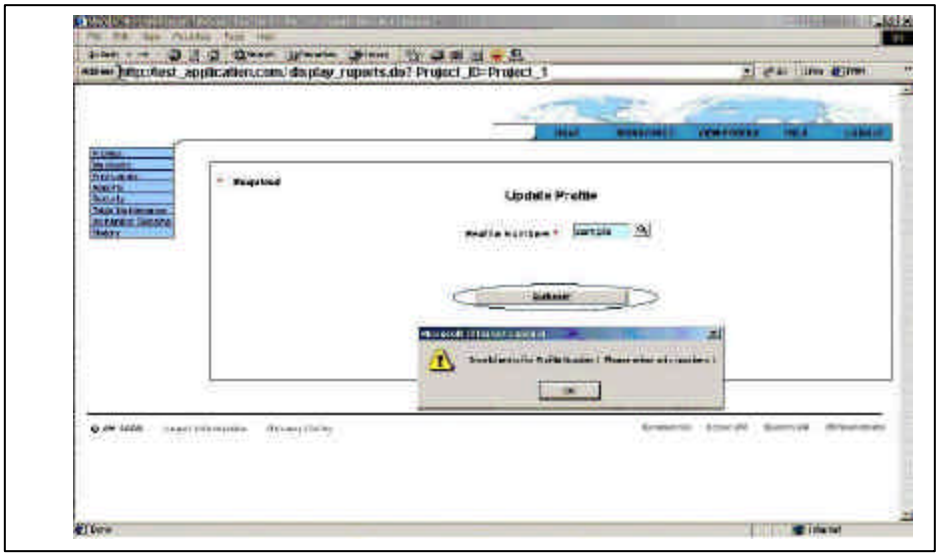

Figure : Shanmugamkumar_8

3.7 Attaching a document that is kept open

Defect Prone Zone – Other
Description
Assume a web-based application used for applying for a job. A screen is available allowing the users to submit their resume which allows only excel sheets and word documents to be submitted.
Execution Sequence
1. Log onto the application
2. Click on "Submit Resume" menu
3. A new screen with “Browse” and “Submit Resume” push buttons is displayed
4. Click on Browse button and select a pdf document
5. Click on Submit Resume button
6. The application will display a message stating that only word documents and excel sheets can be submitted
7. Click on Browse button and select an excel or a word document that is already kept open
8. Click on Submit Resume button

Comments
Your application might display a message stating that only word documents and excel sheets can be submitted. Instead it should accept the submitted document.

4. Benefits of implementation

The variation in the value of test efficiency, effort and defect density before and after the implementation of Common Defects (from the table) indicates the benefits of “Common Defects”

Time	Test Efficiency	Effort	% of Valid Defects
Before Implementation	78.1	50	7.1%
After Implementation	87.2	52	25.1%

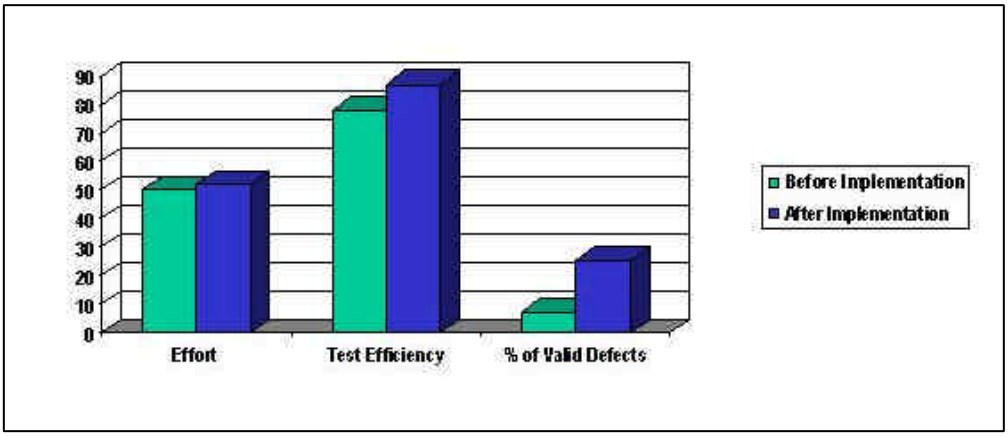

Figure : Shanmugamkumar_9

1. Number of defects found during test execution phase will be more, there by increasing the percentage of valid defects by approximately 8%
2. Less defects in production hence low cost.
3. By capturing these common defects it becomes ease of use for end users
4. Reduces the count for number of defects undetected.

6. Potential Areas of Re-Use: -

We can re-use these common defects in the following areas in future.

1. Mainframe application testing
2. Client-Server Application
3. Localization testing
4. Globalization testing
5. Product testing

7. Learnings: -

1. Cost associated with project reduces to an extent of about 25**,** when these types of defects are found during test execution phase.
2. Clients appreciated this work, as we were able to deliver product with minimal defect, hence improved the quality of product.
3. Learned from common defects, defects need not creep only due to variance of specifications, it can be any of the common defects.

8. Number of Defects

Increase in the number of defect found during test execution phase increases considerably with advent of common defects. The below Graph shows the increase in number of defects before and after common defect.

Time	No. Of Defec
Before Implementation	40
After Implementation	50

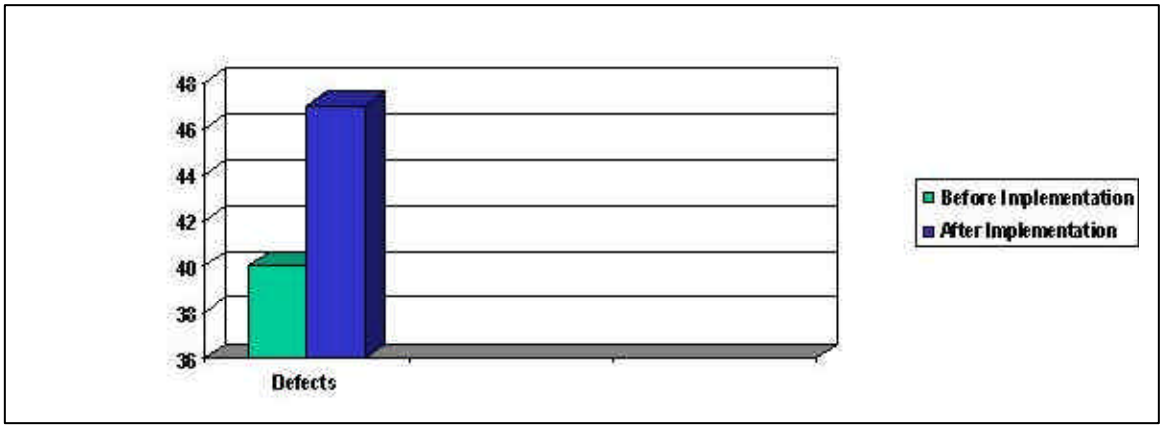

Figure : Shanmugamkumar_10

9. Conclusion:

This presentation concludes that significant amount of effort spent can be reduced by just uncovering common defects across the applications. This approach doesn't leave any room for any known bugs to creep into the system during production. Defects can be found even without executing test cases and doesn't even require much experience and can be done before starting formal execution of test cases. In future we are going to add new common defects to the existing repository and to make it platform independent as well.