Requirement Defect Identification: An Early Stage Perspective

Sandeep Kumar Nayak[1], Raees Ahmad Khan[2] and Md. Rizwan Beg[3]

[1] Department of Computer Science and Application , Integral University
Lucknow, Uttar Pradesh-226026, India

[2] Department of Information Technology, Babasaheb Bhimrao Ambedkar University
(A Central University)
Lucknow, Uttar Pradesh-226025, India

[3] Department of Computer Science and Engineering, Integral University
Lucknow, Uttar Pradesh-226026, India

Abstract

Delivery of reliable software has become a primary concern for the successful software development organizations. Successful and reliable software can be delivering only when the requirement documentation is reliable. There is various threats point in the requirement phase that causes for requirement defects and so defect occurring in the further phases of Software Development process. A key aspect of delivering and improving the software reliability it is necessary to be confident that the requirement delivered to the further phases must be reliable. Reliability measurement is the best characteristic of assessing gathered requirement statistics and their respective compiled documentation. A reliable requirement can be produce only after removing or resolving all types of requirement defects. Here we describe an automated requirement defect identification approach through introducing Defect Data Dictionary which is directly accessible by Requirement Inspection Participant (RIP) and Requirement Inspection Method (RIM) for comparative requirement inspection. This paper provides an overview of the automated approach of how the requirement defect are being detected and resolved to achieve Reliable Requirement Specification (RRS) [8]. Here, the assessment of reliability with respect to requirement defect before and after mitigation through Requirement Defect Detection Framework is given. This may help out requirement analyst for producing the Reliable Requirement Specification.

Keywords: *Reliable Requirement Specification, Reliability Assessment, Requirement Defect Identification, Requirement Inspection Participant (RIP) and Requirement Inspection Method (RIM) Requirement Defect, Severity and priority, Decision Table (DT) Defect Mitigation etc.*

1. Introduction

In the early stage Identification of requirements defects can be made systematic and to approach the highly desirable goal of reliability. The process of paraphrasing the natural language statements to compiled requirement document, definite types of defects may crop up and at the same time supplementary defects may be detected during requirements integration. Through formal representation of Requirement Defect Identification framework for individual requirements under five components (RS, IS, OS, RB, FC) it is possible to detect and resolve requirements defects one at a time.

There are several studies conducted by different researchers for producing reliable software through error removal in code lines and software testing. But there are only few researchers who have given time in defect detection and removal in the requirement phase for delivering the reliable requirement specification. Few authors have given four ways to detect defects a) Checklist Based Detection b) Scenario Based Detection c) Perspective Based Detection d) Traceability Based Detection by [1], some authors depend upon “Defect Density” Model and Design Phase Analysis for defect detection [2], some emphasis on classify the defect similarities and their patterns[3], some researcher narrates to detect, the defects phase wise as a) Elaboration b) Inception c) Construction d) Transition[4] and also detected defect through identification of risk item in the requirement document, establishing relationship between defects and their causes and by recording the requirement defects[5]. Stringent analysis, testing and managing of software reliability should be carried out at the initial stage of System Development Life Cycle (SDLC) [6]. According to Roger S. Pressman and Robert B. Grady the cost and effort incurred in finding and fixing the defects are 1% at requirement phase which is much more less than to fixing at test and deployment phase i.e. 15% and 80% respectively [7]. Both the authors surveyed various industries for elaborating defects they found that more than 50% defects are related to Requirement Phase that means

Requirement Defects = (> 0.5) * Total No. of Defects [7]

2. Proposed Framework for Requirement Defect Detection (D3 Tool)

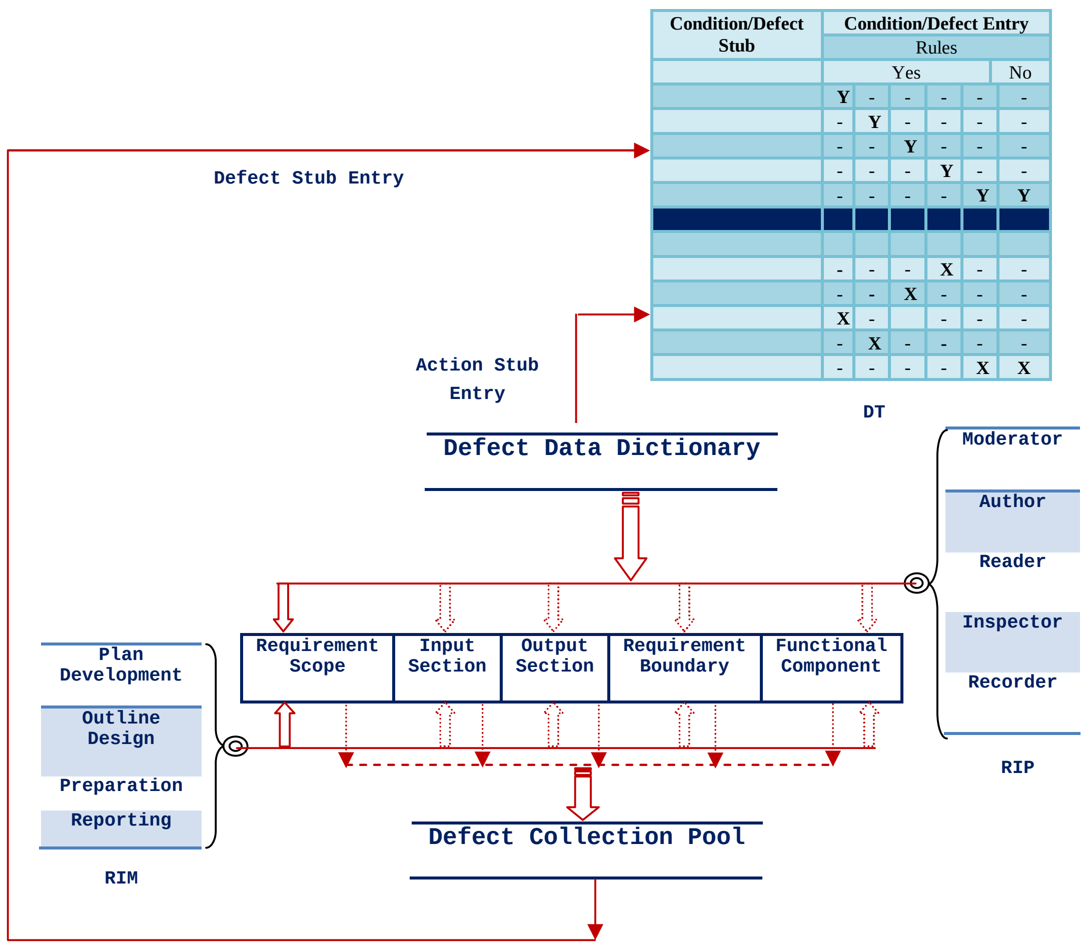
Software defects are the basic reason for malfunctioning and software failure which imposes a direct impact on software reliability [9]. So the defects must take care of from the starting point of software development process.

The proposed framework (Fig 1) comprises a) five classified requirement b) Requirement Inspection Participants (RIP) and Requirement Inspection Method (RIM) c) Defect Data Dictionary d) Defect Collection Pool e) Decision Tree (DT). All the components are processing simultaneously for identifying the requirement defect and stores in Defect Collection Pool.

2.1 Requirement Classification

Initial Requirement is the requirement collected at earliest stage of the software development. After processing the initially collected requirement placed under several specific head [10].

Requirement Scope (Rs): Requirement Scope is directly proportional to customer's objective and responsible to manage statements coming from user's mouth in the form of requirements to deliver agreeable proposed system.



Proposed Framework to Requirement Defect Identification

Fig. 1

It tells about problem domain attribute to draw a sketch for inclusion or exclusion of operational task in new system. Requirement Scope may include four major activities such as 1) Requirement Collection 2) Scope Definition 3) Requirement classification 4) Scope Verification.
Input Section (IS): Input Section is a repository of all type of input component with their definition in data dictionary provided by the customer for the expected output of the proposed system.
Output Section (OS): Output Section directs the concrete, quantifiable and auditable deliverables with their proper expected definition for the proposed system. Requirements of output section are fully dependent on Input Requirement.
Requirement Boundaries (RB): Requirement boundaries are assessable and auditable characteristics in terms of expected outcome for the proposed software. It draws a frame for limiting the input, output and other requirements.
Functional Components (FC): Functional requirements narrated that the required acts which must be perform by the future system. Development of operational component through the analysis of functional requirement and find out coupled requisite measures are involved as an activity. They are associated with specific functions, tasks or behaviors and sometimes known as capabilities or statements of services provided, how the system should react to particular inputs and how the system should behave in particular situations.

2.2 Requirement Inspection Participant (RIP) and Requirement Inspection Method (RIM)

In Requirement Inspection Technique five participants (Table2) used to play vital role through executing their individual responsibilities and they are responsible to follow an appropriate method (Table 1) assigned to them for inspecting the requirement document well to identify defects at early phase.

Table 1: RIM

Method	Activities
Plan Development	• Authentication of Requirement Document • Availability of Role Participant • Schedules structuring
Outline Design	• Task classification among participants • Provide Requirement document for inspection • Inspection meeting and Defect Registration
Preparation	• Technical participants must be instructed for separate learning of requirement document and to find potential defects through review process.
Reporting	• Acceptance on identified defects • Defect Classification

Table 2: RIP

Participants	Roles and Responsibilities
Moderator	• Moderator is responsible for managing overall inspection tasks. • Moderator will plan for Requirement Document Classification. • He will also deliver the proper inspection process schedule. • Moderator will collect all relevant requirement data. • He will also be responsible for issuing Requirement Inspection Report.
Author	• Author is responsible for generating the Requirement Inspection criteria. • Author will provide the Requirement Description for the proper inspection. • Author will also justify the participants role for according to the given inspection criteria.
Reader	• Reader is the leading participant during inspection meet for requirement object revision. • Reader will collect all interpreted sections of the objects for inspector. • Through collecting all objects Reader will emphasize each vital fact for defect identification.
Inspector	• Inspector is responsible for introducing all the requirement objects and identified the defects. • Inspector will frame question for inspection.
Recorder	• Recorder is responsible for collecting all type of Requirement Defects. • He will also deliver the details of Requirement Document. • He will provide proper Decision support for identified defects and recommendations. • He will also collect all inspected defect and requirement residue.

2.3 Defect Collection Pool

Requirement defects must be contained in a tabular form within the database called defect collection pool which may follow a template of specific attributes (Table 3) such as:-

Table 3: Defect Collection Template

Defect Position	Defect Indicator	Defect Cause	Technical Name	Unique Identifier	Defect Definition
Requirement Scope	Functional Actor Missing	Incorrect portrayal for Product	Ambiguous Information	RSD01	Detail description of product mismatched with the actual one.

2.4 Defect Data Dictionary

Defect Data Dictionary (Fig 2) contains the maximum possible type of expected requirement defect in *Requirement Defect Definition Database* and their proper way out as solution pool in *Defect Mitigation Variables* under each classified requirement. The specimen of Requirement Defect Definition (Table 4) and Defect

Mitigation Variables (Table 5) is mentioned here for details.

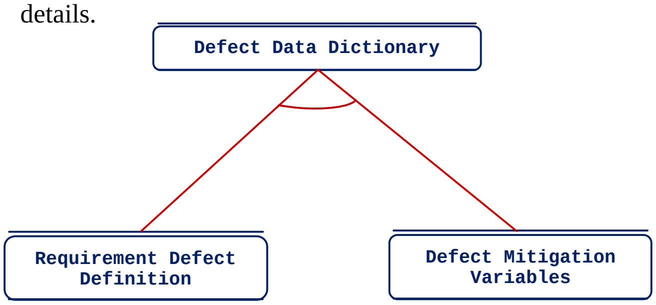


Fig 2: Vital Components of Defect Data

Table 4: Requirement Defect Definition

DEFECT IN REQUIREMENT SCOPE

Identify and Characterized attributes of Defect	
Technical name:	**Ambiguous Information**
Technical Sub name:	Incorrect portrayal for Product
Unique Identifier:	RSD01
Data Type:	String
Definition:	Detail description of product mismatched with the actual one.

Table 5: Defect Mitigation Variables

Unique Identifier	**Mitigation Variables**
RSD01	Redefinition of Product details
RSD02	Inject clear & complete product needs, goals, objectives
ISD01	Creation of input data repository
ISD02	Inject agreeable & compatible naming for input data
OSD01	Give details of output data storage place
OSD02	Inject agreeable & compatible naming for output data
RBD01	Provide specific environment requirement limitation
FCD01	Name of operational actor, its roles & responsibilities must deliver

2.5 Algorithm

Definitions of Inspection Algorithm

Def 1: for Inspection Technique InTech = {E, P, R, D, d, f}

E: Processes involve during Inspection for set of Requirement R, Where $E = \{E_1, E_2, E_3, E_4\}$

P: Participants involve during Inspection for set of Requirement R, where $P = \{P_1, P_2, P_3, P_4, P_5\}$

R: non-empty set of requirements $\{R_1, R_2, R_3, R_4, R_5\}$, such that $\{r_1, r_2, r_3, \text{-----------}, r_n\} \in R_i$ where i = 1 to 5

D: Predefined linear set of defect definition data dictionary

$$D = \bigcup_{i=1}^{m=5} D_i$$

d: Defect Collection Pool containing the identified defect with their unique identification number.

f: such that, $f : R \times D \rightarrow d$ is requirement defect matching function, which shows the matched defect in Requirement R w.r.t. defect definition given in Data Dictionary D, where

If $r_i \in R$ and $x_j \in D$

Then $f(r_i, x_j) \in d$

Def 2: Given an Inspection Technique,

InTech = {E, P, R, D, d, f}

∃ Pointer pointer1 such that,

Int *pointer1;

Pointer1 = &R[i] where i = 0 to 4

Def 3: Given an Inspection Technique,

InTech = {E, P, R, D, d, f}

∃ Pointer pointer2 such that,

Int *pointer2;

Pointer2 = &R[j] where j = 0 to 4

Algorithm:

An Algorithm for requirement defect identification with the help of Defect Data Dictionary based on Inspection Technique called Triple-D Inspection tool.

Input:

Given an Inspection Technique InTech = {E, P, R, D, d, f}, where

$E = \{E_1, E_2, E_3, E_4\}$ is the Inspection Process

$P = \{P_1, P_2, P_3, P_4, P_5\}$ is the Inspection Participants

R = non-empty set of requirements $\{R_1, R_2, R_3, R_4, R_5\}$

D = Predefined linear set of defect definition data dictionary

Output:

Identify the number of Requirement Defect in Requirement Domain R.

Begin:

Step 1: Set the int Pointer, according to Definition 2 & 3

pointer1 = &R [0]

pointer2 = &R [0]

Step 2:

Compute the entries of Requirement domain R such as, by Definition 1

$$R = \bigcup_{i=1}^{m=5} R_i$$

Step 3:

Compute the pointer shift,

Step 3.1:

For, ∀ R [i] ∈ R and For i = 0; i ≤ 4; i++;

pointer1 = pointer1++;

Step 3.2:

For, ∀ R [j] ∈ R and For j = 0; j ≤ 4; i++;

pointer2 = pointer2++;

Step 4:

Compute the matching of requirement domain R and Defect Data Dictionary D,

Step 4.1:

For, ∀ R [i] ∈ R and For i = 0; i ≤ 4; i++;
For, ∀ D [j] ∈ D and For j = 0; j ≤ n; i++;
Let equal R. keywords [i] = match D. keywords [j];
Such that, ∀ r_i ∈ R [i] & ∀ x_j ∈ D [j];
∃ f (r_i, x_j) → True ∈ d;
∃ f (r_i, x_j) ↛ False ∈ d; by Definition 1

Step 5:

for each requirement defect in defect collection pool d, computing the count of defects, as
count = 0;
For k = 0; k ≤ 4; k++;
count = count + d. count [k]; //count the defects of d;

Step 6:

∀ r_i ∈ R [i] & ∀ x_j ∈ D [j];
If f (r_i, x_j) ↛ Φ;
Then return to step 4;

Step 7:

According to Definition 1 the Requirement Defect collected in Defect Collection Pool d;

End

2.6 Decision Table (DT)

Decision Table contains two quadrants of conditions, one is for Requirement Defect and one is for Action Strategy whereas two other quadrants have their respective entries depending upon the rule satisfaction. The decision table quadrants are:-

Condition Stub (Defect Stub) In the first quadrant statement introduces one or more conditions for requirement defects. These defects may be treated as the factor for taking decisions.

Condition Entry (Defect Entry) In the second quadrant of decision table condition entries are meant for completing the condition statement. The entries may be “Yes”, “No” or “don’t care” depending upon the defect rules.

Action Stub (Solution Stub) In the Third quadrant statement introduces one or more mitigation variables in the form of action strategy for requirement defects (Defect Stub). These action strategies may be treated as the steps to be taken when a certain condition of conditions exists.

Action Entry (Solution Entry) In the fourth quadrant of decision table action entries are meant for completing the action strategy statement. The entries may be “Yes”, “No” or “don’t care” depending upon the action strategy rules.

3. Tool Implementation

A sample requirement of Training Information System (TIS) is taken for assessing the Reliability when the defects are identified through Proposed Framework but not mitigated and after the mitigation of defects through mitigation variables obtained by Mitigation Variable Pool (Table 5). Here, Graph 1 represents the failure behavior of requirement before defect mitigation and after defect mitigation where, Graph 2 represents the Reliability to requirement before defect mitigation and after defect mitigation. Graph 3 represents the comparative analysis of Initial Requirement, Identified Defect, Mitigated Defects and Defect Residue respectively.

Through assessing the reliability of sample requirement before defect mitigation (r=0.636) and after defect mitigation (R=0.764) we observe that there is noticeable difference between two reliabilities (R ~ r = 0.128 or 12.8%) which shows the overall degree of reliability for a sample requirement (Reliability Assessment Graph). Therefore it may say that if we move for subsequent passes then degree of reliability increases so forth the Reliable Requirement Specification be achieved within the given time of span in requirement analysis.

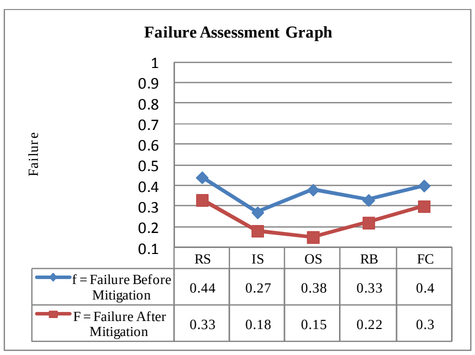
This is the final outcome of the proposed model for Reliable Requirement Analysis. This phase will deliver a Reliable Requirement at the early stage of software development life cycle. Maximum of the Requirement defects which may create problems in structuring the operational parts of the design are removed or fixed for delivering the Reliable Requirement Specification.

4. Conclusion

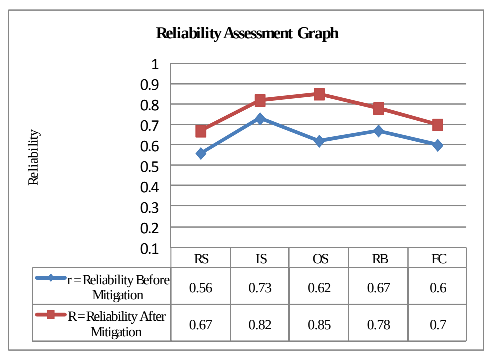
The mentioned illustration under this study shows considerable assurance for defect detection and mitigation as per their severity. Whenever it is finding that the proposed Framework to Requirement Defect Detection by some means unhealthy for identifying a particular defect, Requirement Inspection Participants must review the representation best fit for defect detection. The differences between two successive reliability degrees after defect mitigation may be minimized through proper & concrete introduction of mitigation variables and their implementation. In some cases it may be needed to introduce some additional defect classification, augment our detection processes or sometimes even recommend supplementary mitigation variables in Defect Data Dictionary with respect to specific defect mitigation.

Table 5: Tool Implementation for Reliability Assessment

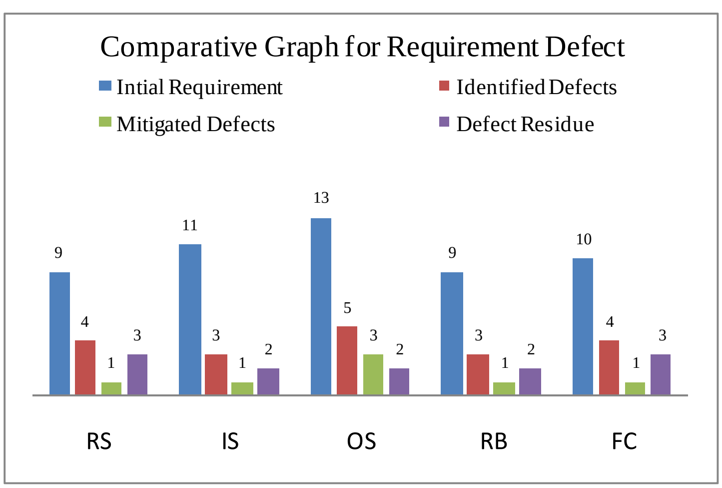
Initial Requirement	Requirement Classification	Inspection Technique	DFR (i)	RD (i)	Fault Requirement Ratio	RReq (r) before Mitigation	W_{max}	DMP	NMD_i	Failure Assessment	RReq (R) after Mitigation
Training Information System (TIS) consists 52 different Types of requirement	RS:N1 =09	InspTech (RS)	DFR_1 = 05	RD_1 = 04	f_1 = 4/9 f_1 = 0.44	r1 = 1- f_1 r1 = 0.56	S1 S3 S2 S4	P1 P3 P2 P4	NMD_1 = 03	F_1 = 3/9 F_1 = 0.33	R1 = 1 – F_1 R1 = 0.67
	IS:N2= 11	InspTech (IS)	DFR_2 = 08	RD_2 = 03	f_2 = 3/11 f_2 = 0.27	r2 = 1- f_2 r2 = 0.73	S1 S2 S1	P1 P1 P4	NMD_2 = 02	F_2 = 2/11 F_2 = 0.18	R2 = 1 – F_2 R2 = 0.82
	OS:N3 =13	InspTech (OS)	DFR_3 = 08	RD_3 = 05	f_3 = 5/13 f_3 = 0.38	r3 = 1- f_3 r3 = 0.62	S3 S4 S1 S2 S2	P3 P4 P1 P2 P1	NMD_3 = 02	F_3 = 2/13 F_3 = 0.15	R3 = 1 – F_3 R3 = 0.85
	RB:N4 =09	InspTech (RB)	DFR_4 = 06	RD_4 = 03	f_4 = 3/9 f_4 = 0.33	r4 = 1- f_4 r4 = 0.67	S1 S3 S2	P4 P3 P2	NMD_4 = 02	F_4 = 2/9 F_4 = 0.22	R4 = 1 – F_4 R4 = 0.78
	FC:N5 =10	InspTech (FC)	DFR_5 = 06	RD_5 = 04	f_5 = 4/10 f_5 = 0.40	r5 = 1- f_5 r5 = 0.60	S2 S1 S1 S3	P2 P4 P1 P3	NMD_5 = 03	F_5 = 3/10 F_5 = 0.30	R5 = 1 – F_5 R5 = 0.70
	InReq = 52		DFR = 33	RD = 19	$f = \sum_{i=1}^{m=5} f_i / 5$ f = 1.82/5 f = 0.364	$r = \sum_{i=1}^{m=5} r_i / 5$ r = 3.18/5 r = 0.636			NMD = 12	$F = \sum_{i=1}^{m=5} F_i / 5$ F= 1.18/5 F= 0.236	$R = \sum_{i=1}^{m=5} R_i / 5$ R= 3.82/5 R= 0.764



Graph 1



Graph 2



Graph 3

This framework development may be treated as most significant and advanced requirement defect identification tool in some extent. Earlier periods of research have developed various tools to automate error analysis and test generation for both requirements and design models but few in the requirement phase. This framework may be capable to identify defects or difficulties earlier as possible when they are least expensive to detect, resolve and prevent from impacting downstream software development activities. The major benefits of this framework can be described in two different manners:

1) **Managerial Profit:** Lesser development costs; Rapid Development process; improved software quality; Performance based objective planning; Reliable Software Delivery.

2) **Industrial Improvement:** Better modeling for further development activity; Fewer defects occurrence; Reduced cost and rework; Earlier identification of hidden defects Effort; Reduction in defect detection.

References

[1] Stefan, B., Michael, H. Software Product Improvement with Inspection: A Large Scale Experiment on the Influence Of Inspection Process On Defect Detection In Software Requirement Documents. Proceedings of the 26th Euro micro Conference, IEEE Explore. 2, pp: 262-269, 2000

[2] Singh, L.K., Tripathi, A.K., Vinod, G., Software Reliability Early prediction in Architectural Design Phase: Overview and Limitations. Journal on Software Engineering and Applications. 4, pp: 181-186, 2011

[3] Fuping, Z., Aizhen, C., Xin, T. Study on Software Reliability Design Criteria Based On Defect Patterns. 8th International Conference on Reliability, Maintainability and Safety (ICRMS), IEEE Explore. pp: 723-727, 2009

[4] Mohan, K.K., Verma. A.K., Srividya, A., Rao, G.V., Gedela, R.K. Early Quantitative Software Reliability Prediction Using Petri-nets. Third international Conference on Industrial and Information Systems, IEEE Explore. pp: 1-6 2008

[5] Lee, E.S., Bae, J.M. Design Opportunity Tree for Requirement Management and Software Process Improvement. International Conference on Multimedia and Ubiquitous Engineering, IEEE Explore, pp: 395-400, 2007

[6] Chao, B., Zhu, X.D., Li, Q., Huang, A.C. Reliability Management in Software Requirement Analysis. International Conference on Management of Innovation and Technology, IEEE Explore. pp: 1104-1107, 2006

[7] Pressman, R.S. Risk Mitigation, Monitoring and Management. Software Engineering – A Practitioner's Approach. Edition 5, McGraw-Hill. 156, 2001

[8] Nayak S.K., Khan R.A., Beg R.,"Requirement Defect Identification and Their Mitigation through Severity and Priority", International Conference on Communication and Electronics Information. Thomson ISI. PP: 427-431, 2012

[9] Y.Z. Gong, "On software's defects," Journal of Academy of Armored For Engineering,, Vol.17, No.1, pp.60-63. (2003)

[10] Nayak S.K., Khan R.A., Beg R., "Reliable Requirement Specification: Defect Analysis Perspective", Springer – LNCS in CCIS. pp: 740-751, 2011