DESIGN OF CONTEMPORARY EXPLORATORY TESTING ALGORITHM AND METRICS

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

in

Software Engineering

Submitted By

NISHTHA HOODA

(Roll No. 801231019)

Under the supervision of:

MR VINOD KUMAR BHALLA

Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

June 2014

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "***Contemporary Exploratory Testing Algorithm and Metrics***", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Mr Vinod Kumar Bhalla* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Nishtha

(Nishtha Hooda)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Vinod K Bhalla

(Mr Vinod Kumar Bhalla)

Assistant Professor

Computer Science Department

Countersigned by

Deepak Garg

(Dr. Deepak Garg)

Head

Computer Science and Engineering Department

Thapar University

Patiala

(Dr. S. K. Mohapatra)

Dean (Academic Affairs)

Thapar University

Patiala

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my supervisor, Mr Vinod Kumar Bhalla, Assistant Professor, Computer Science Department, Thapar University, Patiala. The completion of my thesis was difficult for me without his encouragement. His patience and friendliness were of great help. His detailed knowledge in the field of software engineering and great experience in the area of research enabled me to get a better understanding of the study results. I really appreciate his great support.

I also take the opportunity to thank Dr Deepak Garg, Head of Department, Computer Science and Engineering Department, Thapar University, Patiala for providing us with the adequate infrastructure in carrying out the research work.

I also want to express thanks to all the questionnaire respondents of different software companies those shared their experience and immense knowledge to make this research successful.

I would like to thank my parents and friends for their supervision and ever encouraging moral support, which went a long way in successful completion of my thesis.

Above all, I would like to thank the almighty God for his blessings and for driving me with faith, hope and courage in thinnest of the times.

Nishtha Hooda

ABSTRACT

Software testing is an inevitable phase of the software development life cycle. There is numerous software testing techniques available for different levels of testing. The objective of the thesis is to get a broader view of software testing field first with some real time interaction with the experienced testers of software companies and then to research on the technique which is actually used by testers in the software companies.

To achieve the objective, a set of research questions were framed and questionnaire investigation was conducted with ten software companies. A technique "Exploratory Testing" found to be current trend of competitive testing industry to fulfill the demand of best quality product in short time duration. Experienced software testers shared the method of performing this technique and helped us to differentiate it from random testing.

Several research papers are available for this technique but no algorithm is available till now to understand it. It also lacks metrics to measure the performance of exploratory testing. Hence the status, usage and drawbacks of exploratory testing as well as results of questionnaire investigations are reviewed in the thesis. An *"Exploratory Testing"* algorithm is designed as well as demonstrated in the thesis. Useful testing metrics to measure the success of this technique are also defined and their usage is also demonstrated in the thesis.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Software engineering is the area of research which offers systematic and quantifiable techniques for development of required software. It also defines systematic and disciplined methods of operation and maintenance of well engineered software [1, 2, 3]. Software engineering is the field which includes the application of computer science as well mathematics. It also integrates a well defined method of engineering a software starting from requirement gathering to the analysis, design, quality assessment, implementation and testing[3,4]. Although the term software engineering first meant to discuss in the 1968 during NATO Software Engineering Conference to provoke the software crisis but today it has become a wide area of research integrating countless sub areas as shown in the figure 1.1[2].

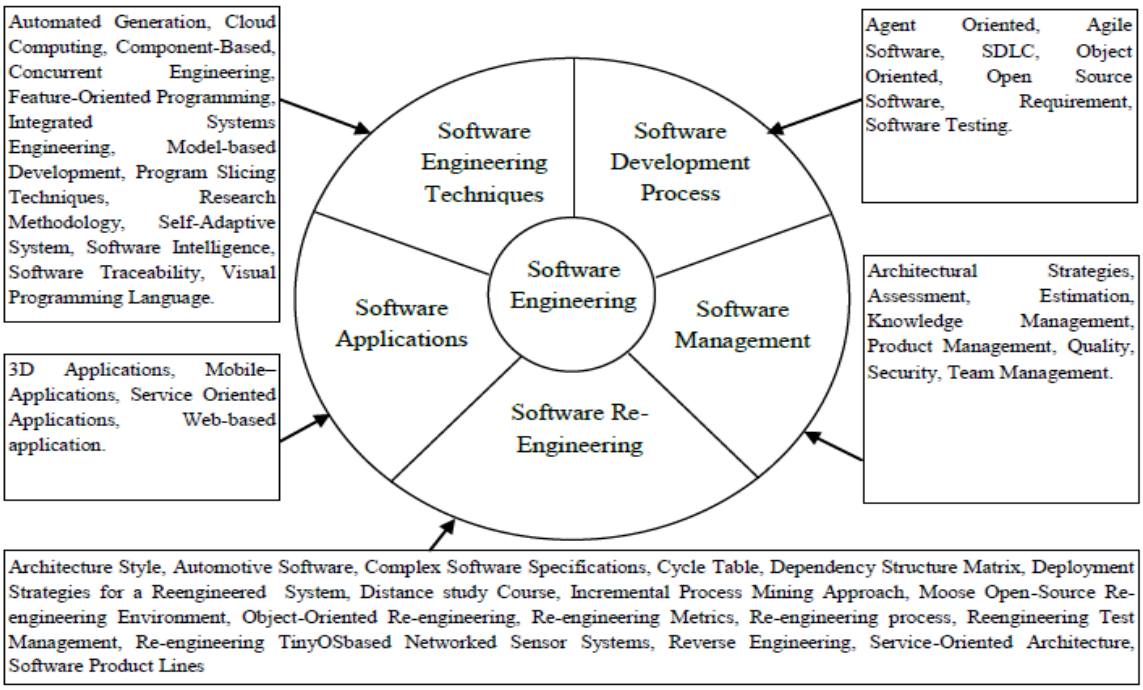

Fig. 1.1 Software engineering as research area [2]

Software testing is about revealing or uncovering bugs from the software is an inevitable phase of software development life cycle [5, 6, 7]. Software testers strive a lot to catch as many bugs as possible before releasing the software. Software testing is also a buzzing area of research in this field [6]. It plays a crucial role in Software Quality Assurance [6, 7, 8]. While doing research on the current trends of software testing techniques, we found the exploratory testing technique is successfully acceptable by software testers in the industry but still lacks acceptance in the mind of researchers due to its comparison with ad hoc testing which is often synonymous of careless work [9, 10, 11].

1.2 MOTIVATION

The best definition of exploratory testing is given by James Bach as "***Exploratory testing is simultaneous learning, test design, and test execution***" [12, 13]. It means the approach eliminates the need of generating planned test cases prior to the test execution. It focuses on exploring the application and giving full freedom to tester to test using their intelligence, creativity without following any predefined and planned documentation [13]. Certainly the technique is opposite to the scripted testing which repeats the same pre-defined test cases every time. So this technique is a new style of testing giving full freedom to every tester to think, learn and test simultaneously. Software testing is considered as cyclic process and generally defined by Software Testing Life Cycle (STLC) [14].

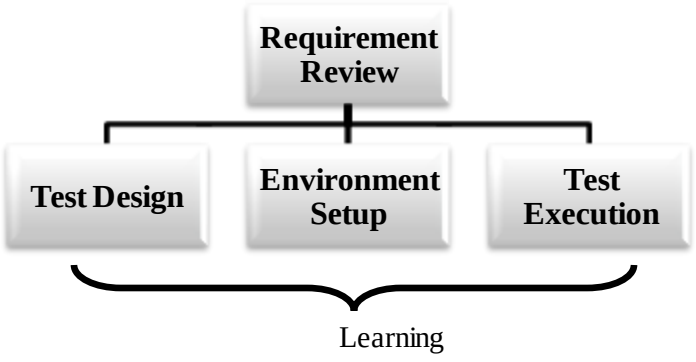

Fig. 1.2 Exploratory testing cycle

STLC defines how the various testing activities are carried out in step by step manner. But exploratory testing does not support the execution of the above phases in this particular order. But the current research shows that industrial software testers prefer to execute multiple phases of STLC in parallel as shown in the figure 1.2. This is considered as an influential approach to test the applications in the real world of software industries where there is always a time pressure of completion. Huge pressure of testing the quality of applications from customer point of view in very short duration leads to the arrival of a new approach called exploratory testing [12,13]. Being a hot topic of research, more research can be done in this area.

1.3 ABOUT THE THESIS

In this thesis, we attempt an in-depth research about the testing technique which is currently used by the software testers. We are interested to work on a technique that is actually used in the software companies so that more work can be contributed in this area as shown in fig 1.3.

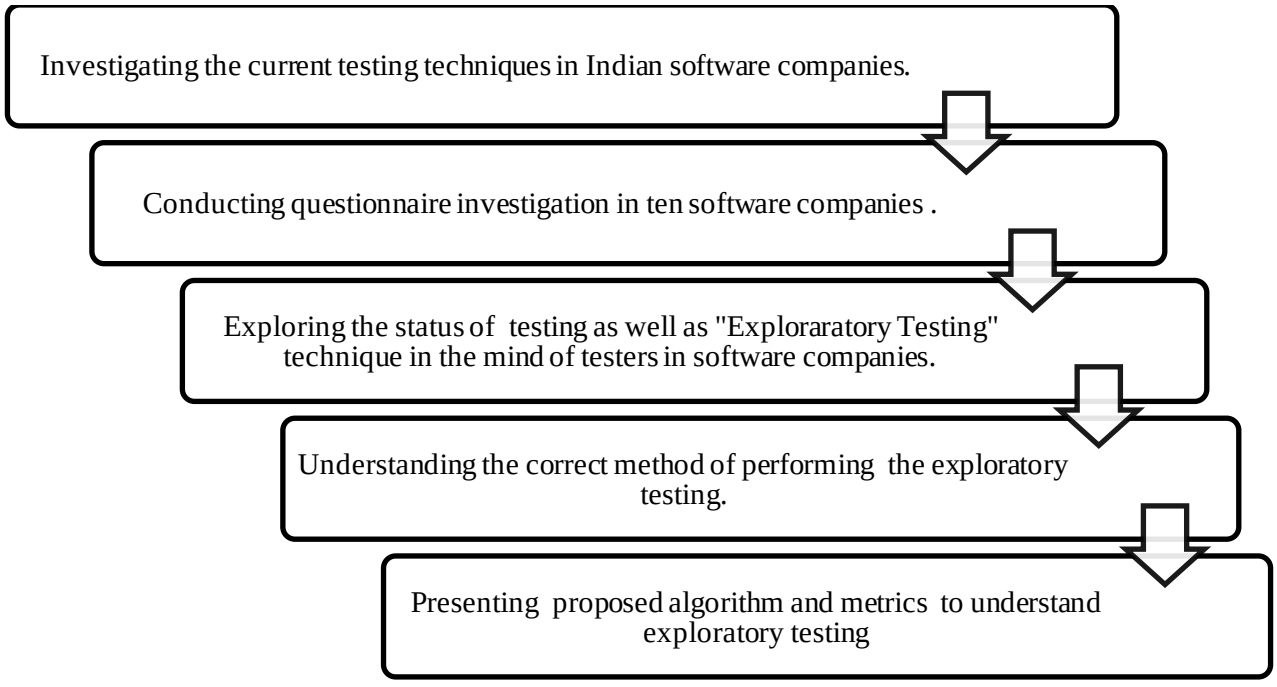

Fig. 1.3 Thesis summary

A small questionnaire investigation is planned to be conducted in ten reputed software companies. The purpose of this investigation is to share the knowledge and experience of software testers in the software companies. The quest is also to explore whether the buzz word of exploratory testing technique or ET in the research paper is actually used by testers or not. We are interested to know the process followed by software testers, status of testing exploratory testing, advantages and drawbacks of using this method. The objective of this thesis is to investigate the characteristics of software projects and the software companies that apply ET in India. Based on the research results, we want to contribute further in this area.

1.4 RESEARCH METHOD

To achieve the objective of thesis, questionnaire investigation of small sample size i.e. ten is framed. It took one month to collect the data from ten software companies. The reason of conducting this small investigation is to get the broader view of the software testing field. As there are numerous testing techniques, but the scope of the thesis is work on the technique which is currently used by software testers in the industry but still lacks the research.

1.5 THESIS OUTLINE

Presenting the main objective of thesis, the first chapter briefly introduces with the work of thesis. Chapter 2, "State of Art with Exploratory Testing" explains the work of different researchers in this area along with the results of questionnaire investigation conducted in ten software companies. Different research questions are framed related to the objective of thesis and finding of different researchers is reviewed in the chapter. Chapter 3, "Problem Statement" explains the problem statement with the help of gap analysis. Chapter 4 "Proposed Solution" explains the proposed solution for it. Chapter 5 "Implementation and Discussion" presents the results and snapshots of the implementation. Chapter 6 shows our further plan for improving the study. At last references and list of publications are attached.

CHAPTER 2

STATE OF ART IN EXPLORATORY TESTING

2.1 INTRODUCTION

This section covers not only the relevant findings of various researchers through their research papers but also gives detailed information about the research done through questionnaire investigation conducted in ten software companies. Literature Review is basically a technical and critical method of reviewing published papers and extracting the findings of different researchers [16]. So this section explains the answers of various research questions so that relevant metadata can be collected and compared for further research work. The research methodology followed is explained in the figure 2.1

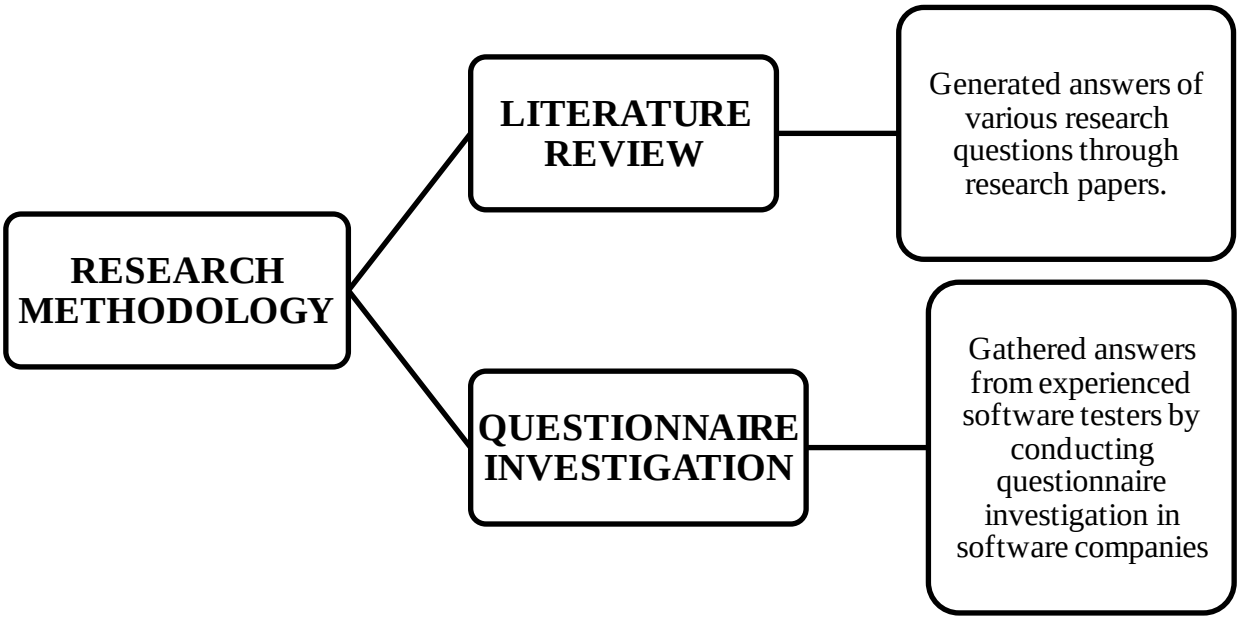

Fig. 2.1 Research methodology

2.2 LITERATURE REVIEW

2.2.1 RESEARCH QUESTIONS

Research questions to review the findings of various researchers are explained in the table 2.1

Table 2.1 Literature review research questions

RQ 1. Does the exploratory testing newly invented technique in the software industry?
RQ 2: Why does it create a misconception with Ad hoc testing and how it is different from ad hoc testing?
RQ 3: Does the approach follow any specific algorithm?
RQ 4: Does the approach contradict other techniques of software testing?
RQ 5: Does the approach depend on skills, knowledge and experience of tester?
RQ 6: Is there any difference found in the defect efficiency of traditional pre test design approach and exploratory testing?
RQ 7: Is it a black box or white box testing technique?
RQ 8: Is the approach acceptability varies from small to large scale companies?
RQ 9: Is the exploratory testing technique domain-specific?
RQ 10: Does the approach offers any metric to assure the confidence in application?
RQ 11: Does the exploratory testing really cost effective technique?

2.2.2 RESEARCH ANSWERS

RQ 1. *Does the exploratory testing newly invented technique in the software industry?*

The technique is not new but introduced by Cam Caner in 1983[13, 14] .But it had not been widely known in research area because it was believed to be just a different term for random or ad hoc testing. Ad hoc testing is always criticized by software engineers as it more a negligent work without any record or documentation. The sloppy nature of ad hoc testing and misconception of complete similarity between ad hoc and exploratory testing is one of the major reasons of why this technique lacks research.

RQ 2: *Why does it create a misconception with Ad hoc testing and how it is different from ad hoc testing?*

Researchers think that there are only two extreme approaches currently used for software testing. They believe that either you can do testing in a complete systematic, planned and formal way or you can just randomly test the software without planning. So the two extreme cases are scripted testing and ad hoc testing. Ad hoc testing sometimes referred as random testing. Although random testing is suggested to applications but ad hoc testing is not considered as good practice as it is not a formal technique. One cannot reproduce results due to the lack of documentation records and designed test cases. But a better technique is needed which is not completely ad hoc. As ad hoc testing is not a systematic way. There is no path and thought of how to go about testing [17,18]. As exploratory testing does not follow any planned predesigned test cases hence it is often confused with random testing. But unlike random testing, exploratory approach record sessions and build test cases during testing. Exploratory testing which neither give privilege to extreme scripted testing which annihilates tester's creativity nor to the extreme random testing which makes the testing process inexplicable after completion [19, 20,21] as shown in fig. 2.2. Scripted testing with the help of various tools is efficient but is not always preferred as too much automation deprives testing process of the creativity and intelligence of testers. Hence we mostly prefer automation tools in the regression testing. Exploratory testing is mid way technique between ad hoc and scripted testing [18, 19].

Fig. 2.2 Exploratory testing status

"Ad hoc Testing" which is chaotic approach in which tester randomly test different modules to check the acceptance of application, exploratory testing is done systematically by introducing testing-sessions to focus specific area of application. These sessions are

generally of 30-40 minutes. It can be extended to an hour depending upon the complexity of goal. Unlike ad hoc testing which randomly choose any part of application without knowing the next step, exploratory testing can be performed very systematically.

RQ 3: *Does the approach follow any specific algorithm?*

The approach does not suggest any specific algorithm. But James Bach and few other practitioners suggest some ways which is formalized here into simple steps. It helps researchers to understand how exploratory testing is actually performed [20]. Exploratory testing consists of two important steps i.e. Scope of session and Journey of exploration. Firstly, testers have to choose the scope of testing the application under test for every session. Scope defines the boundary or area in which tester has to think. Tester has full freedom to apply its creativity but within the defined scope of session. Second step defines the complete journey of exploration within the pre defined scope. Different sessions have different goals, scope and different tours of exploration [20].

RQ 4: *Does the approach contradict other techniques of software testing?*

The approach does not contradict any other testing technique. It can either be used independently to learn and test an application simultaneously or it can be employed partially on the application for risk analysis [22, 23]. It focuses more on non interrupted sessions where testers can even think of other testing techniques of software testing as well like boundary value analysis, decision table to invent more test cases during the session.

RQ 5: *Does the approach depend on skills, knowledge and experience of tester?*

Yes the role of tester in the exploratory testing is highly significant. It can be understood with an analogy of chess game. In chess game, skills and intelligence of players play a crucial role to make the chess game more interesting and entertaining. Rules of chess game may change but what really matters is the skills of players. Similarly rules of exploratory testing may change from one testing team to another but skills of testers remarkably affect the success of testing. This result has already been proved by a field

study conducted with four large organizations with 12 testing sessions in a real time industrial test environment [24]. The study proved that three basic type of knowledge is utilized during exploratory testing. It classifies it into domain knowledge, system knowledge and general software engineering knowledge. This knowledge is materially applied to work on the test oracle during testing sessions which help testers to compare the results of application under test with required expected results. The ultimate goal of exploratory testing is fault recognition which depends how smartly and deliberately the testers are using their skills and knowledge during testing sessions [24, 25].

RQ 6*: Is there any difference found in the defect efficiency of traditional pre test design approach and exploratory testing?*

To answer the above question, a controlled experiment was conducted with 79 advanced software engineering students [22]. The goal was to examine the defect efficiency of two different approaches. No remarkable difference has been found in defect detection efficiency between traditional approach testing approach and exploratory testing. The research proved that the outcome of defects from two different approaches when classified into different category depending upon their technical type, level of difficulty and severity did not show an appreciable difference. The more astonishing result of their research was production of more false defect reports during traditional approach of testing when compared with defect report of exploratory testing. It proves there is no significant profit in predesigned test cases while measuring the testing in terms of defect efficiency [21, 22, 26].

RQ 7: Is it a black box or white box testing technique?

Exploratory testing approach is mainly considered as black box technique because it is used to check the application under test from user point of view without touching the internal source code of the software [22, 23]. The goal is to test the software in short time span without affecting the quality of software. The testers avoid white box testing and focus on black box testing only. The checklist includes all the requirements mentioned in the software requirement specification document.

RQ 8: Does the approach acceptability vary from small to large scale companies?

Any company can use this technique as it does not support any specific size of organization as far as testers have skills to test the application. During the review, we came across a case study where exploratory testing was successfully implemented in Sweden at very large telecom company for three days making the involvement of 80 testers [20]. Another appreciable case study involving three different companies was carried out in Finland [23]. The interview data was collected from seven practitioners from three different companies of different size where size of organization depends on the number of employees. It was concluded that practitioners from different companies even varying in the size were inclined towards exploratory testing approach. Whenever testers come across testing some complex functionality of application, then exploratory testing is preferred to check the quality of application from user point of view [20, 22,23].

RQ 9: Does the exploratory testing technique domain-specific?

The exploratory testing technique is reviewed as a situational practice [20, 22, 23, 27]. It can be applied to test the applications of particular domain like:

i. Applications having low budget and lesser time available for testing.

ii. Application under test has certain complex functionalities whose initial test design is very cumbersome.

iii. Application has to be tested focusing more on the user point of view.

iv. Testing an application where it is also expected from testers to learn more and more about the application by exploration along with finding defects.

vi. Testing applications which require high level of risk analysis and software applications with no deterministic results.

vii. Domain of applications whose defects would not cause harm to human life.

viii. Testing the applications whose software requirement specification (SRS) is not clearly defined or available.

RQ 10: Does the approach offer any metric to assure the confidence in application?

As per our knowledge, no standard test metric has been defined for measuring exploratory testing[23].Although exploratory testing claims good defect efficiency[21,22,26] but management of test coverage during exploratory testing is the biggest short coming of this technique.

RQ 11: Does the exploratory testing really cost effective technique?

Exploratory testing does look very time and cost effective when we have very less time available for testing. Example a website can be tested in very few days saving time and predesigned test cases efforts. But some evidences have also proved that the technique could cause the organization to suffer from technical debt. Technical debt is the bad consequences suffered by organization at later stages due the various shortcuts carried by the employees during software life cycle. It may cause the management of organization to suffer from loss of repairing and patching the product during maintenance stage [27]. Loss may occur in term of cost or loss in reputation. So, the managers and practitioners of organization should be aware of the drawbacks on excessive usage of this technique along with its benefits.

2.3 QUESTIONNAIRE INVESTIGATION

2.3.1 NEED OF INVESTIGATION

Software testing is the process of not just finding errors in the program but also bringing confidence that the software is doing what it is supposed to do [8,9,10]. Many researchers strive a lot to bring knowledge to the students about the practices followed in testing world through training and experiments [7, 9]. A survey was also conducted in Australia by ICT Industry during 2002-2003. 65 software organizations were the active participants of that survey. The goal was to research on the current practices on software testing followed in the different organizations [28]. Similar survey was conducted in software

companies of Norwegian [29]. Another interesting effort was done in Sweden to highlight the contemporary aspect of software testing. According to them, these studies are really useful but very few researchers are actually working in this direction [30]. Such surveys are very beneficial as they make us aware of the real aspects of particular area of knowledge. One such effort is made in the thesis by contacting ten software companies in India and conducting a questionnaire survey. The rationale is to bring the knowledge about the actual software testing methodologies followed in the software companies of India.

2.3.2 DETAILED DESCRIPTION

Data was collected from ten different software companies in India. All companies have different objectives as shown in table 2.2. The various companies are categorized on the basis of Capability Maturity Model (CMM) level and their rank in the Fortune 500 list 2013 as shown in Table 2.3.

Table 2.2 Software companies details

S.No.	Company Name	Goal
1	Accenture	Provides technology solutions and consultancy services
2.	Crestech Software Systems	Provides independent software testing services
3.	Morgan Stanley	American multinational financial services corporation
4	R Systems	Provides software development and maintenance support
5	Sears Holding	Provides application support in retail domain
6	Sapient	Marketing and consulting company
7	Snapdeal	E commerce company provides platform to sell products online
8	Tata Consultancy Services	Indian Multinational I.T services and consultancy company
9.	United Health Group	Health care company
10	Ways2Sms	Product oriented company focus mainly on mobile applications

Table 2.3 CMM level, Fortune 500 status and respondents experience

SNO	COMPANY NAME	CMM LEVEL	PRESENT IN FORTUNE 500 LIST?	TECHNICAL EXPERIENCE OF TESTER(Years)
1	Accenture	5	No	0.5-1.0
2	Crestech Software Systems	3	No	2-3
3	Morgan Stanley	5	Yes	2-3
4	R Systems International Limited	5	No	1-2
5	Sears Holding	3	Yes	1-2
6	Sapient	5	Yes	8-10
7	Snapdeal	-	No	3-4
8	Tata Consultancy Services	5	No	3-4
9	United Health Groups	3	Yes	1-2
10	Ways2sms	-	No	0.5-1

A maturity model can be used as a good touchstone for comparing different organizations. Maturity of different organizations can be compared on the basis of clients they are handling and the type of projects they are working with. It basically measures capability of their software processes [31]. Fortune 500 is the list published annually by Fortune magazine to list the top 500 US corporations on the basis of gross revenue. These two factors are investigated to categorize the ten companies and the various respondents participated in our questionnaire research are chosen with different level of technical experience in their field as explained in the table 2.3.

2.3.3 RESEARCH QUESTIONS AND ANSWERS

This research through questionnaire investigation is conducted in two parts:

i. General Questionnaire: Firstly few questions were asked in order to know the status of

testing in the companies by asking some general research questions (RQ).

ii. Status of Exploratory Testing: Questions were asked to know the status of exploratory testing in the software companies. The research questions were divided into two parts as shown in the figure

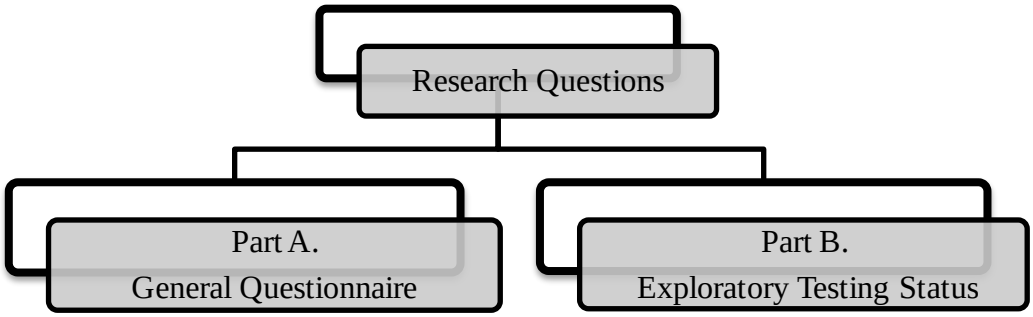

Fig. 2.3 Research questions division

2.3.3.1 General questionnaire

RQ1. *How much important is software testing in the software company?*

Every respondent have same answer to this question i.e. software testing is very crucial part of software development life cycle and cannot be skipped. This question is important to initialize the research work. They all suggested software testing to start as early as the software development starts.

RQ2. *Is there any independent specialized testing team in the company?*

Almost every company has independent testing team in their company and nobody hires third party testers from separate companies as shown in table 2.4. Although few companies don't make separate testing team and any technical person randomly participate in the testing process.

Table 2.4 Need of independent testing team

	Yes, we have independent testing team	No independent team, we all are involved in the testing process
Accenture	✓	
Crestech	✓	
Morgan Stanley	✓	
R Systems		✓
Sears Holding	✓	
Sapient	✓	
Snapdeal	✓	
TCS		✓
United Health Groups	✓	
Ways2sms		✓

Table 2.5 Need of testing training

	Special training is needed on which techniques should be used and how to implement them	No special training is needed. Small overview of product is enough to start the testing work.
Accenture	✓	
Crestech	✓	
Morgan Stanley		
R Systems		✓
Sears Holding	✓	
Sapient	✓	
Snapdeal	✓	
TCS		✓
United Health Groups		✓
Ways2sms	✓	

RQ3. *Are the software testers trained before testing starts in your company?*

Although most of the companies prefer to train and discuss on the testing techniques before testing the projects but surprisingly few CMM level 5 companies don't even bother to give any special training for testing as shown in table 2.5. They believe to test

the software from user point of knowledge without applying any special technical methods of testing.

RQ4. How many testers in the testing team are certified testers?

Hardly any company focuses on updating the knowledge of testers. Even the foundation level certification exams are not passed by the testers in the team as shown in table 2.6. However the result shows that the company like Crestech which is testing centric company prefers to choose certified testers in the team as updating the knowledge in their field is their primary goal.

Table 2.6 Need of testing certification

	Hardly any one.	Yes we have certified testers in the team
Accenture	✓	
Crestech		✓
Morgan Stanley	✓	
R Systems	✓	
Sears Holding	✓	
Sapient		✓
Snapdeal	✓	
TCS	✓	
United Health Groups	✓	
Ways2sms	✓	

RQ5. *Does your company follow any standard software testing model or specific testing approach which is always followed independent of the project?*

Almost every company prefers to follow V model for testing the project. Few companies also suggested scrum model but majority of companies answers with V model only.

RQ6. What is the ratio of usage of black box (testing GUI) to white box testing (testing internal code) techniques in your projects?

The result was surprising as companies don't bother to test the internal source code of freshly developed software as shown in table 2.7. Every company focuses on checking the

portion of software visible to the client. According to them checking every line of code in such a time to delivery pressure is not possible so they prefer to pass the software from customer point of view. Defects if occur due to wrong logic or poor coding after delivery of software are corrected only in the next versions of software.

Table 2.7 Ratio of use of black box to white box testing

	Black Box Testing: White Box Testing
Accenture	70:30
Crestech	70:30
Morgan Stanley	70:30
R Systems	80:20
Sears Holding	90:10
Sapient	90:10
Snapdeal	70:30
TCS	70:30
United Health Groups	80:20
Ways2sms	70:30

RQ 7 *What is the normal ratio of automated to manual testing used in your software projects?*

Manual testing is mostly preferred in most of the companies as shown in table 2.8. Automated testing is only done when task is highly repetitive like in regression testing.

RQ 8 *Do you use any special tools for testing software?*

Companies prefer freeware testing tools. Several tools are reported by various companies. Preferred tools are:

a. Selenium.

b. RFT

c. QC

d. JBehave

e. QTP

f. Opkey

Table 2.8 Ratio of manual to automated testing

S. No.	Company Name	Manual : Automated Testing
1	Accenture	90:10
2	Crestech Software Systems	70:30
3	Morgan Stanley	80:20
4	R Systems International Limited	80:20
5	Sears Holding	70:30
6	Sapient	90:10
7	Snap deal	80:20
8	Tata Consultancy Services	80:20
9	United Health Groups	80:20
10	Ways2sms	80:20

2.3.3.2 Exploratory Testing Status

Exploratory testing is simultaneous learning, test design, and test execution [12, 13]. This approach that terminates the requirement of generating planned test cases prior to the test execution is very hot topic of research in the testing world. The researchers working on exploratory testing had proved that there was no significant profit in predesigned test cases while measuring the testing in terms of defect efficiency [20]. The clear status of this technique is investigated in further research questions.

RQ9. *Have you heard about exploratory testing technique?*

Every company is aware of exploratory testing technique. Few testers are aware of the technique but still not following it due to the pressure of following traditional approach. Some testers recognize it as random testing to test the software in short time span.

RQ10. *Do you think pre designing of test cases and too much documentation is barrier for tester's skills and creativity?*

Most of the testers believe that pre planned test cases and too much documentation restricts their intelligence and creativity as shown in table 2.9.

Table 2.9 Need of pre-designed test cases

	Yes, pre designing test cases is wastage of time and somewhere it restricts the tester's creativity and intelligence	No, conventional testing suggest pre designing of test cases. It is not wastage of time.
Accenture	✓	
Crestech	✓	
Morgan Stanley	✓	
R Systems	✓	
Sears Holding	✓	
Sapient		✓
Snapdeal	✓	
TCS	✓	
United Health Groups		✓
Ways2sms	✓	

RQ11. *Do you actually reuse the test cases of your past projects or refer them for your current projects?*

60 percent of the companies maintain records of test cases so that they can be reused later for future projects as shown in table 2.10. Remaining companies prefer random testing.

RQ12. *How much level of percentage of random testing (testing software without following any pre designed test cases or documentation) is preferred in your projects?*

To this question we get varied answers as shown in table 2.11. The deviation is due to the type of projects they are handling. If we relate the answers of previous questions with the current one, we have observed that the companies which may get similar projects avoid random testing and maintain full documentation of every test case they are executing. The reason is that it can be reused for the future similar projects. But the companies whose

projects don't give them scope to reuse the test cases prefer random testing of software from user point of view.

Table 2.10 Reuse of test cases

	Yes test cases are reused very often.	Sometimes only	No, we never get similar projects again.
Accenture	✓		
Crestech	✓		
Morgan Stanley		✓	
R Systems	✓		
Sears Holding	✓		
Sapient			✓
Snapdeal		✓	
TCS	✓		
United Health Groups	✓		
Ways2sms		✓	

Table 2.11 Percentage of random testing used

	0-20%	20-30%	30-40%	40-50%	50-60%	60-70%	70-80%	80-90%	90-100%
Accenture	√								
Crestech	√								
Morgan Stanley								√	
R Systems								√	
Sears Holding	√								
Sapient							√		
Snapdeal							√		
TCS			√						
United Health Groups			√						
Ways2sms								√	

RQ 13 *Do you think only independent testing team with great expertise can employ exploratory testing technique?*

All the testers whether they are using this technique or not believe that exploratory testing is about employing the tester's intelligence and creativity. The rate of success of exploratory testing rises if the tester is having a good experience in its field.

RQ 14 *Do you think very large organizations can only use exploratory testing?*

All testers believe that exploratory testing can be used by every range of organizations. Those who are using this technique found it easy to deploy as it does not demand any huge budget of testing.

RQ 15 *Do you think exploratory testing technique is domain specific?*

Most of the testers think that this technique can be employed only in the certain cases of domain like when we have a lot of time pressure and there is no time to write the test cases at early stages.

RQ 16 *Would you like to recommend the "exploratory testing" technique to your testing team?*

Although few testers are currently using the exploratory testing technique to deliver the fully tested software in short duration of time but the testers who were not aware of this method also found it interesting and they all planned to suggest it to their test manager and team mates for future projects.

2.4 RESULTS OF REVIEW AND QUESTIONNAIRE INVESTIGATION

i. Software testing is an important and emerging area of research of software engineering field. All the software testers consider it as crucial phase of software development. The phase is inevitable as the main goal of every software company is to produce the best quality product which is free from defects.

ii. Software testing is not only done at the end but also preferred at different levels of software development life cycle. To fulfill this goal, uncountable testing techniques are used in software industries for different testing purposes.

iii. Although every software company focuses on software testing but still they lack certified testers. Testing centric companies give more preference to certification exams and updating knowledge.

iv. There are different software testing techniques available in the books for different testing objectives but still there is discrepancy between methods defined in the books and how they are implemented in the software companies. Every software company practice different style of testing the product according to the time available to them.

v. Exploratory Testing is one of the hot areas of research to produce best quality products in lesser available time. It is a buzz word in the research because of competitive software world having the race of producing best quality product in short time period.

vi. The technique lacks research because of its comparison with random and sloppy work but work of many researchers cleared this confusion. Exploratory testing is a systematic approach and unlike random testing, it is reproducible.

vii. Many questionnaire investigations have already been conducted outside India to work on the defect efficiency of exploratory testing. Results proved that exploratory testing better than traditional approach of software testing [28, 29, 30].

viii. There is still a great need to research on this technique. No research paper has still been found explaining the complete steps of this technique. No algorithm design and testing metrics are yet defined for this technique to measure the test coverage and success of testing. No testing tools are proposed by researchers to understand it. Hence it can be done in the thesis to contribute in this area of research.

CHAPTER 3

PROBLEM STATEMENT

3.1 GAP ANALYSIS

The problem can be easily stated by following simple steps as shown in the figure.

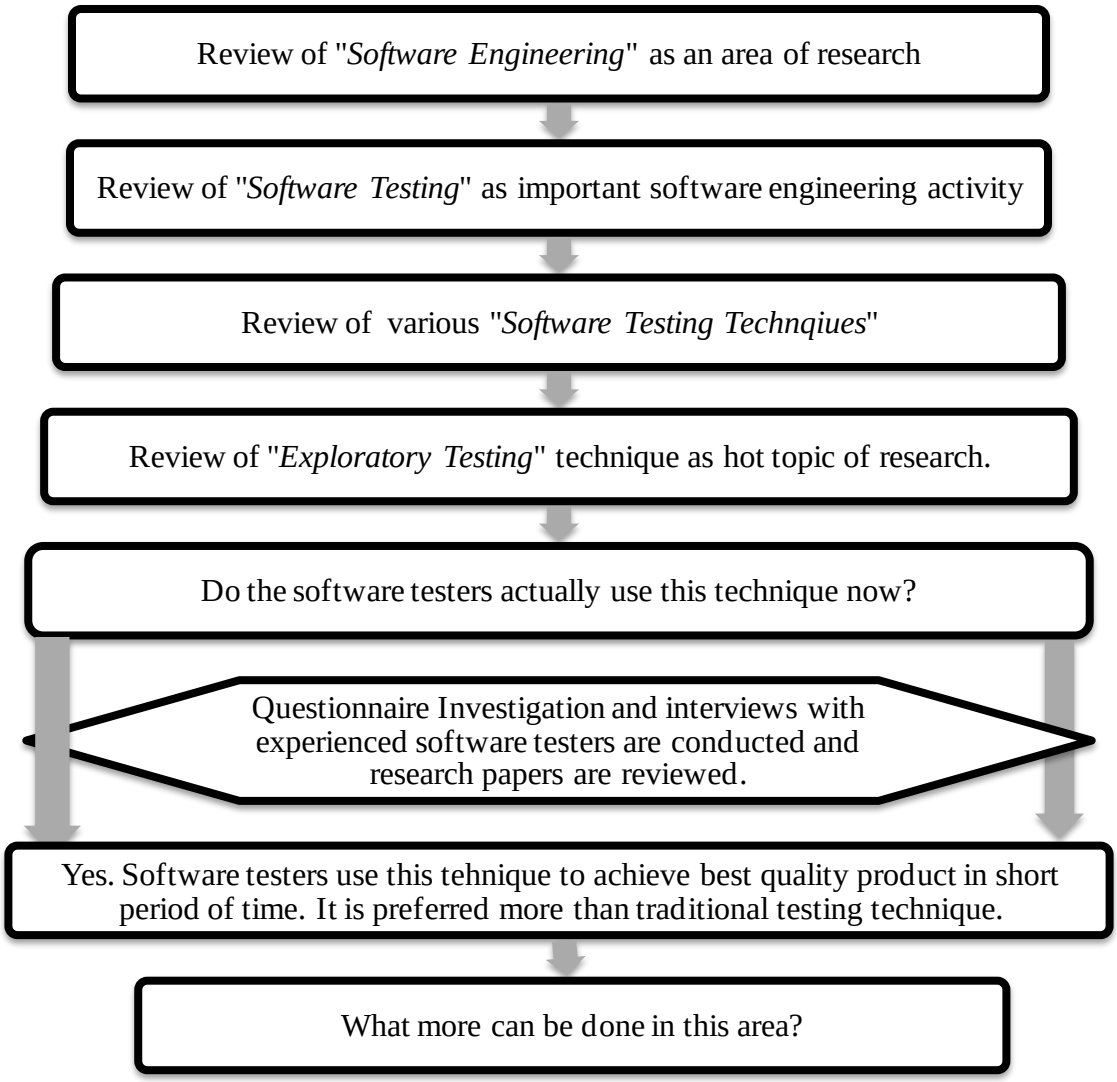

Fig. 3.1 Problem identification

After reaching the target area of research, gap analysis can be done. Gap analysis is useful in order to reach from the current state to desired state or target state of research.

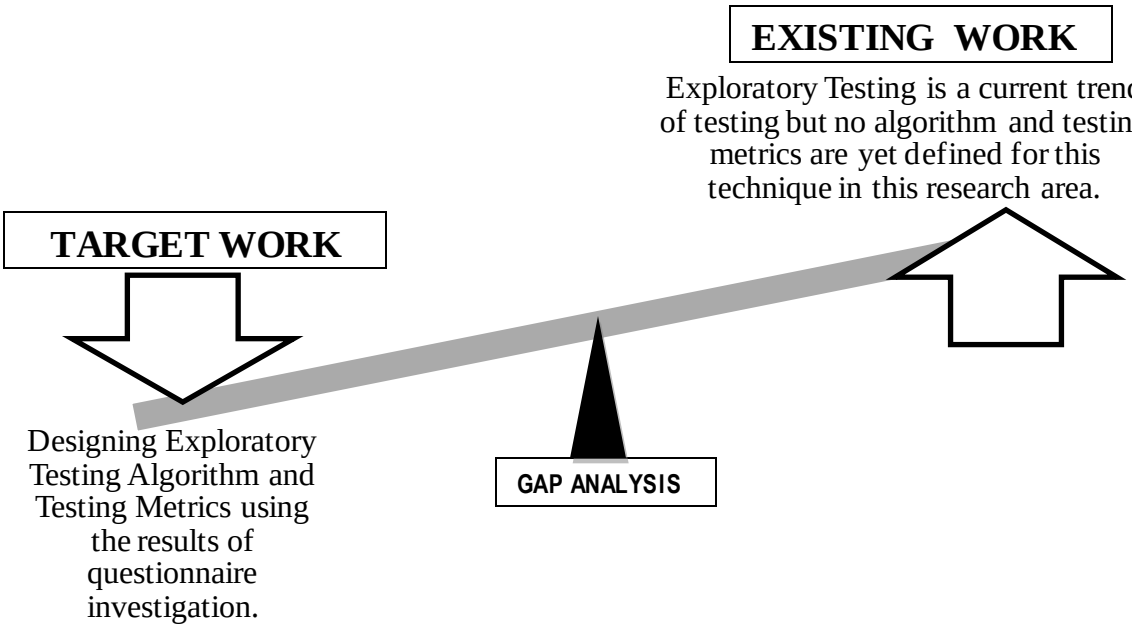

Fig. 3.2 Gap analysis

3.2 TARGET PROBLEM

i. To design and implement the "*Exploratory Algorithm*" which completely explains the exploratory testing technique. A tool is required to be built which demonstrate all testing tasks required to perform exploratory testing.

ii. The rationale is to understand the exploratory testing technique by designing a formal algorithm and to differentiate it from ad hoc testing work. The steps should be clearly defined so as to explain the complete testing process through "Exploratory Testing" technique.

iii. There is also a need to define the testing metrics to measure the test coverage and to define the success of testing using this method.

CHAPTER 4

PROPOSED SOLUTION

4.1 INTRODUCTION

Exploratory testing is one of the popular techniques used by testers under time to delivery pressure to save testing time but it does not get much respect in the research field due to its comparison with ad hoc testing [17,18]. Testers from ten software companies are asked about the technique. As per our knowledge, there is no algorithm or formal steps and metrics present any where to understand this technique. Testers who are already using this technique are asked about its usage. An algorithm is designed for this technique. Important management testing tasks and useful testing metrics are explored as well. A tool using advanced PHP is built to employ this technique.

4.2 EXPLORATORY TESTING ALGORITHM

i. Algorithm explains exploratory testing as a group of formal tours having numerous sessions. The flow chart of proposed "Exploratory Algorithm" is shown in the figure 4.1.

ii. A session is a formal meeting generally of 50 minutes to one hour where group of testers perform the testing. These sessions can be recorded so the results can be reproduced or reused later if required.

iii. Exploratory Testing consists of number of tours. Each tour is unique having unique testing objective and consisting of countable sessions. Simple tour may complete in one session and complex tours may take more sessions to meet the tour objective.

iv. The number of sessions depends upon the project complexity and scope of tour objective. Project is explored and test cases are designed simultaneously while discussing in a group meeting.

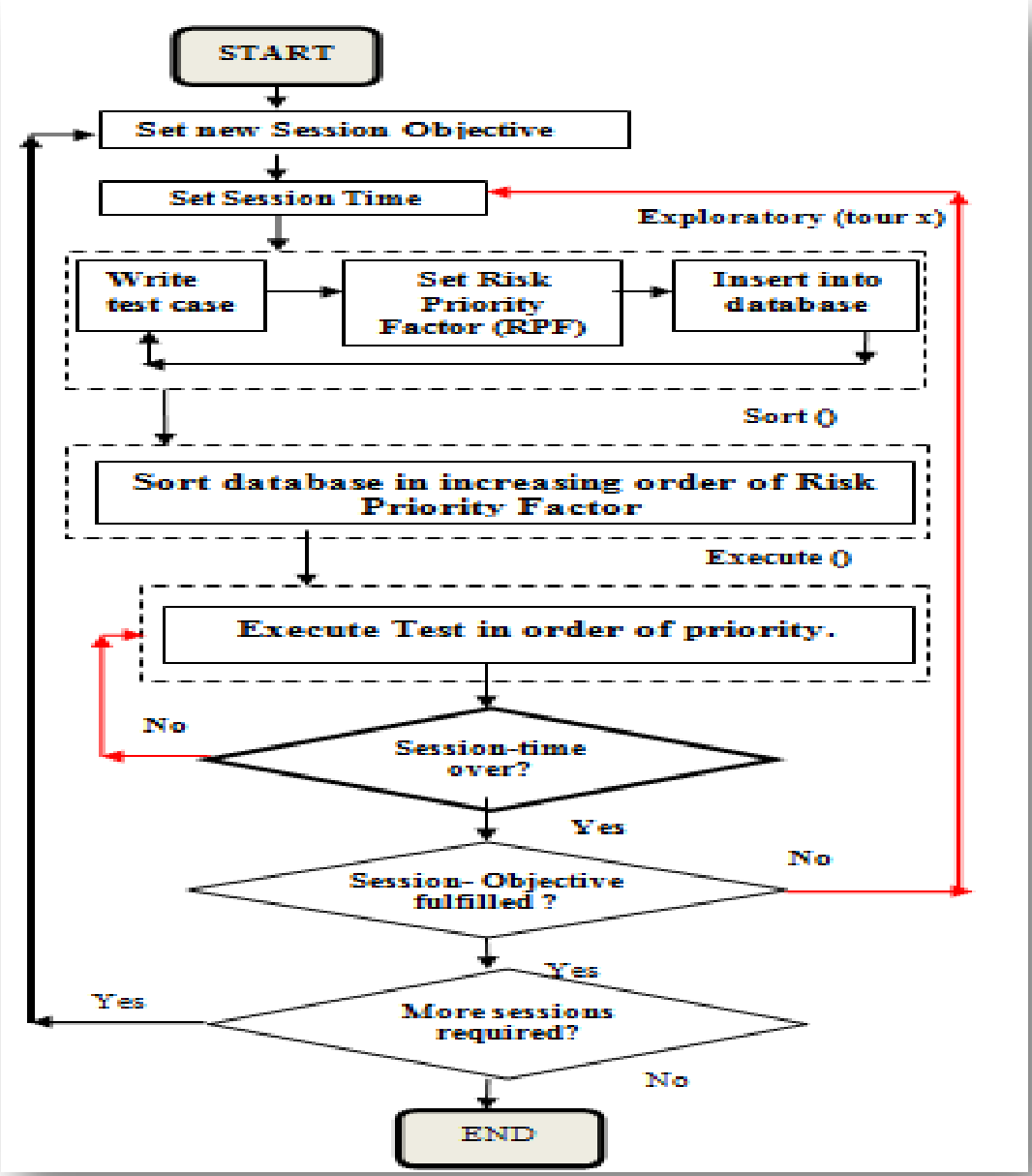

Fig. 4.1 Exploratory testing algorithm flowchart

v. Risk Priority Factor i.e. RPF is an important element of algorithm which analyzes the risk involved in the project. RPF is assigned a value 1 to 10. Lesser the value, higher is the risk involved with the test objective. So the test cases with value 1 should be treated with higher most priority and hence are

executed initially. Test cases are sorted according to RPF priority value because sessions are of limited time. As this technique is practiced under time to delivery pressure, so the test cases with lower priority like no. 7-10 can be skipped to save time.

The pseudo code for particular session in any unique tour x in the designed algorithm is mentioned below-

Begin

{

1. Set SESSION_OBJECTIVE for TOUR(X)
2. Set time for 50 to 100 minutes as SESSION_TIME.
3. Set EXIT CRITERIA= SESSION_OBJECTIVE
4. Start new session (**CALL EXPLORATORY (TOUR(X)**)

    // call exploratory (tour (x)) function for tour X.

5. **SORT()** // call sort() function
6. WHILE (TIMER<= SESSION_TIME)

    {

7. **EXECUTE()** //call execute() function

    }

8. IF (EXIT CRITERIA= = PASS)

    {

9. Close Session

    }

10. ELSE

    {

11. GOTO 2.

}

12. GOTO 1

}

END

CALL EXPLORATORY (TOUR(X))

{

Insert test case into database

{

Set TEST ID

Set TEST OBJECTIVE

Set CURRENT INPUT

Set EXPECTED OUTPUT

Set RISK PRIORITY FACTOR from 1 to 10.

}

SORT ()

{ SELECT * FROM table_name

ORDER BY RPF ASC; } //SQL QUERY

EXECUTE()

{

Execute sorted test cases.

}

4.3 EXPLORATORY TESTING METRICS

Software metrics is a large area of research and any software engineering work is incomplete without its measurement methods [32]. Quantitative measurement through testing metrics is the best way to judge the success of testing. Based on the requirements and designed algorithm, exploratory testing metrics are derived as follow-

1. **Tour Count (TC) =** Number of Tours Exercised = A+B+C……

   where A,B,C…are unique tours exercised.

   a) Each tour is unique and defines a unique testing objective within the scope of the project. More the complexity of application under test, more tours are required to test the project.

   b) The metric is important as it divides the project in into small modules and tours are defined for every module to be tested.

   c) The division of project into different modules simplifies the work of testing.

   d) Tour count value specifies that how many areas of project have been tested.

   e) Larger tour count value brings more confidence for the product quality in the mind of tester.

2. **Tour Complexity =** No. of sessions required to complete one tour

   a) Tour Complexity defines how complex the testing objective of tour is. It is defined in the number sessions required to fulfill the objective of tour.

   b) Simple tour may finish in one session with required time of less than one hour but complex tours need more than one session. If the scope of the tour is large then more testing is required.

   c) The need of this metric is aroused from the application of Pareto Principle. According to this principle, defects get clustered at one area. It is also considered as an important principle of software testing. According to Pareto Principle, defects follow the 80:20 rule of project management [33]. It means 80 percent of

the defects get clustered around same 20 percent of the area [34]. If the tour complexity is highest, it means the sessions needed to complete that specific tour is the highest. It discloses the area of project which is complex having largest number of defects. So the specific area of project is needed to be focused so that more defects can be found to produce defect free product.

d) It helps the testers to judge which area of project is more attacked by defects. Sometimes complex code may also lead to more defects. So complex tours leads to more sessions and hence number of test cases generated to test more defects get also raised.

3. Test Case Count

#Test_Cases (tour(x)) = No. of test cases executed during x^{th} tour

a) It specifies the number of test cases generated during one unique tour x.

b) Larger the complexity of a tour then more is the number sessions required to fulfill the tour objective and hence more test cases are need to be generated during sessions.

c) Large number of test cases does not always produce defect free software. The test cases should be relevant to fulfill the testing objective of any specific tour.

d) In exploratory testing algorithm, Risk Priority Factor (RPF) value is assigned to every test case to define the priority of any unique test case.

e) If any tour has maximum number of test cases with RPF value 1-5, hence tour has covered more test cases to produce a risk free product.

4. Test Coverage

A. $$\textbf{Tours Test Coverage} = \frac{No.of\ successful\ \ Tours}{Tour\ Count\ (TC)} * 100$$

where successful tours are number of tours that are executed successfully as per defined objective.

a) Test coverage is considered as one of the important testing metrics to measure the success of testing. It measures the completeness of software testing goal as per defined customer requirements. Hence considered as a good tool for predict the quality of software. [35, 36].

b) Test coverage of tour is defined to measure the completeness of testing for any unique testing tour.100% tour coverage does not mean the area or scope of project tested under that particular project is free of defects. It only means that test cases generated as per defined tour objective has been successfully executed. More test cases can also be generated as per the requirement of meeting the testing objective of particular tour.

B. **Total Test Coverage**

$$\frac{\#testcases\_executed(tourA) + \#testcases_{executed(tourB)} + \cdots \ldots}{\#testcases(tourA) + \#testcases(tourB) + \cdots \ldots}$$

a) Total Test coverage is calculated to measure the completeness of testing of the complete project.

b) 100% total test coverage does not mean 100% defect free project. It only implies the success of test cases written to fulfill the test objective. More test cases can be designed to find more defects.

c) This metric helps to judge the success of testing the project under test. If tester has to decide the acceptability of project from user point of view, project has to successfully pass all the generated test cases during each and every tour of testing.

d) Test coverage plays a vital role in judging the quality of product. The goal of the tester is to make the test coverage 100%.

e) Although testing never ends even after 100% test coverage is achieved as more test cases can be generated as per the requirement of project. But it is necessary that the test cases which get failed during session of testing must be attended and defect should be removed from the project.

5. Tour Effectiveness by Risk Priority Factor (RPF)

$$\textbf{Tour Effectiveness (tour x)} = \frac{\sum(10-\text{RPF}) * \#\text{Test\_Cases (RPF (n))}}{\text{maximum value}} * 100$$

Where n=1 to 10

a) Larger the percentage of test effectiveness of any tour, more effective is the tour as it includes the execution of test cases of prime concern.

b) RPF value depends upon how much important it is to remove the defect when compared with customer requirements. If tour effectiveness is high, it means tour has considered maximum test cases to check the prime customer requirements.

c) Risk assessment is major issue of software testing activity [37]. Risk analysis is the major advantage of exploratory testing algorithm where risk priority values are assigned to all test cases.

d) The test cases are sorted in ascending order of priority because high priority test cases should not be skipped at any cost to produce risk free software.

e) It helps the testers to judge which tour has effectively focused on the test cases that are of prime concern. It does not relate with severity of project test cases as severity has more concern with the test cases which may lead to the failure of project. Priority helps the test to judge which test cases should be attended as early as possible, hence very helpful to produce risk free product.

f) The main goal of any exploratory testing during any unique tour execution is to improve the value of risk effectiveness as much as possible.

CHAPTER 5

PROJECT IMPLEMENTATION AND DISCUSSION

5.1 IMPLEMENTATION

A tool is developed in Advanced PHP language and designed exploratory algorithm and metrics are implemented in the tool to test the application.

5.1.1 PROJECT GOAL

This Exploratory testing tool is a standalone system. The main objective is to provide a foundation to the user to understand the exploratory testing technique.

5.1.2 SPECIFIC REQUIREMENT:

a. Hardware Requirements

i. 25 MB of free space on the hard drive.
ii. CD-ROM
iii. Intel Pentium 300 MHz or higher (or equivalent)
iv. 256 MB available RAM.

b. Software Requirements:

i. Microsoft Windows.
ii. PHP (version 5.3 or above)
iii. Apache Sever (Version 2.2 or above)
iv. My SQL (Version 5.1 or above)

5.1.3 APPLICATION UNDER TEST (AUT) EXAMPLE

The application to be tested is called Application Under Test (AUT). Here we have taken the example of an image processing editor with all basic image editing operations which acts as testing target or application under test (AUT). Goal of software testing is to check whether the requirement mentioned in the Software Requirement Specification (SRS) are

achieved in the implemented software. Software Requirement Specification (SRS) consists of three things which should be clearly defined as shown in table 5.1.

i. Goal of project
ii. Functional Requirements
iii. Non Functional Requirements

Table 5.1 SRS of Application Under Test

Goal of project
To develop a simple image editor application with all basic operations of image editor.
Functional Requirements
An image processing application editor is developed with the following basic requirements:
Image Loading
Cropping and image orientation
Brightness
Saturation
Resolution
Hue Modifier and color change.
Quantization
Binarization
Ordered Dithering
Spatial Filtering
Morphological operations like dilation and erosion
Edge Detection
Histogram
Non Functional Requirements
User Friendly
Good performance
Portable
Extensibility

5.2 DEMONSTARING EXPLORATORY TESTING ALGORITHM

5.2. 1 Initial Testing Tasks

Tool consists of various modules to perform various testing tasks as shown in figure 5.1.

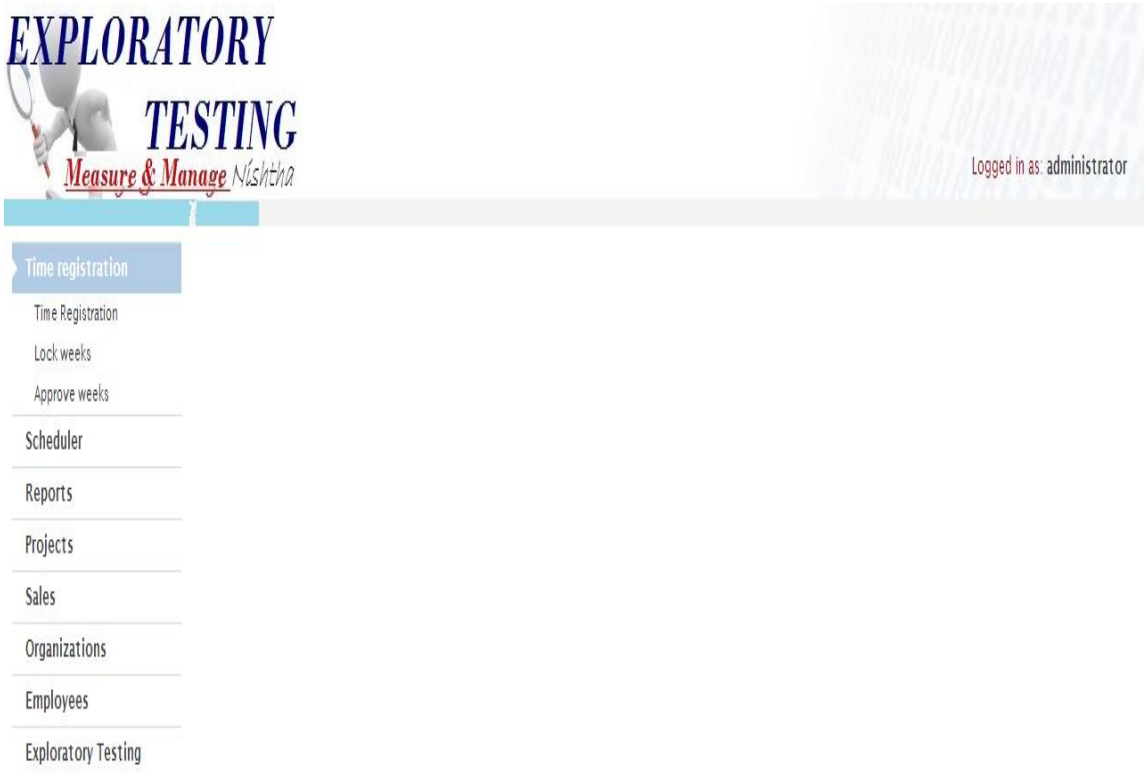

Fig. 5.1 Exploratory testing tool home screen

To start the testing of any project, project detail need to be added to the testing tool as shown in figure 5.2.

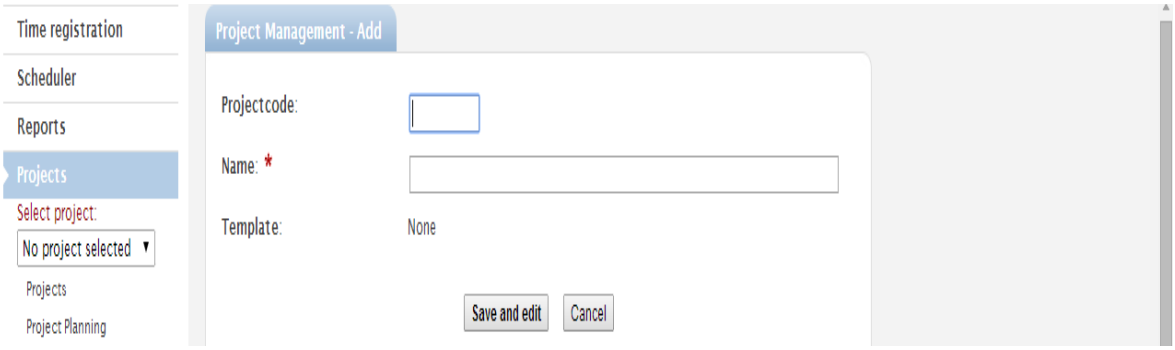

Fig. 5.2 Managing project details window

Once the project detail is added to the database "projects", it can be viewed in the project window at the bottom of page of "projects" module as shown in figure 5.3.

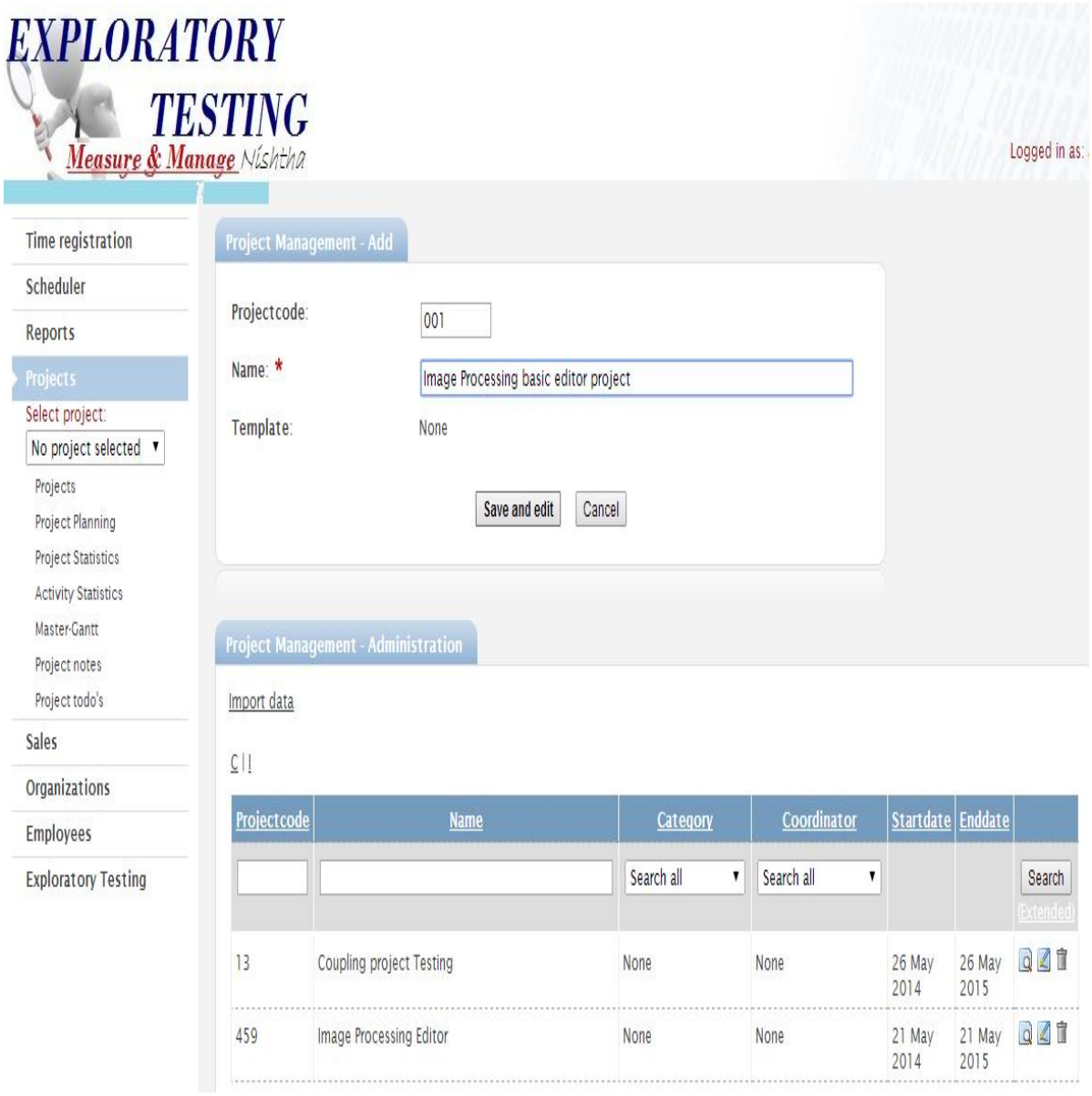

Fig. 5.3 Project database window

Once the name of the project is added, further detail of the project can be added to the project. Click on the name of project in the project database window will open a form to enter the further project details as shown in 5.4.

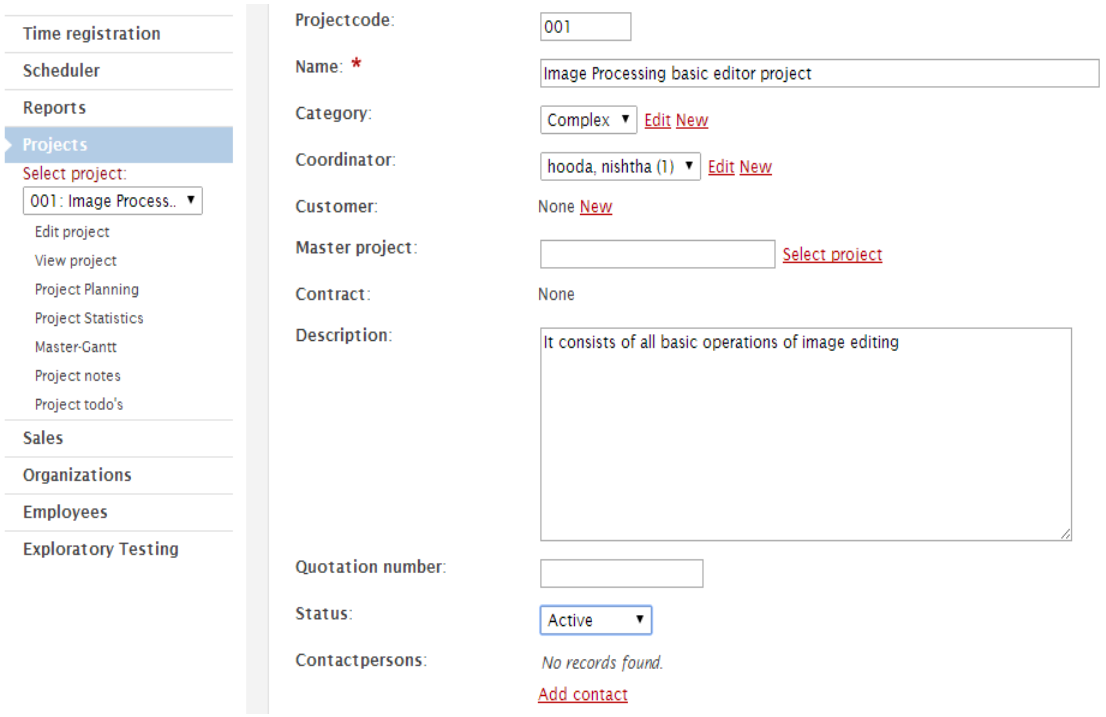

Fig. 5.4 Project detail form

Saved project can also be edited. Another window will be opened to manage more details as shown in figure 5.5.

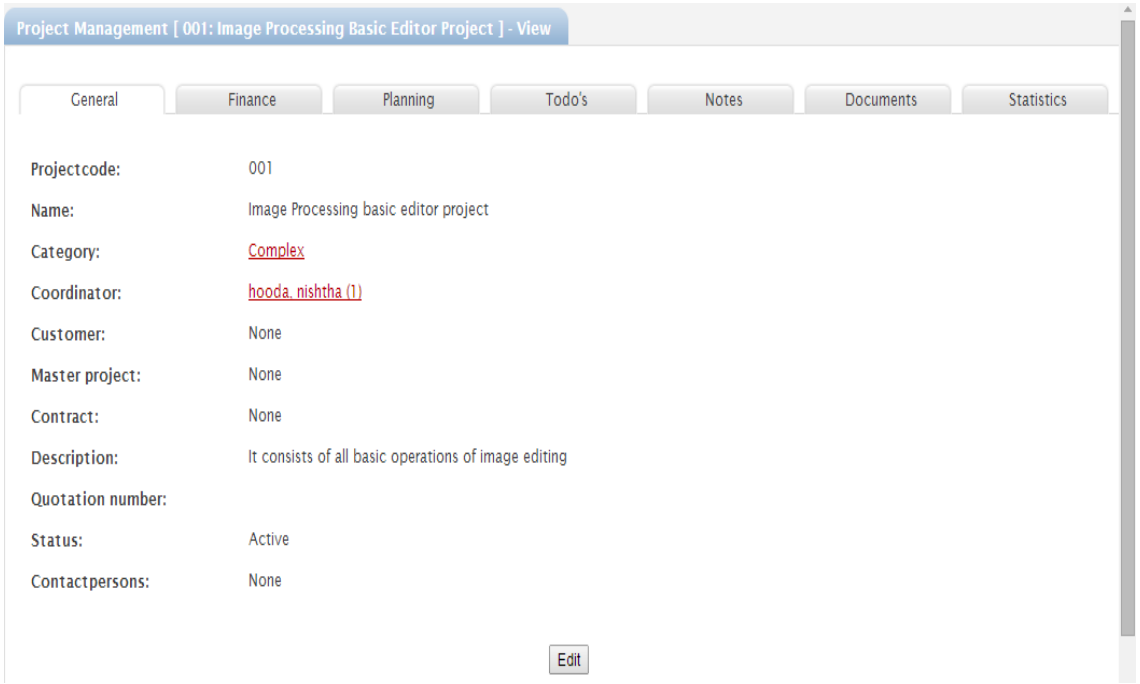

Fig. 5.5 Project detail edit form

Click on “Exploratory Testing” menu to start any project. The saved project window will be opened as shown in the figure 5.6.

Fig. 5.6 Exploratory testing menu

Click on “Start Exploratory” to start any saved project as shown in figure 5.7.

Fig. 5.7 Exploratory testing home screen

Click on "Start Tour" to move further. It offers you to add the tour details to continue as shown in the figure 5.8.

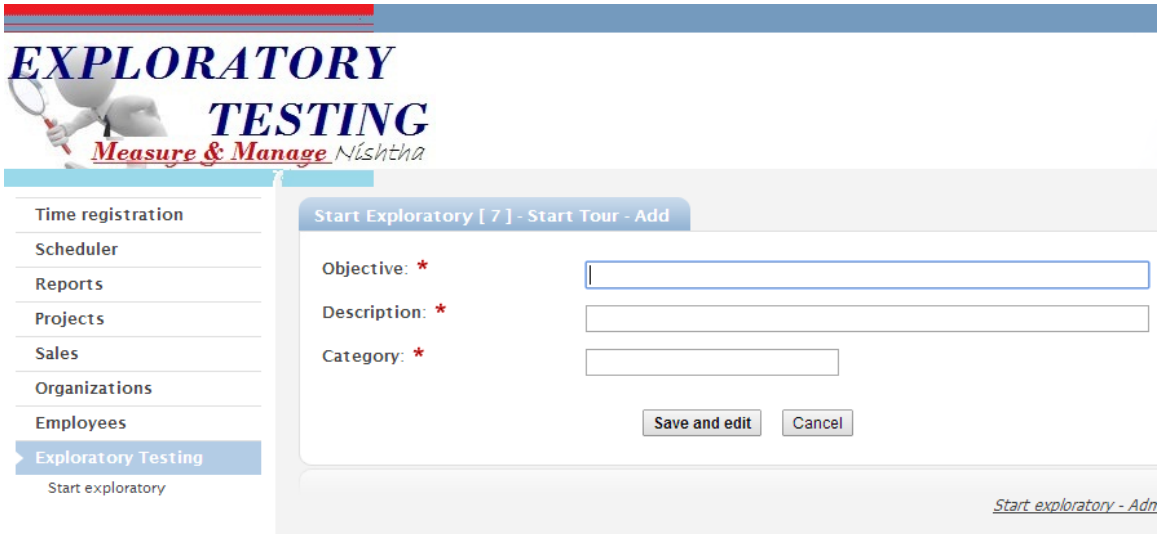

Fig. 5.8 Exploratory testing tour window

Click on "save" to save tour details and to move to the next step of exploratory testing as shown in the figure 5.9.

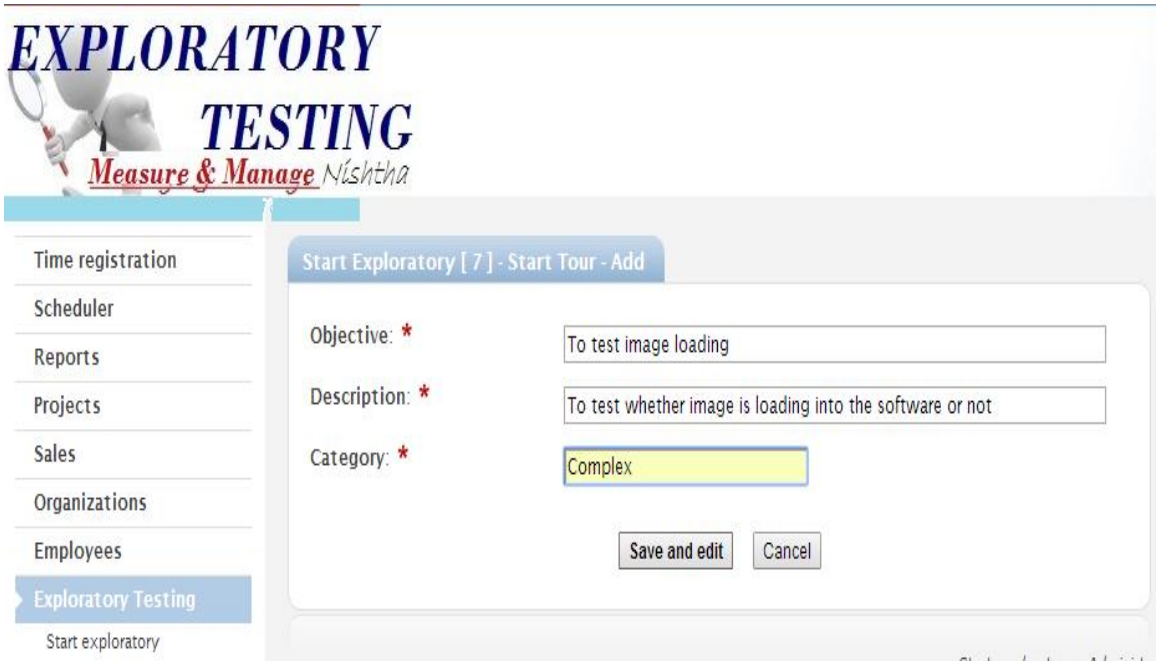

Fig. 5.9 Exploratory tour- adding screen.

Exploratory testing starts with the collection of tours. Tours objective depends upon the functional requirements. The exploratory testing covers testing of all the functional and non functional requirements. Both simple and complex tours were executed. To explain in brief, examples of both simple and complex tour are explained below:

5.2.2 Simple Tour Example

Testing starts with the simplest tour of testing whether application is successfully accepting image for editing or not as shown in figure 5.10.

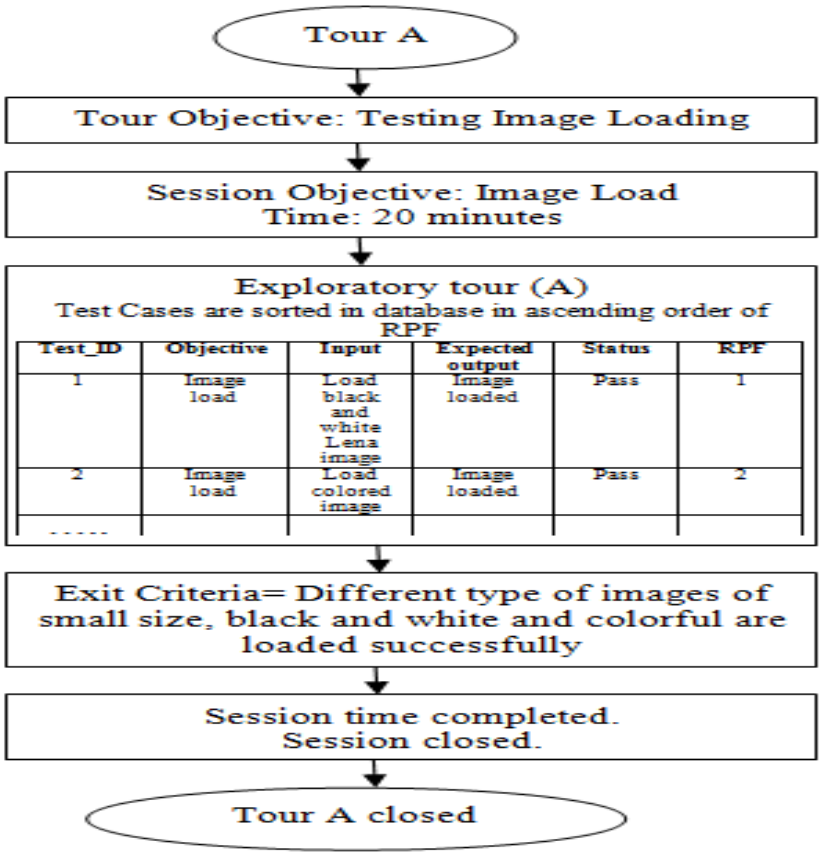

Fig. 5.10 Simple tour example

Tour A is started with formal meeting and small session of 20 minutes was considered sufficient to meet the tour objective. Various test cases are designed and loaded in the database and application is simultaneously tested for the same.

5.2.3 Complex Tour Example

i. It consists of more than one session. Testing starts with the complex tour of testing whether application will successfully performing image edge detection or

not. Tour B is started with formal meeting with group of students and a session of different time are chosen as shown in figure 5.11.

ii. Edge detection is an important tool used in many areas of image processing like feature detection and feature extraction [38]. The main goal of applications in this area is to extract points in the digital image where significant change in the brightness or discontinuity is observed [39].

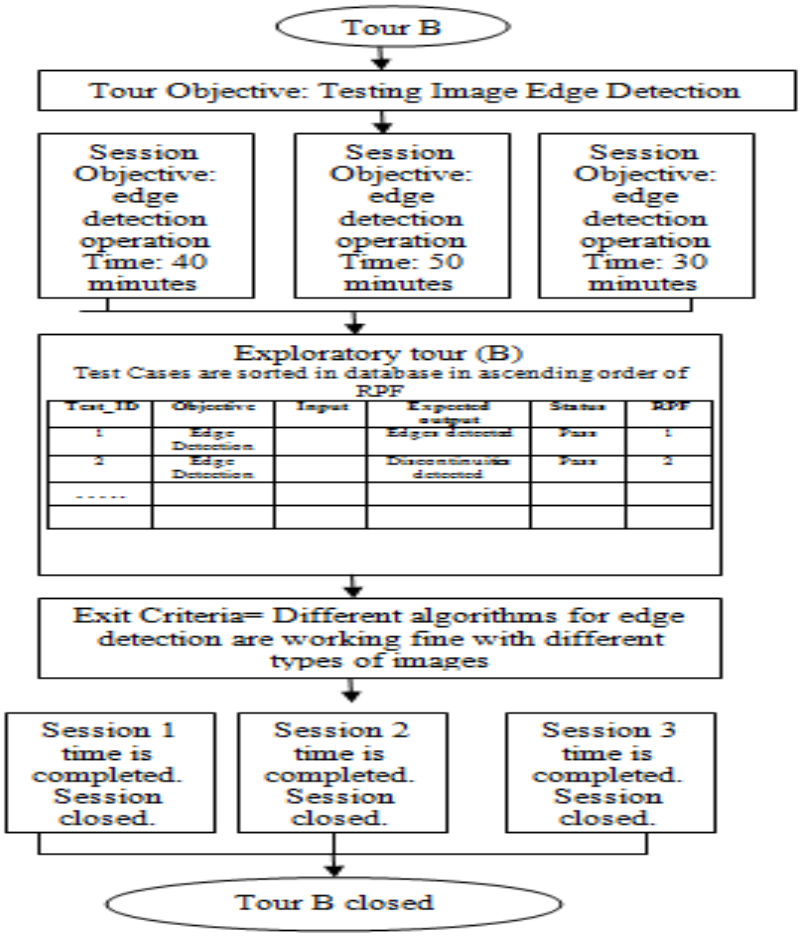

Fig. 5.11 Complex tour example

iii. This complex tour is completed in three sessions of 50 minutes each. Various test cases are designed for tour B with different values of RPF and loaded in the sorted database and application is simultaneously tested for the same.

As the tours are finalized, next step is to start the session. It will ask you to confirm the tour details and add session details to move to the next step as shown in figure 5.12.

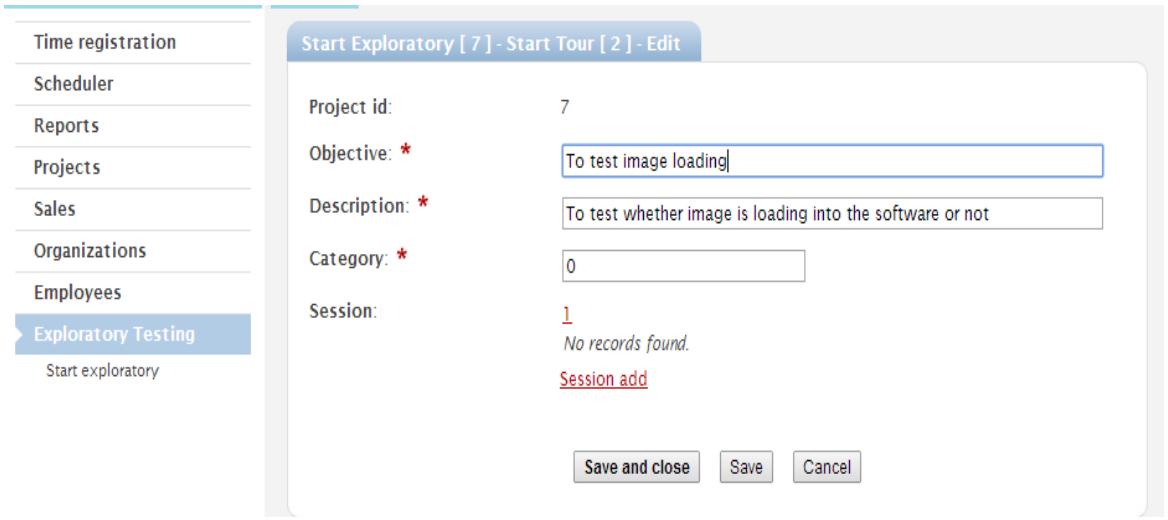

Fig. 5.12 Exploratory testing tour editing window

Click on "Session Add" to start a new session as shown in the figure 5.13

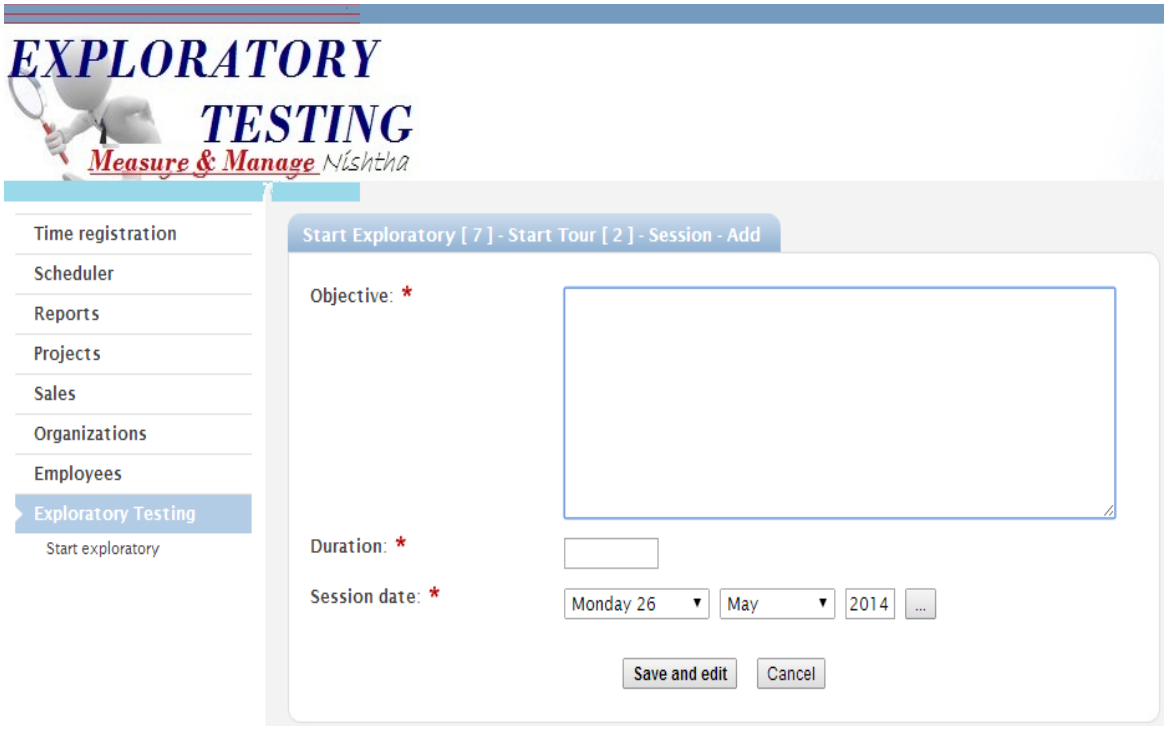

Fig. 5.13 Exploratory testing session home screen

Fill the session details and click on “Save” to move to the next step as shown in figure 5.14.

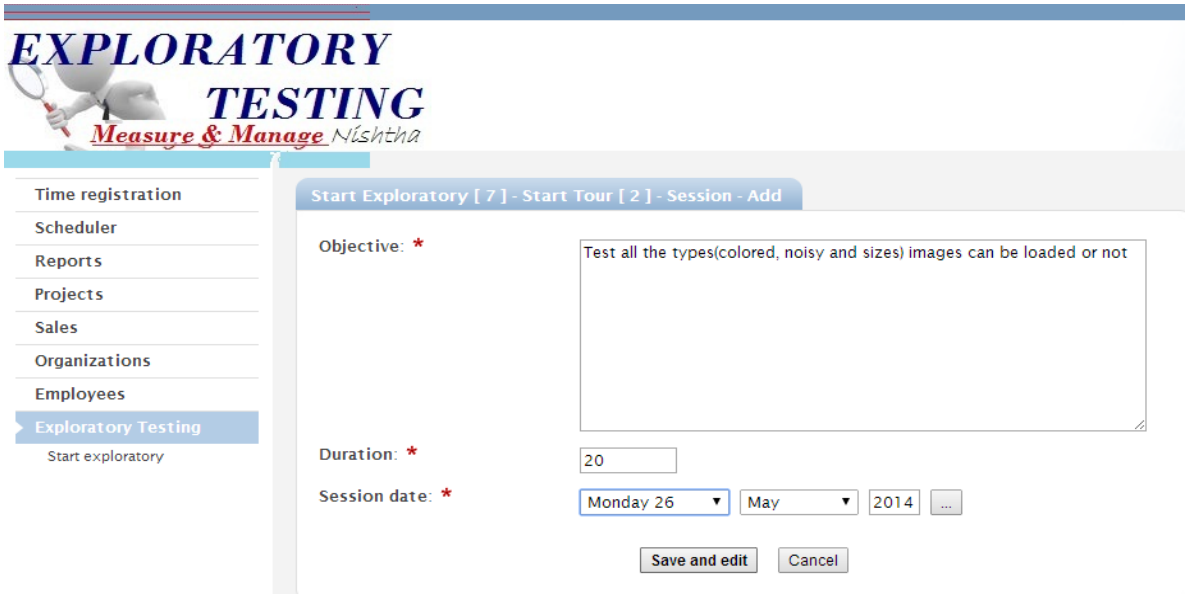

Fig. 5.14 Exploratory testing session adding screen

It asks you to create the “test cases” in the given session shown in the figure 5.15.

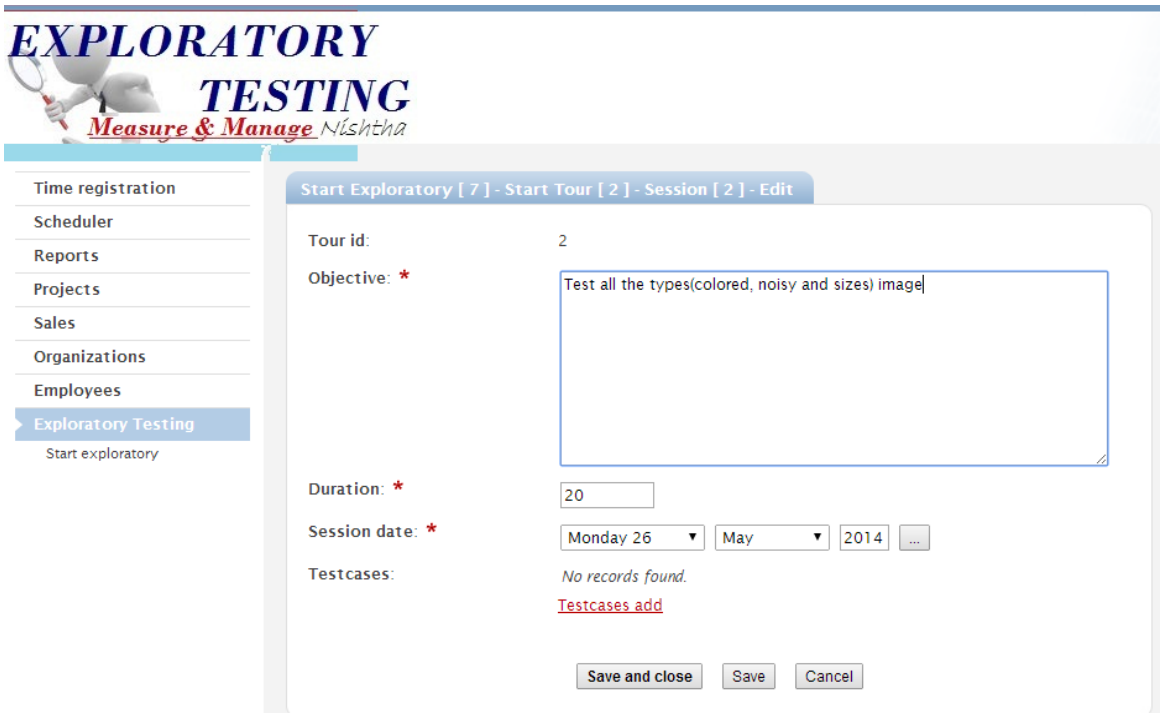

Fig. 5.15 Exploratory testing test case generation screen

Generated testcases get stored in the “testcase” table and can be visible at the bottom of test case generation window as shown in the figure 5.16. The testcases get automatically ordered according RPF value.

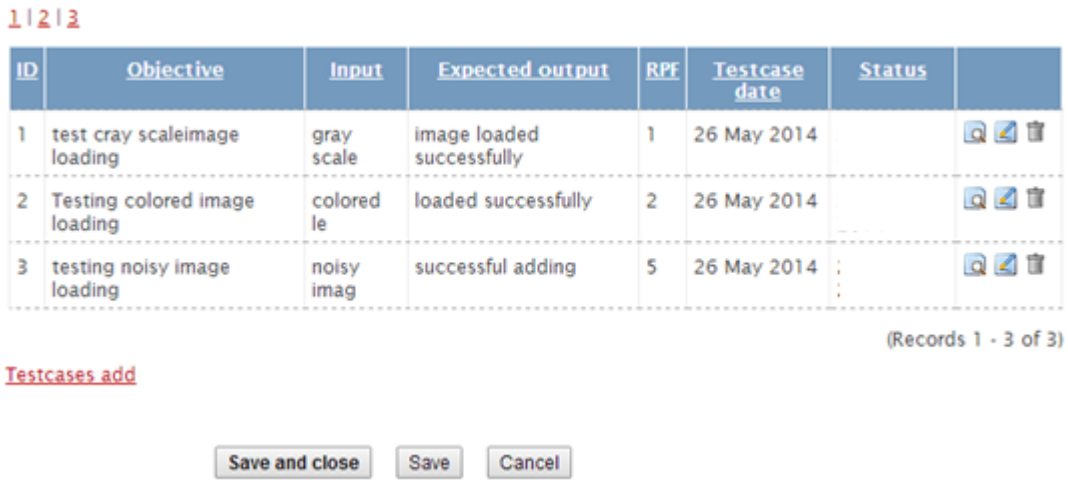

Fig. 5.16 Exploratory testing test case database window

Testing team members details can also be added in the employee module of the tool as shown in the figure 5.17.

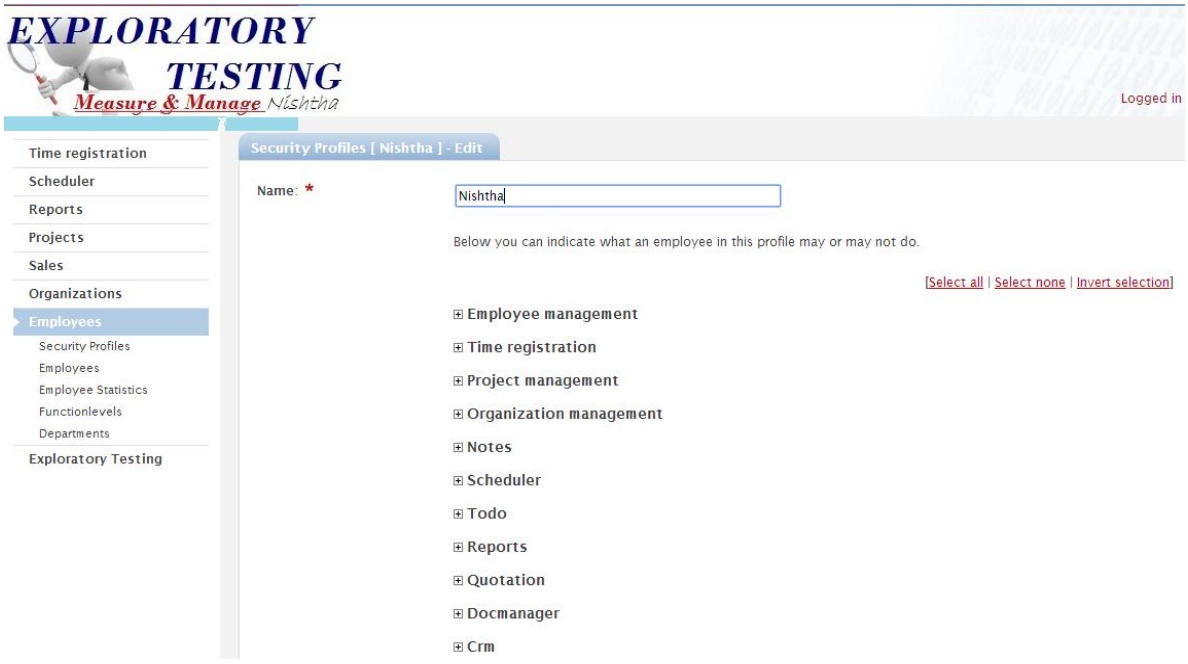

Fig. 5.17 Exploratory Testing employee details screen

Manage authority details of any tester in “exploratory testing” as shown in figure 5.18.

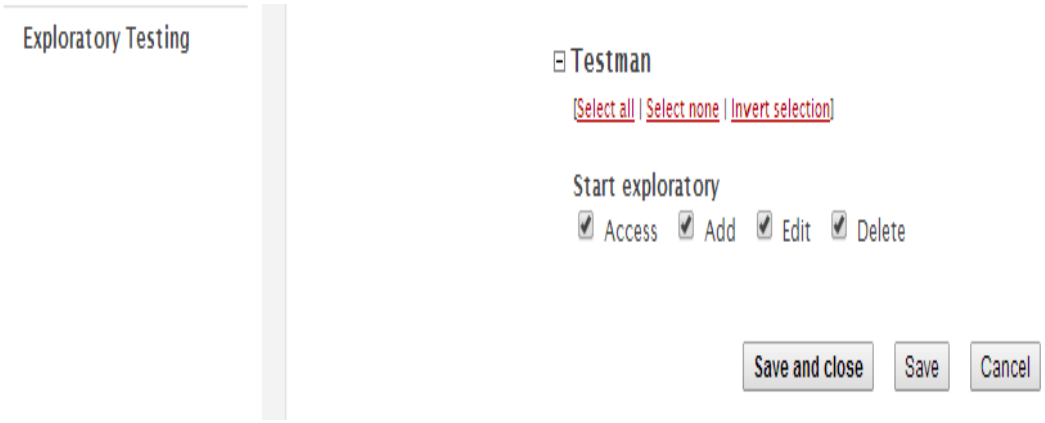

Fig. 5.18 Employee authority add detail screen.

Manage notes and attached documents during sessions of exploratory testing by clicking on “choose file” as shown in the figure 5.19.

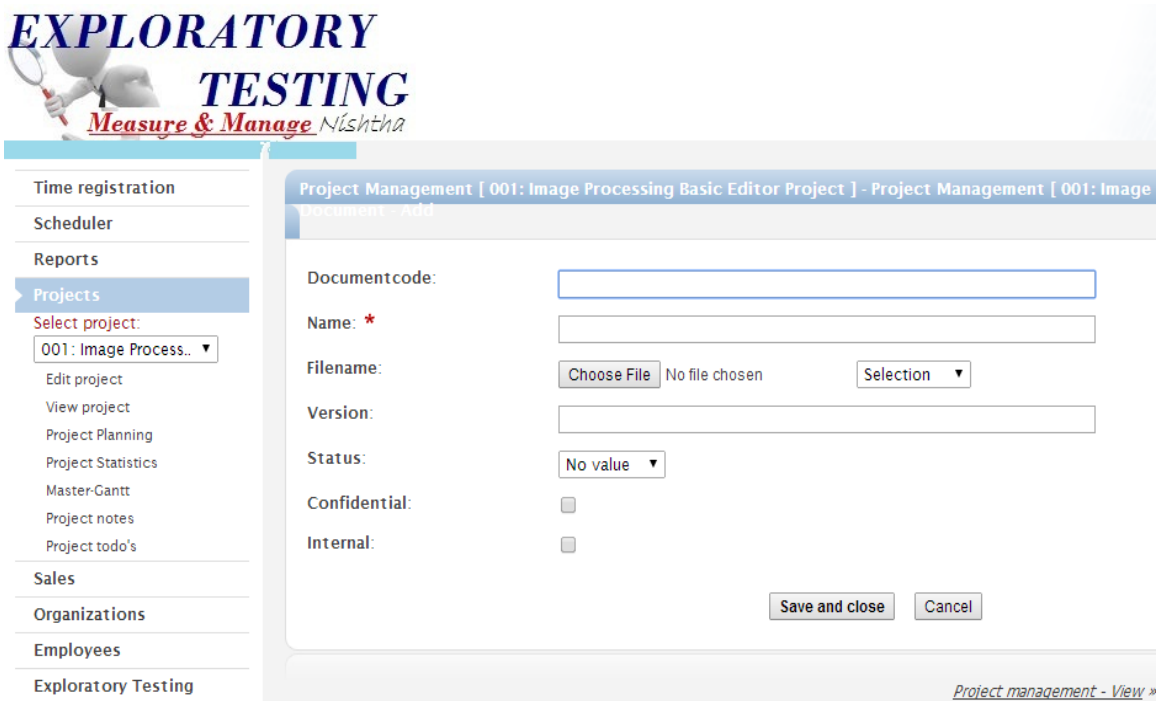

Fig. 5.19 Exploratory testing notes and documents-attached window

Manage details of every phase and activity of exploratory testing as shown in the figure 5.20. Start date, status and ever description can be managed in this tool.

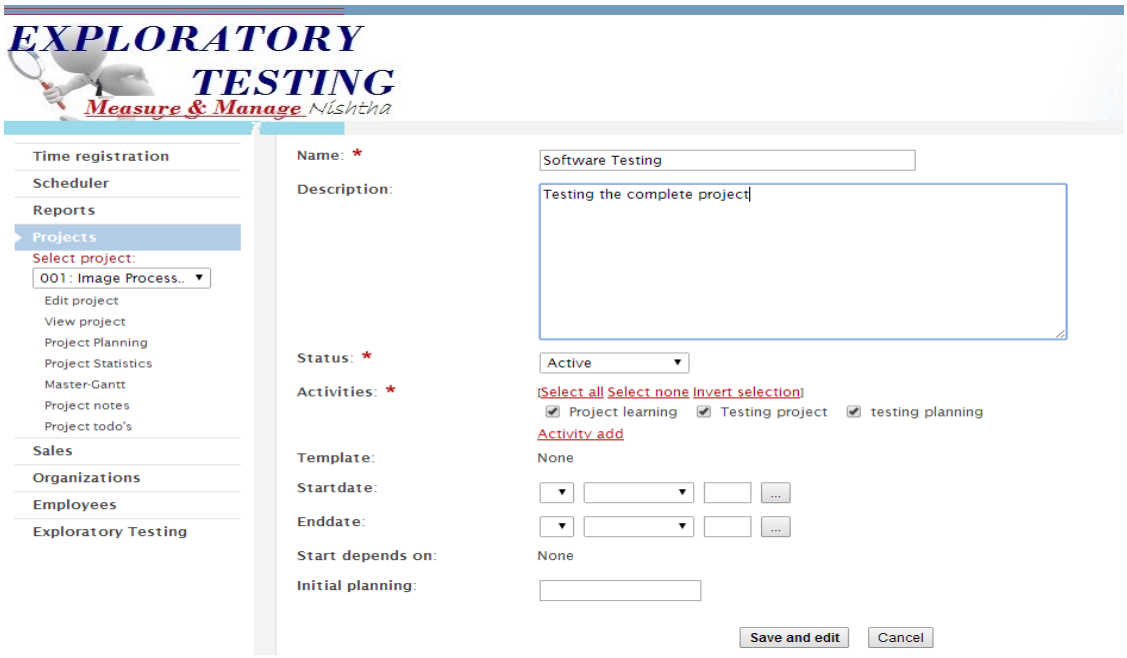

Fig. 5.20 Exploratory testing activity adding window

Manage schedule of every step of exploratory testing as shown in the figure 5.21

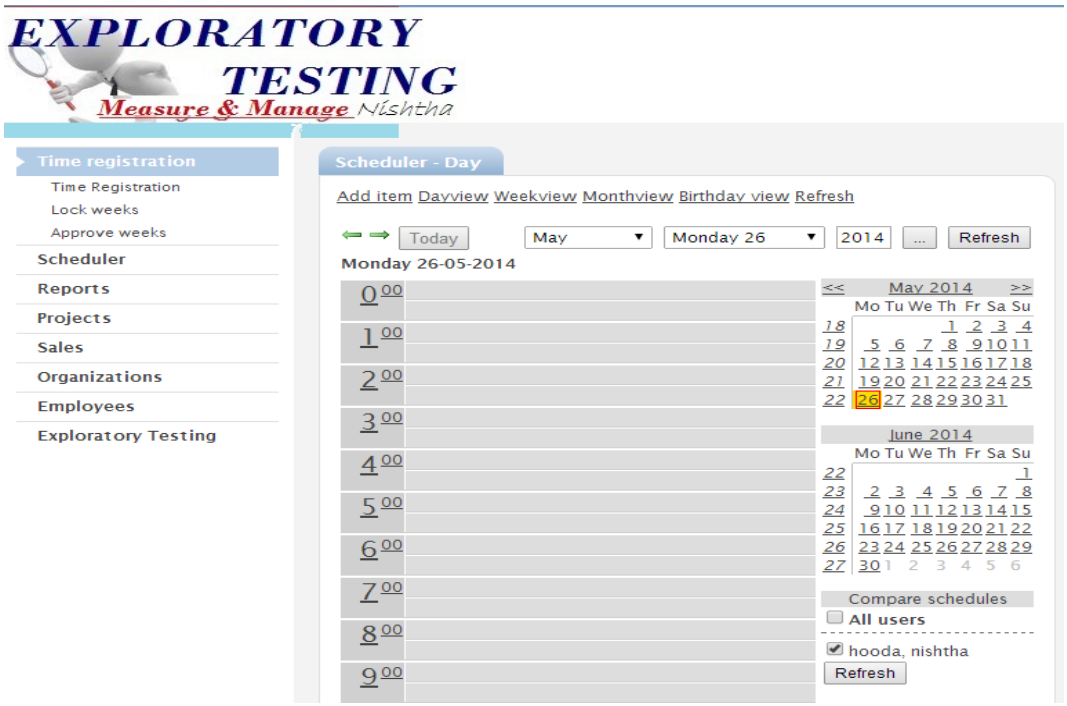

Fig. 5.21 Exploratory testing activity scheduler window

Enter the Project Id to check the status of exploratory testing. It will show number of tours, sessions and test cases entered of the respective project as shown in the fig. 5.22.

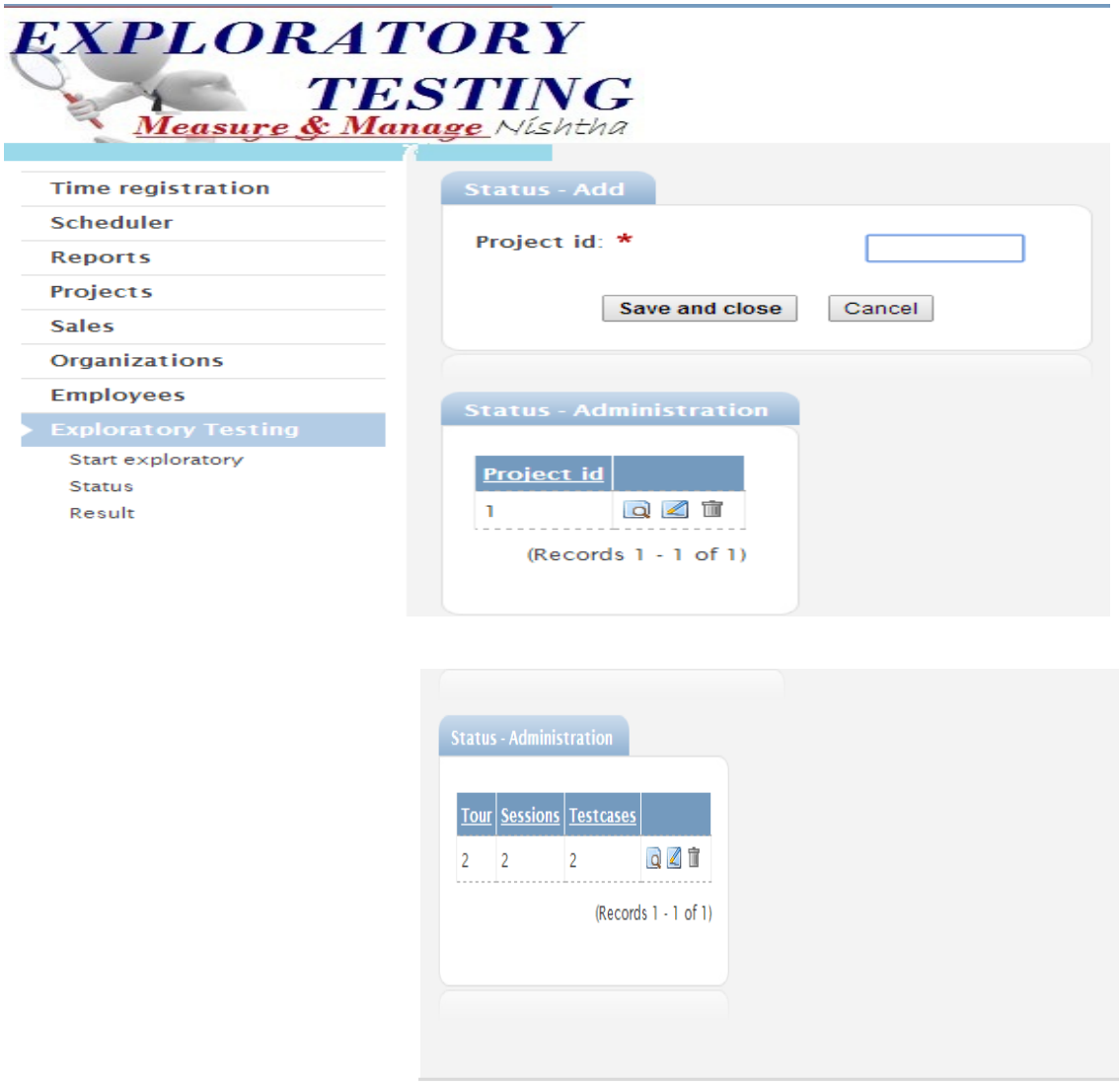

Fig. 5.22 Status of exploratory testing

5.3 TESTING METRICS CALCULATIONS

To calculate various designed testing metrics, a table is drawn as shown in table 5.2. It explains the example of calculating various metrics for application under test.

Table 5.2 Exploratory testing metric example

No.	Tours Objective	#Sessions	#Test cases	Status
1	Image Loading	1	10	Pass
2	Cropping and image orientation	1	8	Pass
3	Brightness	1	5	Pass
4	Saturation	1	6	Pass
5	Resolution	1	5	Pass
6	Hue Modifier and color change.	2	22	Pass
7	Quantization	2	18	Pass
8	Binarization	3	28	Pass
9	Ordered Dithering	3	24	Pass
10	Spatial Filtering	3	25	Pass
11	Morphological operations like dilation and erosion	3	31	Pass
12	Edge Detection	3	32	Pass
13	Histogram	2	16	Pass
	Total----------→	26	196	

i. Tour Count =13

ii. Simple Tour =5

iii. Complex Tours= 8

iv. Tours Test Coverage = (No. of successful Tours)/(Tour Count(TC)) * 100 =13/13 = 100%

v. Total Test Coverage

= (#testcases_executed(tourA)+(#testcases_executed(tourB) +…… ……)/(#testcases(tourA)+#testcases(tourB)+……)

=196/196

=100%

Table 5.3 Test cases for tour A

Test Id	Test Objective	Input	Expected Output	Status	RPF
1	Load grayscale image	Gray scale lena image	Image loaded successfully	Pass	1
2	Load black and white image	Black and white image	Image loaded successfully	Pass	1
3	Load colored image	Colored image	Image loaded successfully	Pass	1
4	Load image of less than 100 KB	Image of size 50 Kb	Image loaded successfully	Pass	3
5	Load image of more than 1 MB	Image of size 1.5 MB	Image loaded successfully	Pass	2
6	Load image of .jpg format	.jpg image	Image loaded successfully	Pass	2
7	Load image of .png format	.png image	Image loaded successfully	Pass	4
8.	Load noisy image	Load image of noise 50	Image loaded successfully	Pass	4

		decibels			
9.	Load corrupted image	Load corrupt image	Image is not loaded	Pass	5
10.	Load more than one image at a time	Loading 2 images	Image is not loaded.	Pass	6

Risk Analysis: Tour Effectiveness (Tour A)

= 3 tests of RPF 1 + 2 tests of RPF 2 + 1 test of RPF 3 + 2 tests of RPF 4 + 1 test of RPF 5 + 1 test of RPF 6

= 3* (10-1) + 2* (10-2) + 1* (10-3) + 2* (10-4) + 1* (10-5) +1* (10-6)

= (3*9) + (2*8) + (1*7) + (2*6) + (1*5) +(1*4)

=27+ 16 + 7 + 12+ 5 + 4

=71

Maximum value = 10*9 = 90

% tour effectiveness= (90/71)*100= 78.9%

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

Exploratory Testing is a current trend of software market to win the race of assuring quality to the software products in the competitive world. The formal algorithm and defined metrics differentiates it from ad hoc testing. It is a systematic approach and cannot be called same as careless and sloppy ad hoc testing.

6.2 SUMMARY OF CONTRIBUTION

Questionnaire Investigation with experienced testers conducted in the ten software companies helped us to know the status of software testing as well the usage of exploratory testing technique in the software industry.

Formal algorithm is designed and demonstrated with example to understand the technique and to differentiate it from sloppy ad hoc testing. Software testing metrics are also designed and used in the project to measure the testing work.

6.3 FUTURE SCOPE

Exploratory testing metrics help testers to judge the success of testing in short time span. More testing metrics can be designed and implemented for future work. Complex tools can be built in future to encourage this technique.

REFERENCES

[1] Abran Alain Moore et al. ,"Introduction," in *Guide to the Software Engineering Body of Knowledge*, IEEE Computer Society, 2004, pp. 12

[2] Ahmed Saleem Abbas, W. Jeberson, V.V. Klinsega, "A Literature Review and Classification of Selected Software Engineering Researches," in *International Journal of Engineering and Technology*, Vol. 2, No. 7, July, 2012, pp. 1

[3] Roger S. Pressman, "Introduction", in *Software engineering: A Practitioner's Approach*, McGraw-Hill, ISBN 0-07-365578-3, 5th ed., 2005, pp.21

[4] Ian Sommerville, "Introduction", in *Software Engineering by Ian Sommerville*, 7th ed., 2004, pp. 45-51

[5] Bill Hetzel, "An Introduction", in *The Complete Guide to Software Testing*, 2nd ed., 2004, pp.12

[6] Abran Villavicencio, "Towards the Development of a Framework for Education in Software Measurement," in *Eighth International Conference on Software Process and Product Measurement*, Joint Conference of the 23rd International Workshop, 2013, pp. 115

[7] Collofello K., Vehathiri K., "An Environment for Training Computer Science Students on Software Testing," in *Frontiers Education 35th Annual Conference*, Indianopolis, 19-22 Oct. 2005, pp. T3E – 6

[8] Rex Black, Erik Van Veenendaal, Dorothy Graham, "Testing Throughout The Software Lifecycle," in *Foundations of Software Testing ISTQB Certification*, 3rd ed., 2012, pp.27.

[9] Ilene Burnstein, "Introduction to Testing as an Engineering Activity," in *Practical Software Testing: A Process-Oriented Approach Springer Professional Computing*, , ISBN: 978-0-387-95131-7, 2003, pp. 4-5

[10] Ilene Burnstein, "Testing Fundamentals," in *Practical Software Testing: A Process-Oriented Approach Springer Professional Computing*, ISBN: 978-0-387-95131-7, 2003, pp.19

[11] Rex Black, Erik Van Veenendaal, Dorothy Graham, "Test Design Techniques", in *Foundations of Software Testing ISTQB Certification*, 3rd ed., 2012, pp.71

[12] Lee Copeland, "Exploratory Testing," in *A Practitioner's Guide to Software Test Design*, 2004, pp. 202

[13] Cem Kaner.(2008), A Tutorial in Exploratory Testing, [Online]. Available: http://www.kaner.com/pdfs/QAIExploring.pdf

[14] Rex Black, Erik Van Veenendaal, Dorothy Graham, "Testing Levels," in *Foundations of Software Testing ISTQB Certification*, 3rd ed., 2012, pp. 42

[15] Fu-Shiau Li ,Wei-Ming Ma, Chao A., "Architecture Centric Approach to Enhance Software Testing Management," *in Intelligent Systems Design and Applications IEEE Eighth International Conference,* Vol. 1, Kaohsiung, 26-28 Nov. 2008, pp.654-659

[16] Carnliner, Saul, "Workshop in Conducting Integrative Literature Reviews," in *Professional Communication Conference (IPCC) 2011 IEEE International,* Cincinnati, OH, USA, 17-19 Oct. 2011, pp. 1-3

[17] Itkonen, J. , Mäntylä , M.V., Lassenius, C., "How do testers do it? An Exploratory Study on Manual Testing Practices," in *Empirical Software Engineering and Measurement ESEM 2009. 3rd IEEE International Symposium*, 15-16 Oct. 2009, Lake Buena Vista, pp. 494 – 497

[18] Kumar, S., Wallace, C., "Guidance for exploratory testing through problem frames", in *Software Engineering Education and Training (CSEE&T) IEEE 26th Conference* , 19-21 May 2013, San Francisco, CA, pp. 284-288

[19] T. Sviridova, Lviv, D. Stakhova, U. Marikutsa, "Exploratory testing: Management Solution," in *Experience of Designing and Application of CAD Systems in Microelectronics (CADSM) 12th International Conference*, Polyana Svalyava,19-23 Feb. 2013, pp. 361

[20] Henrik Andersson.(2009), "Implementing Exploratory Testing at large organization," [Online].Available:http://itknowledgeexchange.techtarget.com/software-quality/eight-days-80-testers-exploratory-testing-case-study-at-cast-2009/

[21] V. Prakash, S. Gopalakrishnan, "Testing efficiency exploited: Scripted Versus Exploratory Testing," Vol. 3**,** 8-10 April 2011, Kanyakumari, pp. 168 - 172

[22] Juha Itkonen, Mika V. Mantyla and Casper Lassenius, "Defect Detection Efficiency: Test Case Based vs. Exploratory Testing," in *IEEE First International Symposium on Empirical Software Engineering and Measurement*, 20-21 Sept. 2007, Madrid, pp. 61-70

[23] Juha Itkonen and Kristian Rautiainen, "Exploratory Testing: A Multiple Case Study," in *IEEE First International Symposium on Empirical Software Engineering and Measurement*, Queensland, Australia, 17-18 Nov. 2005

[24] Juha Itkonen, Mika V. Mantyla, Casper Lassenius "The Role of the Tester's Knowledge in Exploratory Software Testing," in *IEEE Transactions on Software Engineering*, May 2013, Vol. 39, No. 5, pp. 707-724

[25] L. Shoaib, A. Nadeem, A. Akbar "An Empirical Evaluation of the Influence of Human Personality on Exploratory Software Testing," in *IEEE 13th International Multitopic Conference INMIC* , 14-15 Dec. 2009, Islamabad, pp-1-6

[26] Wasif Afzal et al., "An experiment on the effectiveness and efficiency of exploratory testing", Empirical Software Engineering journal, Print ISSN 1382-3256, Online ISSN 1573-7616 Springer US, April 2014, pp. 267

[27] Syed Muhammad Ali Shah, Marco Torchiano, Antonio Vetro, Maurizio Morisio, "Exploratory testing as a Source of Testing Technical Debt," in *IT Professional IEEE Computer Society Digital Library*, 08 March 2013, pp.25

[28] S.P. Ng, T. Murnane, K. Reed, D. Grant, T.Y. Chen, "A Preliminary Survey on Software Testing Practices in Australia," in *Australian Software Engineering Conference*, 2004, Australia, pp. 116-125

[29] Deak A., Stalhane T. ,"Organization of testing activities in Norwegian Software Companies," in *Software Testing Verification and Validation Workshops(ICSTW) IEEE Sixth International Conference* ,18-22 March 2013, Luxembourg, pp. 102-107

[30] Adnan Causevic, Daniel Sundmark, Sasikumar Punnekkat, "An Industrial Survey on Contemporary Aspects of Software Testing," in *Software Testing, Verification and Validation (ICST) Third International Conference*, 6-10 April 2010, Paris, pp.393- 401

[31] Mei YongGang , Xi'an, Ding JianJie, "Software Measurement Process Capability Maturity Model," in *Computer Modeling and Simulation, ICCMS 2010 Second International Conference*, 22-24 Jan. 2010, Sanya, Hainan pp. 400-402

[32] Barbara Kitchenham, "What's up with Software Metrics?- A Preliminary Mapping Study," in *Journal of Systems and Software*, Vol. 83, Issue 1, Jan 2010, pp. 37–51

[33] Yang Wu,Ying Peng, "The 80/20 rule in enterprise business management application," in *IEEE 2nd International conference of Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC)*, Deng Leng, China,8-10 Aug. 2011, pp.1613

[34] M. Iqbal, M. Rizwan, "Application of 80/20 rule in software engineering Waterfall Model," in *International Conference on Information and Communication Technologies*, Karachi, 15-16 Aug. 2009, pp. 225

[35] Chi-Ming Chung, Tamshui, "Software development techniques-Combining Testing and Metrics," in *IEEE Computer and Communication Systems 10th International Conference*, 24-27 Sep 1990, Vol. 1, Taiwan, pp- 424 - 428

[36] Joseph R. Horgan, Michael R. Lyu , Saul London, "Achieving Software Quality with Testing Quality Measures," in *IEEE Computer*, Vol. 27, Issue 9, Aug 2002, pp. 60-69

[37] Mitrabinda Ray, Durga Prasad Mohapatra, "Risk analysis: A Guiding Force in the Improvement of Testing," in *IET Software*, doi: 10.1049/iet-sen.2011.0081, Vol. 7, Issue 1, February 2013, p. 29-46

[38] Alasdair McAndrew,Anne Venables, "A Secondary Look at Digital Image Processing," *in Proc. 36th SIGCSE technical symposium on Computer science education*, 2005, pp. 337-338

[39] You-yi Zheng, Ji-lai Rao, Lei Wu , "Edge Detection Methods in Digital Image Processing," in *Proc. IEEE Computer Science and Education 5th International Conference*, 24-27 Aug. 2010, Hefei, pp. 471 – 473

LIST OF PUBLICATIONS

PUBLISHED

i. Nishtha Hooda and Vinod K. Bhalla, "*Exploratory Testing: A Review and Questionnaire Investigation***"** in *Engineering Sciences International Journal, International Conference on Mathematics and Engineering Sciences -2014, Chitkara University, Chandigarh, March 2014*.

ii. Nishtha Hooda and Vinod K. Bhalla, " 2014 Testing Trend Is Not Purely Ad hoc: Exploratory Testing Algorithmic Design and Metrics Implementation" in Proc. *Elsevier International Conference* on *Emerging Research in Computing, Information, Communication and Applications,* 1st Aug. 2014, Bangalore.

COMMUNICATED

i. Nishtha Hooda and Vinod K. Bhalla ,"2014 Trend of Software Testing: Survey on Testing Practices and Exploratory Testing Usage", *Springer Symposium on Search-Based Software Engineering*, Brazil

ii. Nishtha Hooda and Vinod K. Bhalla, "Exploratory Testing by Statistical Analysis" in *International Journal of Advanced Research in Computer Science and Software Engineering* (IJARCSSE)