Integrating Exploratory Testing In Software Testing Life Cycle
A Controlled Experiment

Tanveer Ahmed
Chejerla Madhusudhana Raju

School of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona
Sweden

Internet : www.bth.se/com
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57
This thesis is submitted to the School of Engineering at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering. The thesis is equivalent to 30 weeks of full time studies.

Authors:
Tanveer Ahmed
Address: Karlskrona, Sweden
Email: tanveershaik4u@gmail.com.

MadhuSudhanaRaju.Chejerla
Address: Karlskrona, Sweden
Email: msraju2009@gmail.com

University advisor:
Dr. Wasif Afzal
School of Computing
Blekinge Tekniksa Hogskola, Sweden
Abstract

Context. Software testing is one of the crucial phases in software development life cycle (SDLC). Among the different manual testing methods in software testing, Exploratory testing (ET) uses no predefined test cases to detect defects.

Objectives. The main objective of this study is to test the effectiveness of ET in detecting defects at different software test levels. The objective is achieved by formulating hypotheses, which are later tested for acceptance or rejection.

Methods. Methods used in this thesis are literature review and experiment. Literature review is conducted to get in-depth knowledge on the topic of ET and to collect data relevant to ET. Experiment was performed to test hypotheses specific to the three different testing levels: unit, integration and system.

Results. The experimental results showed that using ET did not find all the seeded defects at the three levels of unit, integration and system testing. The results were analyzed using statistical tests and interpreted with the help of bar graphs.

Conclusions. We conclude that more research is required in generalizing the benefits of ET at different test levels. Particularly, a qualitative study to highlight factors responsible for the success and failure of ET is desirable. Also we encourage a replication of this experiment with subjects having a sound technical and domain knowledge.

Keywords: Exploratory testing, Experiment, Effectiveness, Software testing life cycle.
ACKNOWLEDGEMENT

In the name of Allah most beneficent and most merciful

All praise is due to Allah so we praise him and we seek his refuge and we bear witness that Muhammad saw is his messenger and devotee.

At the outset, I am grateful to Almighty Allah who has bestowed me with his mercy and blessings to complete my thesis.

We would like to thank our parents and my siblings for their unconditional love and support. We would like to thank Dr. Wasif Afzal whose support and help has been substantial in completing the master thesis. Finally we would like to thank our joint family, friends and fellow students for their moral and emotional support.

- Tanveer Ahmed
- Chejerla Madhusudhana Raju
Table of Contents

INTEGRATING EXPLORATORY TESTING IN SOFTWARE TESTING LIFE CYCLE........1
A CONTROLLED EXPERIMENT .............................................................................1
ABSTRACT ...........................................................................................................2
ACKNOWLEDGEMENT .........................................................................................3

1 PROLOGUE .......................................................................................................8
  1.1 INTRODUCTION ............................................................................................8
  1.2 MOTIVATION ................................................................................................9
  1.3 RELATED WORK ..........................................................................................9
  1.4 AIMS AND OBJECTIVES .............................................................................10
  1.5 RESEARCH QUESTION ...............................................................................10
  1.6 THESIS STRUCTURE ..................................................................................11
  1.7 ABBREVIATIONS/GLOSSARY .....................................................................12

2. BACKGROUND ..................................................................................................13
  2.1 SOFTWARE TESTING .................................................................................13
  2.2 LEVELS OF TESTING ................................................................................14
      2.2.1 V model of Testing ............................................................................15
  2.3 EXPLORATORY TESTING ...........................................................................17
      2.3.1 Benefits of ET ..................................................................................19

3. EXPERIMENT .....................................................................................................20
  3.1 EXPERIMENT DEFINITION ..............................................................20
      3.1.1 Object of the study .............................................................................20
      3.1.2 Purpose ..............................................................................................20
      3.1.3 Quality focus ....................................................................................21
      3.1.4 Context ...............................................................................................21
      3.1.5 Perspective .........................................................................................21
  3.2 EXPERIMENT PLANNING .................................................................21
      3.2.1 Context Selection .............................................................................21
      3.2.2 Hypothesis formulation ....................................................................21
      3.2.4. Selection of subjects ......................................................................22
  3.3 EXPERIMENTAL DESIGN ...........................................................................22
      3.3.1 Experiment design type ....................................................................23
  3.4 INSTRUMENTATION ..................................................................................24

4. RESULTS AND ANALYSIS ...............................................................................26
  4.1 RESULTS ......................................................................................................26
  4.2 NORMAL DISTRIBUTION ........................................................................28
  4.3 UNIT LEVEL USING WILCOXON TEST .............................................29
      4.3.1 Hypothesis testing at Unit test level .................................................31
  4.4 INTEGRATION LEVEL USING TWO SAMPLE T-TEST ......................32
      4.4.1 Hypothesis testing at Integration level .............................................34
  4.5 SYSTEM LEVEL USING WILCOXON TEST .......................................35
      4.5.1 Hypothesis testing at System test level ............................................37

5. DISCUSSION .....................................................................................................39
  5.1 OBSERVATION .............................................................................................39

4
List of tables:

Table 1: Experiment design ................................................................. 23
Table 2: Descriptive statistics of experiment ........................................ 26
Table 3: Overview of parametric/non-parametric tests for different designs .................................................................................................................. 27
Table 4: Tests of Normality (unit level) ............................................... 28
Table 5: Tests of Normality (integration level) ..................................... 28
Table 6: Tests of Normality (system level) ......................................... 28
Table 7: Hypothesis Summary at Unit level ........................................ 31
Table 8: Hypothesis Summary at Integration level .............................. 33
Table 9: Hypothesis Summary at System level .................................... 36
Table 10: Assessment form ................................................................. 47
Table 11: Defect Count Report ............................................................. 47
Table 12: Representation of different software test levels in our experiment ................................................................................................. 48
Table 13, 14 & 15: Wilcoxon statistical test at Unit Level ................... 29
Table 16, 17: Two sample T-test at Integration Level ......................... 33
Table 18, 19 & 20: Wilcoxon statistical test at System Level ............... 36
Table 21: Seeded Defects Report ......................................................... 48
Table 22: Seeded Defects Source code ............................................... 51
List of figures

Figure 1: Testing hierarchy [14] ................................................................. 14
Figure 2: V model .................................................................................. 15
Figure 3: Experiment execution .............................................................. 25
Figure 4: Bar graph visualizing Unit Level Defects ............................... 32
Figure 5: Bar graph visualization of integration level defects ................ 35
Figure 6: Bar graph visualization of system level defects ....................... 51
Figure 7: Screen shot of web application used in the experiment .......... 65
Figure 8: Screen shot of web application used in the experiment .......... 66
Figure 9: Screen shot of web application used in the experiment .......... 67
Figure 10: Screen shot of web application used in the experiment ........ 68
1 PROLOGUE

1.1 Introduction

Software Testing is considered as one of the vital phases in Software Development Life Cycle due to huge amount of cost involved into this phase [2]. In Software testing there are two types of testing techniques, manual and automated. Manual testing [MT] helps finds defects manually without the help of tool. Other techniques like test automation are complementary techniques which avoid repetitiveness and create room for creative MT [3][ 6].

Exploratory testing is a kind of manual testing practice in software testing. James Bach [5] defined Exploratory testing as “simultaneous learning, test design and test execution”, Also “ET is an approach to testing that emphasizes the freedom and responsibility of each tester to continually optimize the value of his work done by treating learning, test design, and test execution as mutually supportive activities that run in parallel throughout the project [5]” . Exploratory testing is quite opposite to traditional testing approaches which use the concept of predefined test cases. ET stresses the need of using testers’ skills rather relying on predefined test cases [8].

Earlier exploratory testing was called as ad hoc testing [5]. Exploratory testing only relies on tester skills and do not follow any particular procedure for testing. In Exploratory testing human intelligence and other human personality traits play an important role in determining the bugs [12]. In Exploratory testing there is no lengthy documentation.

There are numerous studies about effectiveness of ET [11] but knowledge of these studies is limited to utilizing tester’s skills. Our purpose in this thesis to conduct an experiment which evaluates the effectiveness of integrating exploratory testing (ET) with software testing life cycle (STLC) more specifically targeting three levels of STLC i.e., unit level, integration level and system levels. The next section is motivation for doing the thesis.
1.2 Motivation

Software testing is necessary in software development life cycle since it saves costs and improves quality of the overall product. ET is a manual approach to software testing which aims at finding defects; its main purpose is to continually optimize finding defects and putting less effort on documenting, planning and scripting [11]. Reports in literature have suggested that ET can be orders of magnitude more efficient than scripted testing [5].

There are prolific studies by various authors in the field of software testing but there are few articles on ET. Much of the literature about ET is available through text books, practitioner reports, and electronic material on the internet [11]. Practitioner reports have argued about ET to be effective and cost efficient [3, 35, and 39]. There are studies in literature which have mentioned about the effectiveness and efficiency [33, 34, 35, 36] of ET which has to be further investigated [11]. There is need to investigate various aspects of exploratory testing, experience and the knowledge involved in it [47]. Our current knowledge about ET and its very usage with the software test levels is very limited. These things are to be further explored in terms of incorporating ET with Software test levels. In this thesis we seek to evaluate the effectiveness of integrating ET at different software test levels.

1.3 Related Work

Prakash and Gopalakrishnan [37] conducted an experiment for testing efficiency of exploratory testing and scripted testing. The experiment was based on simple calculator to test its functionality. The result of the experiment indicated that there was no use of predesigned test cases when testing. Juha Itkonen et al. [5] researched about the defect detection efficiency of ET and test case based testing by performing a controlled experiment with software engineering students. They emphasized the fact that there were no underlying differences in the type of defects detected, severities and difficulty in detection.

Andhy Tinkham et al. [7] have argued about the different learning styles of testers that are used to find defects with the help of ET. They are stressing the point, that tester’s personality and skills affects process of ET. Jorg Denzinger [8] proposed a new method in ET that uses tool-support by a tester to detect unwanted behaviour. This was done with the help of two case studies, which helped in identifying not previous known defects. According to [38], testers need to apply testing techniques when they are applying experience based exploratory testing approaches. The main
of their study was to improve understanding of exploratory testing in terms of level functional testing.

Agruss et al. [9] emphasized the fact that chances of finding new defects decreases substantially once a predesigned test case is run. Testers often do not depend on test strategies but rather apply various techniques to uncover defects; these strategies are not found in predesigned test cases [9]. Pyhajarvi et al. [10] performed an action research about exploratory testing. They have presented the use of cycles of time framework to integrate testing into development.

Phil Laplante [12] in his research has explored different needs of ET in avionics and space shuttle systems by doing different explorations like environmental, input, output, behavioural, language etc. Research is still going on in ET to test its suitability and applicability in different areas. The next section explains the aims and objectives of the experiment.

1.4 Aims and Objectives

The aim of this thesis is to comment on the effectiveness of ET in detecting defects at the three testing levels of unit, integration and system. This aim will be met by testing three hypotheses aimed at finding the defect detection ability of ET at the three levels of unit, integration and system testing.

By doing this we will be able to analyse the effectiveness of ET in each test level.

1.5 Research Question

We seek to answer the following research question in this thesis:

RQ: Can ET detect all the defects seeded at unit, integration and system levels of testing?

This research will focus on effectiveness of ET in detecting seeded defects at different software test levels. The motivation for selecting effectiveness is that there are studies which state that there is need for investigating effectiveness and efficiency in ET [11].
1.6 Thesis Structure

Chapter 1(Introduction): This section contains introduction to exploratory testing, aims and objectives, research questions and motivation for doing thesis.

Chapter 2(Background): This section highlights the background and literature survey in connection with ET. It also summarises about testing, different software test levels and explanation about ET.

Chapter 3(Research methodology): This section of master thesis contains the experiment part, which was conducted on the guidelines of wohlin et al. In this section there are objectives of the experiment, experiment definition, planning, design and execution.

Chapter 4(Results and Analysis): The fourth chapter of this thesis document presents the results of the experiment and its analysis. The analysis of the experiment was done based on the descriptive statistical methods.

Chapter 5(Discussion): This chapter of the master thesis presents discussion based on the results achieved and further course of research that can be carried out. Validity threats associated with the thesis are also discussed in this section.

Chapter 6(Conclusion): This chapter provides final conclusion of the thesis. The outcome of the thesis is discussed in this part of the master thesis.

Chapter 7(Reference)

Chapter 8(Appendix)
# 1.7 Abbreviations/Glossary

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STLC</td>
<td>software test life cycle</td>
</tr>
<tr>
<td>UT</td>
<td>Unit testing</td>
</tr>
<tr>
<td>IT</td>
<td>Integration testing</td>
</tr>
<tr>
<td>ST</td>
<td>System testing.</td>
</tr>
<tr>
<td>MT</td>
<td>Manual testing</td>
</tr>
<tr>
<td>SUT</td>
<td>Software under test</td>
</tr>
<tr>
<td>ET</td>
<td>Exploratory testing</td>
</tr>
</tbody>
</table>
2. BACKGROUND

The field of software testing is always looking for cost effective methods in its quest to ensure quality in all the phases. Software testing is considered to be one of the vital phases in Software Development Life Cycle (SDLC). A huge amount of cost is involved in this phase [1]. The defects found by Manual Testing (MT) are mostly new and other techniques like test automation are complementary techniques that avoid repetitiveness and create room for creative MT [2][4].

The following sections explain the notion of software testing and approach to testing i.e., exploratory testing and different software test levels and related work.

2.1 Software testing

Software testing is also called as program testing and is part of the process verification and validation. This is done with the aim of finding faults in the software system. This should be included early in the process of software development to ensure high quality and to save costs. Researchers have tried to define software testing according to the usage. Here are few of the definitions.

Software testing is the process of uncovering evidence of defects in software system [17].

Testing shows the presence, not the absence of bugs [16].

The process of operating a system or component under specified conditions observing recording the results, and making an evaluation of some aspect of the system or component [15].

Software testing is an empirical, technical investigation conducted to provide stakeholders with information about the quality of the product under test [18].

Testing is the process of analyzing a software item to detect the difference between existing and required conditions and to evaluate the features of the software item [30].

Software testing follows a hierarchal approach; this was reported by [14], on the topmost level there are different levels of software testing cycle, in the next level methods followed by the types. Our thesis mainly deals with the levels of software testing and seeking the possibility incorporating exploratory testing into these levels.
A Simple classification of testing consists of two methods black box, white box. Black box testing is also called as specification based testing and white box testing is also called as structure based testing. On the hierarchy of testing these methods are at the second level [14].

The bottom level of testing hierarchy consists of testing types, for example Functional, Algorithmic, Positive tests, negative tests etc.

- Functional: “Tests designed to determine if specific functions/features work as specified” [14].
- Algorithmic: “Tests designed to determine if specific algorithms have been implemented correctly” [14] etc.

Apart from these test types there are also there are many others. The following figure gives the overview of testing hierarchy,

![Testing hierarchy figure]

Figure 1: Testing hierarchy [14]

### 2.2 Levels of testing

Companies use v model to demonstrate testing at different levels, however they don’t follow sequential process. There are four software test levels viz. Unit, Integration, System and Acceptance. Our primary concern is to find the effectiveness of integrating exploratory testing at unit, integration and system levels. ET is not appropriate to use with acceptance testing it can be used elsewhere [9].
2.2.1 V model of Testing

The V model of testing is often mistaken with the waterfall. The v model is an extension of the simple waterfall model where each test level is concerned with each process phase [10]. There are many variants of v model as reported by [27][28]. In v model of testing the test levels unit and integration testing are considered as low levels of testing [29], whereas system and acceptance testing are considered as high levels of testing [10].

The motivation for selecting v model is that it is state of art and practically taught on all the courses [10]. Therefore it becomes necessary to study about v model and different levels in it. The following figure 2 is v model, after each test level there is regression testing. Our main concerns in the thesis are the software test levels unit, integration and system.

![V-model of testing](image)

Figure 2: V-model of testing

2.2.1.1. **Unit Testing:** Unit testing (UT) is the lowest level of testing in Software testing life cycle [21]. The smallest part of code is check against requirements in unit testing [19]. It has been widely used after the introduction of object oriented languages like java and C++ [19]. UT is vital part of software development practice and if units are incorrect then the chances of build reliable systems narrows down [20].

**Objective:** The primary objective of unit testing is to detect defects in logic, data by testing individual modules [14].
Examples of units are classes, functions and interfaces. The following are the defects which could occur at unit testing.

- Incorrect data type
- Testing function procedures or event handling
- Variable declared but never used
- Mismatching of data like for integer type floating type data is given
- For loop(index issues)
- While loop(end condition might be wrong)

2.2.1.2 Integration testing (IT): It is the next of level of testing after UT. Integration testing is the process of verifying interactions between system components [23][25]. The main purpose of IT is to test behavior of components as a whole [24]. It aims at building program structures besides testing the interfaces [26]. It is most useful and effective technique for performing testing [22].

Objective of IT: To find defects in interfaces between units [14]. The following are the sample defects which could occur at integration testing.

- Correct call from each other functions
- Transfer right data
- Access interfaces at right time
- Check correct state of component

2.2.1.3 System Testing (ST): The third level of software testing is System testing. In this subsystems are integrated to make up whole system and then tested [26].

Objective: The objective of system testing is to verify whether all requirements are met as earlier defined [14].

The following are the defects which could occur at system testing

- Incorrect/absence of data
- Actual functionality of code is missing
- Logic or calculation of data is incorrect or inconsistent.
- Spelling errors in pages
Acceptance testing is the conformance of requirements by the end customers or it is their approval for what has been delivered [31]. In our thesis as there are no customers we have ignored the acceptance testing part of the v model.

2.3 Exploratory testing

In the recent past Exploratory testing (ET) has been gaining tremendous importance due to responsibility of individual tester in finding defects early in the software test cycle. ET is a manual testing practice in software testing. Many authors have defined ET but the most widely accepted one is given by James Bach. According to Bach [3] ET is simultaneous learning, test design and test execution. He also explained ET as Any testing to the extent that the tester actively controls the design of the tests as those tests are performed and uses information gained while testing to design new and better results[3]. Another established definition was given by kaner [18] in which he defines ET as an approach to testing that emphasizes the freedom and responsibility of each tester to continually optimize the value of his work done by treating learning ,test design, and test execution as mutually supportive activities that run in parallel throughout the project.

ET is quite opposite to traditional testing approaches that use the concept of predefined test cases. ET stresses the need for using testers’ skills rather relying on predefined test cases [5]. Earlier ET was called as ad hoc testing [3], due to its close nature to ad hoc testing. ET only relies on tester skills and does not follow any particular procedure for testing. There is no lengthy documentation in ET. Human intelligence and other human personality traits play an important role in detecting defects [6]. Even though ET looks like unplanned approach it is still structured and planned approach in testing [10]. There have always been misconceptions associated about ET that were proven false and are reported in [32]. In fact ET is promoted as a valid approach and valuable part of effective set of quality assurance practices [11]. The following were the characteristic properties of ET that were reported by [32].

- ET provides complete coverage of the application.
- ET is a structured approach.
- ET doesn’t lack transparency and visibility.
- ET requires less time than scripted testing.
- ET can be applied to complex systems.
- Organizations always practice some kind of ET.
• ET testers need training.
• ET testers have same skill set as that of scripted tester.

Testers in exploratory testing learn about the product, its weakness, about the market of the product, specific areas in which the product could fail as they progress in the testing of the product [5]. The following characteristics of ET were identified and reported by [45] as

• Interactive
• Creativity
• Drive down fast results
• Concurrence of cognition and execution

ET is a structured process conducted by a skilled tester, or by less skilled testers or users working under reasonable supervision [35]. The structure of ET comes from the following [35]

• Test design heuristics
• Chartering
• Time boxing
• Perceived product risks
• The nature of specific tests
• The structure of product being tested
• The process of learning the product
• Development activities
• Constraints and resources offered by the project
• The skills, talents, and interests of the tester
• The overall mission of testing

Juha Itkonen and Rautiainen [11], have discussed the following five properties of ET.

• ET is testing without mission, instructions and with test cases not given in advance.
• When a tester uses user manual his results are guided by earlier performed tests.
• ET aims at detecting defects instead of using comprehensive set of test cases.
• It is simultaneous learning of system under test, test design and execution.
• ETs effectiveness relies on tester’s knowledge, skill and experience.

2.3.1 Benefits of ET

Apart from these characteristics ET has many benefits which are reported by [3][11][40][5][11] such as

• Rapid feedback
• Quick learn ability about the product
• Diversifying the testing process
• Finding important bugs in short time
• Cross check of other tester work
• Isolation of particular defect
• Non reliance on documentation.

In software testing community, there are many manual testing practices of which exploratory testing saves a lot of resources for the organization. In addition to the above benefits there are additional advantages of ET [3][31][40] such as

• Improvising on scripted tests
• Interpreting vague test instructions
• Product analysis and test planning
• Improving existing tests
• Writing new test scripts
• Regression testing based on old bug reports
• Testing based on reading the user manual and checking each assertion.

Some of the short comings of exploratory testing are listed below [11]

➢ Difficulty of tracking the progress of individual testers
➢ It is considered hard to find out how work proceeds because there is no planned low-level structure that could be used for tracking the progress.
➢ ET has no capability of preventing the defects.
➢ Scripted testing has the ability to detect defects early at requirements gathering and design level.
3. EXPERIMENT

Experiments are launched when we want to control the situation and manipulate behavior. The effect of the manipulation is measured and based on this, statistical analysis is performed. To access and compare the effectiveness of different techniques, controlled experiments are powerful [48]. Our main idea of experiment is from the future work of [11], which says that experiments are to be conducted with students to know the effect of using different techniques in exploratory testing. We would like to experiment exploratory testing at different test levels to know how ET works and test its efficiency. Therefore Experiment is the most suitable research methodology to know the cause and effect of ET at different test levels for our research. After performing literature review we came to know that there is lack of study on ET with software test levels. Therefore experiment was conducted keeping in mind it is useful to practitioners and researchers in academia as well as industry people. The thesis is suitable to researchers as well as Practitioners in field of software testing and exploratory testing who are looking to explore new horizons of ET with software test levels. The experiment is performed as per the experiment process as suggested by Wohlin et al. [13].

Pilot Experiment

A pilot experiment was conducted with the help of two industry practitioners. All the objects, instruments and tools were provided to them and then they were asked to find defects in the code. The average time of the two practitioners was considered for the final experiment in the thesis.

3.1 Experiment Definition

The aim of this experiment is to assess the effectiveness of ET in detecting defects at the three testing levels of unit, integration and system.

3.1.1 Object of the study

According to Wohlin et al. [13], an object of study is the entity that is studied in the experiment. The object in our experiment is ET which is being tested for its effectiveness at the three testing levels of unit, integration and system. The object in our experiment could also be taken as the software under test (SUT), which in our case is a website developed specifically for the purpose of meeting our objectives.

3.1.2 Purpose

"The purpose of the experiment is to test if ET is able to find all the seeded defects at the three levels of unit, integration and system levels of testing."
3.1.3 Quality focus

The primary effect under study in this experiment is the effectiveness of ET in finding seeded defects at the three levels of unit, integration and system testing.

3.1.4 Context

The subjects in the experiment are students of MSc Computer Sciences at Blekinge Institute of Technology, Sweden. The subjects will be testing a web application at the three levels of unit, integration and system. We can characterize our experimental context as "multi-test within objects" study [13] as we are examining a single object across a set of subjects.

3.1.5 Perspective

The experiment results are interpreted from the viewpoint of a researcher.

Based on the above elements of an experiment definition, we can now define our experiment:

*Analyze ET, for the purpose of evaluating whether it can detect all the seeded defects at the three levels of unit, integration and system test (with respect to ET’s effectiveness) from the point of view of the researcher in the context of a multi-test within objects study.*

3.2 Experiment Planning

3.2.1 Context Selection

The context of our experiment is on-line with students as subjects. We have a scaled-down website as our SUT.

3.2.2 Hypothesis formulation

According to Wohlin et al. [13] hypothesis formulation is basis for statistical analysis in an experiment. Experiment definition is formulated into two hypothesis null and alternative hypotheses. In null hypothesis there are no underlying trends in the experiment. Alternative hypothesis is accepted when null hypothesis is rejected. The following are the null and alternative hypotheses for our experiment.
3.2.2.1 Null Hypotheses, $H_0$

ET detects all the defects seeded at Unit, Integration and System level.

$H_0$1: ET detects all the defects seeded at unit level.
$H_0$2: ET detects all the defects seeded at integration level.
$H_0$3: ET detects all the defects seeded at system level.

3.2.2.2 Alternative Hypotheses, $H_1$

ET does not detect all the defects seeded at Unit, Integration and system levels of testing.

$H_1$1: ET does not detect all the defects seeded at unit level.
$H_1$2: ET does not detect all the defects seeded at integration level.
$H_1$3: ET does not detect all the defects seeded at system level.

3.2.3. Selection of subjects

The subjects for the experiment were Master students in software engineering with Blekinge Institute of Technology. The subjects were selected based on the pre-assessment conducted by the authors. Subjects with the basic knowledge of .NET and who have done the course verification and validation were chosen to test the web application. The motivation for choosing subjects from this course is that students in this course are taught testing methods.

A brief explanation was given to the students for a period of about 15 minutes in the form of user guide (See appendix B for user guide). The user guide includes explanation about software test levels and the testing technique they are going to use in the experiment. Subjects were then asked to test a web application (see appendix c) which was a purchase management system for a period of sixty minutes.

3.3 Experimental design

There are three important design principles as in [13]: randomization, blocking and balancing.

Randomization applies on the allocation of objects, subjects and in which order the tests are performed [13]. The selection of subjects for our experiment was pseudo-random, meaning we selected subjects based on a screening. The screening was done to ensure the subjects had knowledge of .NET, testing levels and methods. Since
there was only one treatment at a time (i.e., testing at a particular level) and only one software under test (SUT), all the subjects were given the same treatment and SUT at testing at a particular level.

Blocking is used to eliminate the undesired effect in the study [13]. We used blocking in our experiment by screening the subjects, as described in the last paragraph.

Balancing is about assigning each treatment to an equal number of subjects. Our design is balanced as each treatment was assigned to an equal number of subjects.

### 3.3.1 Experiment design type

There are basically four standard design types for which statistical analysis methods are suitable. They are

- One factor with two treatments
- One factor with more than two treatments
- Two factors with two treatments
- More than two factors each with two treatments

Our experimental design type is one factor with more than two treatments. The factor is exploratory testing while the treatments are three, one each corresponding to the three levels of testing.

Table 1 gives an overview of the design type used in the experiment.

**Table 1: Experiment design**

<table>
<thead>
<tr>
<th>Factor</th>
<th>Subjects</th>
<th>Treatment</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1.Unit</td>
</tr>
<tr>
<td>Exploratory testing</td>
<td>35</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>X</td>
</tr>
</tbody>
</table>
3.4 Instrumentation

There are three types of instruments which are included in the planning phase of an experiment namely objects, tools and guidelines, these are prepared beforehand of the experiment [13].

- **Objects:** The object in the experiment was web application which was developed by the authors which contained induced defects by the authors. The web application was developed using C#, ASP. Net and for the back end of the web application SQL server database was used.

- **Guidelines:** At the outset, the subjects were given preliminary training about exploratory testing and software test levels in the form of user guide see appendix B. The tasks for subjects were made clear and their requirements regarding the web application testing i.e. purchase management system testing were made clear.

- **Measurements:** Data collection was done in the form of word files (see appendix A), finally the data collected from the each subject was converted in excel for analysis.

3.5 Experiment execution

The experiment was conducted with 35 Master students in Software engineering and computer science at Blekinge Institute of Technology. A total of fifty students were invited to experiment out of which 40 responded. Based on the pre assessment 35 students were finalized for the experiment. The date of the experiment was then informed to the participants. All the instruments, i.e., objects, guidelines and forms (see appendix A) necessary were available before the start of the experiment.

The total duration of the experiment was 75 minutes. Out of which first fifteen minutes of the experiment were for training. The subjects (students) were then explained about goal of the experiment, ET and test levels. The remaining sixty minutes were for finding the defects at three levels UT, IT and ST. Thirty minutes were given to identify defects at unit level, fifteen for Integration level and fifteen for System level.

The experiment was conducted on two occasions due to the fact that paying attention to thirty five people at once was difficult. On the first occasion twenty students
participated and in the next fifteen. Since ET is a random approach no test cases were provided to the subjects. The subjects then recorded the bugs in the sheet. Figure 3 gives an overview of experiment execution.
4. RESULTS AND ANALYSIS

4.1 Results

In this section the results are analyzed. The data of the result is analyzed by using design from experiment which is “one factor with more than two treatments”. For doing statistical tests, we initially performed normality on the data sets obtained after applying ET on different software test levels. Normality of different software test levels is identified using Shapiro-Wilk tests. The level of significance is set to 0.05 [13]. By using normality tests, parametric and non-parametric are classified and selection of appropriate statistical tests is decided [13]. In our case, Unit and System level tests come under non-parametric as their significant value is (Unit Level .002, System Level .012) and Integration level comes under parametric tests as its significant value is .005. For parametric tests we used two sample t-test and non-parametric tests used is Wilcoxon test [13]. We used two sample t-test to analyze the results of Integration level and Wilcoxon test to analyze results of Unit and Integration levels. The qualitative data collected after the experiment is given in Table 2.

Table 2: Descriptive statistics of experiment

<table>
<thead>
<tr>
<th>Subjects</th>
<th>Seeded Defects at Unit level</th>
<th>Defects after experiment at Unit level</th>
<th>Seeded Defects at Integration level</th>
<th>Defects after Experiment at Integration level</th>
<th>Seeded Defects at System level</th>
<th>Defects after experiment at System level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>17</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td>13</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
<td>15</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>20</td>
<td>8</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>11</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>20</td>
<td>17</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>20</td>
<td>11</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>14</td>
<td>20</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>20</td>
<td>12</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>20</td>
<td>19</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>18</td>
<td>20</td>
<td>12</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Design</td>
<td>Parametric</td>
<td>Non-Parametric</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------------</td>
<td>----------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>One factor, one treatment</td>
<td>t-test</td>
<td>Mann-Whitney</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>One factor, two treatment, completely</td>
<td>F-test</td>
<td>Chi-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>randomized design</td>
<td></td>
<td>Binomial Test</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>One factor, two treatment, paired</td>
<td>Paired t-test</td>
<td>Wilcoxon</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>comparison</td>
<td></td>
<td>Sign Test</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>One factor more than two treatments</td>
<td>ANOVA</td>
<td>Kruskal-Wallis</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chi-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>More than one factor</td>
<td>ANOVA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Overview of parametric/non-parametric tests for different designs [13]

The experiment design type is one factor with more than two treatments for the whole of experiment, but since we analyzed the individual test levels statistically, so we reduced our treatments to any one of unit, integration and system, one at a time. Therefore Wilcoxon rank sum test is more suited for a non-parametric test. For parametric test, two sample t-test is suitable.
4.2 Normal Distribution

Normal distribution in general is identified as normal curve which looks like “Bell-shape”. Generally normal distributions differ with respect to means and standard deviations [42]. Skewness refers to out of line or distorted on one side or blindness of data in left or right side. Kurtosis is the measure of the peakedness of a probability distribution; a normal distribution has 0 kurtosis [41].

<table>
<thead>
<tr>
<th>Table 4: Tests of Normality(Unit Level)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shapiro-Wilk</td>
</tr>
<tr>
<td>Statistic</td>
</tr>
<tr>
<td>Unit Level</td>
</tr>
</tbody>
</table>

At unit level the significant value of shapiro-wilk is 0.002 respectively which is less than most significant P-value (0.05). This means at unit test level Non-Parametric test Wilcoxon is needed for statistical test.

<table>
<thead>
<tr>
<th>Table 5: Tests of Normality(Integration Level)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shapiro-Wilk</td>
</tr>
<tr>
<td>Statistic</td>
</tr>
<tr>
<td>Integration level</td>
</tr>
</tbody>
</table>

At integration level the significant value of shapiro-wilk is 0.005 which means that shapiro-wilk test p-value is 0.005 and therefore parametric test is suitable. This means at integration test level parametric test two sample t-test is selected for statistical test.

<table>
<thead>
<tr>
<th>Table 6: Tests of Normality(System Level)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shapiro-Wilk</td>
</tr>
<tr>
<td>Statistic</td>
</tr>
<tr>
<td>System Level</td>
</tr>
</tbody>
</table>

At System level the significant value of shapiro-wilk is 0.012 which is less than most significant P-value (0.05). This means at system test level Non-Parametric test Wilcoxon is needed for statistical test.
4.3 Unit Level using Wilcoxon Test

- Wilcoxon test is used as an alternative to T-test [13], this test is used to determine which of measures in a pair is greater than the other with the help of rank differences.
- Wilcoxon test is conducted to evaluate whether defects at seeded level are greater than defects detected at unit test level for ET.
- Defects at unit level are analyzed by using Wilcoxon test and the results are shown below with their descriptive statistics and ranks.
- Finally with the help of hypotheses testing at unit level, acceptance or rejection of null hypotheses is checked.

<table>
<thead>
<tr>
<th>Table 13: Descriptive Statistics at unit level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Seeded Defects</td>
</tr>
<tr>
<td>After Experiment Defects</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 14: Test Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Z</td>
</tr>
<tr>
<td>Asymp. Sig. (2-tailed)</td>
</tr>
</tbody>
</table>
Table 15: Ranks

<table>
<thead>
<tr>
<th>N</th>
<th>Mean Rank</th>
<th>Sum of Ranks</th>
</tr>
</thead>
<tbody>
<tr>
<td>26a</td>
<td>14.46</td>
<td>376.00</td>
</tr>
<tr>
<td>1b</td>
<td>2.00</td>
<td>2.00</td>
</tr>
<tr>
<td>8c</td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

a. After Experiment Defects at Unit Level < Seeded Defects at Unit Level
b. After Experiment Defects at Unit Level > Seeded Defects at Unit Level

**Decision Rule: Asymp.(Sig)< 0.05**

**Decision Taken: Reject the Null Hypothesis(Ho1)**

**Null Hypothesis(Ho1):** ET detects all the defects seeded at Unit level

**Alternate Hypothesis(Hi1):** ET does not detect all the defects seeded at Unit level

Table 13, 14 & 15: Wilcoxon statistical test at Unit Level

The statistical results are analyzed in the following steps by using table 13, 14 & 15

**Step 1: Assumption of Hypothesis**

Null hypothesis states that “there is no difference in seeded defects vs. defects detected by subjects” [46]. Our null hypothesis assumption is stated below as

Null Hypothesis (H<sub>0</sub>1): ET detects all the defects seeded at Unit level.

Alternative Hypothesis states that “there is a difference seeded defects vs. defects detected by subjects” [46]. Our alternative hypothesis assumption is stated below as

Alternate Hypothesis (H<sub>i</sub>1): ET does not detect all the defects seeded at Unit level.

**Step 2: Significant value**

Significance level or α = 0.05 which is the probabilistic value or commonly used p-value.
Step 3: Test statistic

From the table 15, the mean of negative rank (14.46) is higher than mean of positive rank (2.00). From the table 15 'a' denotes that, defects detected by subjects are less than seeded defects. Asymp. Sig. (2-tailed) which is P-value or significant value is 0.000 from the table 14 which is exponentially a small value. Test statistic is statistically significant if $\alpha<0.05$.

Step 4: Conclusion

P-value $\sim 0.001 \sim (\alpha<0.05)$

The above notation shows that there is statistically significant difference between seeded defects and defects detected by subjects. We conclude from Wilcoxon test that ET does not detect all the seeded defects seeded at unit level.

4.3.1 Hypothesis testing at Unit test level

The results indicated from the statistics in figure 4 indicates $p<0.05$. The smaller the p-value, the more convincing is the rejection of null hypothesis ($H_0$) [43]. The significant value is .001 from the figure 4 shows that p-value is less than 0.05. Therefore null hypothesis is compared with alternate hypothesis and alternate hypothesis is accepted which means that ET does not detects all the defects seeded at unit level. Bar graph is used to visually display statistical results.

Table 7: Hypothesis Summary at Unit level

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Null or Alternate Hypothesis</th>
<th>Significant</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>ET does not detect all the defects seeded at unit level.</td>
<td>Alternate Hypothesis</td>
<td>0.001</td>
<td>Reject the Null Hypothesis</td>
</tr>
</tbody>
</table>
Figure 4: Bar graph visualizing Unit Level Defects

From the above figure, it is clearly showing that mean of seeded defects (20) at unit level is higher than mean of defects detected after performing (12.34) experiment. Therefore Alternative Hypothesis (H1) is accepted which states that, ET does not detect all the defects seeded at unit level and is visually shown with the help of bar graphs showing means in the y-axis and seeded defects, defects detected after experiment on x-axis.

4.4 Integration Level using Two Sample T-test

- Two sample t-test is the simple t-test which is used to compare sample populations and checks whether there is significant difference between their means or not. The result obtained from the test is ‘t’ value, when is then used to determine the significant value or p-value[49].
- T-test is also to compare the actual difference between two means in relation to the variation in the data [50].
- T-test takes two sets of data then calculates the means, standard deviations and standard errors. Based on ‘t’ value and p-value obtained through statistical tests, statistical significance between means or populations can be determined.
- Defects at integration level are analyzed by using two sample t-test and the results are tabulated as below with their descriptive statistics.
Finally with the help of hypothesis testing at integration level, acceptance or rejection of null hypothesis is checked.

<table>
<thead>
<tr>
<th>Table 16 : Group Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integration Test level</td>
</tr>
<tr>
<td>N</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>Descriptive statistics</td>
</tr>
<tr>
<td>seeded</td>
</tr>
<tr>
<td>after experiment</td>
</tr>
</tbody>
</table>

Table 17: Two Sample T-test

<table>
<thead>
<tr>
<th></th>
<th>Test for Equality of Variances</th>
<th>T-test for Equality of Means</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F</td>
<td>Sig.</td>
</tr>
<tr>
<td>Seeded and After experiment defects values</td>
<td>Equal variances assumed</td>
<td>91,314</td>
</tr>
<tr>
<td></td>
<td>Equal variances not assumed</td>
<td></td>
</tr>
</tbody>
</table>

Figure: Two Sample T-test at Integration Test level

**Decision Rule:** p-value (Sig.)< 0.05

**Decision Taken:** Reject The Null Hypothesis(Ho2)

**Null Hypothesis(Ho2):** ET detects all the defects seeded at Integration level

**Alternate Hypothesis(Hi2):** ET doesn't detect all the defects seeded at Integration level

Table 16, 17: Two Sample T-test at Integration test level

The table 16 indicates mean for seeded defects at integration level is 5.00; the mean for defects after experiment is 2.89. the standard deviation for seeded defects at integration level is 0.00, the standard deviation for defects after experiment is 1.430. The number of participants or subjects are 35.

The paired t-test statistics results are analyzed in the following steps by using tables 16, 17.
Step 1: Assumption of Hypothesis

Null hypothesis states that “there is no difference in seeded defects vs. defects detected by subjects” [46]. Our null hypothesis assumption is stated below as

Null Hypothesis ($H_0$): ET detects all the defects seeded at integration level.

Alternative Hypothesis states that “there is a difference seeded defects vs. defects detected by subjects” [46]. Our alternative hypothesis assumption is stated below as

Alternate Hypothesis ($H_1$): ET does not detect all the defects seeded at integration level.

Step 2: Significant value

Significance level or $\alpha = 0.05$ which is the probabilistic value or commonly used $p$-value.

Step 3: Test statistic

From the table 17 Sign-value indicates that one pair set of means are not equal. From the means matrix we conclude that there is statistical difference between seeded defects and defects found by experiment. $P$-value or significant value from the figure 5 is 0.001. Test statistic is statistically significant if $\alpha < 0.05$.

Step 4: Conclusion

$$P\text{-value} \sim 0.001 \sim (\alpha < 0.05)$$

The above notation shows that there is statistically significant difference between seeded defects and defects detected by subjects. Since that the mean number of seeded defects is (5.00) which is greater than mean number of defects detected by subjects which is (2.89) from figure 16, we conclude from two sample t-test that there is significant difference between the seeded defects and defects detected by subjects during experiment at integration level.

4.4.1 Hypothesis testing at Integration level

The results indicated significant difference is equal to 0.001, $p$-value $< 0.05$. The smaller the $p$-value, the more convincing is the rejection of null hypothesis ($H_0$) [44]. As seen in table. Descriptive statistics at integration level, the mean of seeded defects (5) at integration level is greater than defects after experiment (2.89). The significant value is .001 from the figure 5 shows that $p$-value is less than 0.05. Therefore null hypothesis ($H_0$) is compared with alternate hypothesis ($H_1$) and alternate
hypothesis is accepted which means that ET does not detect all the defects seeded at integration level. Bar graph is used to visually display statistical results.

Table 8 Hypothesis Summary at integration level

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Null or Alternate Hypothesis</th>
<th>Significant</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>ET does not detect all the defects seeded at Integration testing</td>
<td>Alternate Hypothesis</td>
<td>0.001</td>
<td>Reject the Null Hypothesis</td>
</tr>
</tbody>
</table>

Figure 5: Bar graph visualization of integration level defects

From the above figure 8, it is clearly showing that mean of seeded defects (5) at integration level is higher than mean of defects detected after performing (2.89) experiment. Therefore Alternative Hypothesis ($H_2$) is accepted which states that, ET does not detect all the defects seeded at integration level and is visually shown with the help of bar graphs showing mean in the y-axis and seeded defects, defects detected after experiment on x-axis.

4.5 System Level using Wilcoxon test

- Wilcoxon test is used as an alternative to T-test [13]; this test is used to determine which of measures in a pair is greater than the other with the help of rank differences.
Wilcoxon test is conducted to evaluate whether defects at seeded level are greater than defects detected at system test level for ET.

Defects at system level are analyzed by using Wilcoxon test and the results are shown below with their descriptive statistics and ranks.

Finally with the help of hypotheses testing at system level, acceptance or rejection of null hypotheses is checked.

### Table 18: Descriptive Statistics

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>Mean</th>
<th>Std. Deviation</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>SeededDefects</td>
<td>35</td>
<td>5.00</td>
<td>.000</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>AfterExperiment Defects</td>
<td>35</td>
<td>3.29</td>
<td>1.226</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

### Table 19: Test Statistics

<table>
<thead>
<tr>
<th></th>
<th>System test level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z</td>
<td>-4.757b</td>
</tr>
<tr>
<td>Asymp. Sig. (2-tailed)</td>
<td>.001</td>
</tr>
</tbody>
</table>

### Table 20: Ranks

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>Mean Rank</th>
<th>Sum of Ranks</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Test Level</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Negative Ranks</td>
<td>29a</td>
<td>15.00</td>
<td>435.00</td>
</tr>
<tr>
<td>Positive Ranks</td>
<td>0b</td>
<td>.00</td>
<td>.00</td>
</tr>
<tr>
<td>Ties</td>
<td>6c</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Decision Rule: **Asymp.(Sig)< 0.05**

**Decision Taken:** Reject the Null Hypothesis(Ho3)

Null Hypothesis(Ho3): ET detects all the defects seeded at System level

Alternate Hypothesis(Hi3): ET does not detect all the defects seeded at System level

Table 18, 19 & 20: Wilcoxon test at System Test Level
The wilcoxon statistics results are analyzed in the following steps by using table 18, 19 & 20.

**Step 1: Assumption of Hypothesis**

Null hypothesis states that “there is no difference in seeded defects vs. defects detected by subjects” [46]. Our null hypothesis assumption is stated below as

Null Hypothesis ($H_0$): ET detects all the defects seeded at system level

Alternative Hypothesis states that “there is a difference seeded defects vs. defects detected by subjects” [46]. Our alternative hypothesis assumption is stated below as

Alternate Hypothesis ($H_1$): ET does not detect all the defects seeded at system level

**Step 2: Significant value**

Significance level or $\alpha = 0.05$ which is the probabilistic value or commonly used P-value.

**Step 3: Test statistic**

From the table 20, the mean of negative rank (15) is higher than mean of positive rank (0.00). From the table 20 'a' denotes that, defects detected by subjects are less than seeded defects. Asymp. Sig. (2-tailed) which is P-value or significant value is 0.000 from the table 19 which is exponentially a small value. Test statistic is statistically significant if $\alpha<0.05$.

**Step 4: Conclusion**

P-value $\sim 0.001 \sim (\alpha<0.05)$

The above notation shows that there is statistically significant difference between seeded defects vs. defects detected by subjects. Since that the mean number of seeded defects is (5) which is greater than mean number of defects detected by subjects which is (3.29) from table 18, we conclude that from Wilcoxon test there is significant difference between the seeded defects vs. defects detected by subjects during experiment at system level.

4.5.1 Hypothesis testing at System test level

The results indicated from the statistics in table 20 indicates $p<0.05$. The smaller the $p$-value, the more convincing is the rejection of null hypothesis ($H_0$) [43]. The significant value is .001 from the figure 6 shows that p-value is less than
Therefore null hypothesis is compared with alternate hypothesis and alternate hypothesis is accepted which means that ET does not detect all the defects seeded at system level. Bar graph is used to visually display statistical results.

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Null or Alternate Hypothesis</th>
<th>Significant</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>ET does not detect all the defects seeded at system level.</td>
<td>Alternate Hypothesis</td>
<td>0.001</td>
<td>Reject the Null Hypothesis</td>
</tr>
</tbody>
</table>

Table 9: Hypothesis Summary at System level

![Visual Graph at System test level](image)

Figure 6: Bar graph visualization of system level defects

From the above figure 9, it is clearly showing that mean of seeded defects (5) at system level is higher than mean of defects detected after performing (3) experiment. Therefore Alternative Hypothesis (H₃) is accepted which states that, ET does not detect all the defects seeded at system level and is visually shown with the help of bar graphs showing means in the y-axis and seeded defects, defects detected after experiment on x-axis.
5. DISCUSSION

5.1 Observation

Our research focused on effectiveness of ET in finding seeded defects at the three levels of unit, integration and system testing. The results obtained from the experiment are mathematically studied by using different statistical tests like Wilcoxon and paired-t test and visually analyzed with the help of Bar graphs. Based on the results we conclude that ET is effective with minimum amount of documentation moreover testing process takes minimal time.

Experimentation of ET testing, at different test levels in web application didn’t yield better results. But the beneficial aspects of using ET are that subjects didn’t use any specific techniques or methods during testing. Subjects used their technical skills and creativity to find more defects. The results of the experiment revealed that effectiveness of finding defects at unit, integration and system software test levels are less than seeded defects in web application. The application of knowledge or technical skills has the main effect on the ability to find defects. Subjects having good testing and object oriented programming skills are able to find seeded defects easily.

Sometimes subjects needed help or advice during experimentation even though user manual is provided to them; one of the authors acted as an instructor and helped subjects while performing experiment. The defects in the web application was maintained confidentially and they were not discussed with subjects . Sometimes subjects got bored after performing experimentation, at this time break or snacks is provided to subject to give some relaxation and we believed that if the subjects lose interest on the testing it had effect on result, it is also seen with some subjects defects result. The time factor has also its effect on result of finding defects, some subjects needed more time than what we have given but we have restricted time to limited.

The reasons for the rejection of null hypothesis at three test levels is that subjects we have selected might be

- Inexperienced in testing web applications even though they have working experience on object oriented programming technologies
- Ability to explore defects
- Interest towards experimentation process.

We have used web application with a few web pages and database design, a similar experiment with more functionalities should be done in future with big web
applications like MVC frameworks, enterprise applications to be able to generalise results.

5.2 Validity Threats

Validity is the fundamental question which states that how far the experimental results are valid [13]. Validity threats are classified into four types as internal, external, construct and conclusion threats. These threats are mapped to different stages of experiment as described below.

5.2.1 Internal validity

An internal validity threat says that, a relationship exists between independent and dependent variables [13].

- Maturation: Maturation is that the performance of detecting defects. In our experiment, at the outset it is good but performance decreases once experiment goes on after 45 minutes at different test levels, the reason might be due to fatigue. In order to mitigate this threat break is provided for the subjects.

- Selection: Experienced vs. inexperienced subjects may have an impact on dependent variable. Some of the subjects who are having good experience on object oriented programming did well in identifying defects. However we cannot infer that inexperienced subjects are not good in testing, they also performed testing well with good results equivalent to experienced subjects. An effort was made so that experienced and inexperienced subjects were at same level by providing little training to inexperienced students.

- Mortality: Experimental mortality is that whether any subjects dropped out while performing experiment because this will have impact on the dependent variable. No subject dropped out of the experiment instead in our case few subjects got tired at different test levels.

5.2.2 External validity

External validity is considered as the degree to which the results are generalized with other group’s settings.

Wrong Selection of subjects: One of the threats is related to wrong selection of students as subjects for the experiment; reason is that in general software developers
generally perform unit testing and integration level but all students who are involved in verification and validation course need not have software development knowledge. To overcome this threat we ensured that we select right students as subject for the experiment. We made sure that we take students who have experience in software development as well as software testing.

Wrong place or environment: One of the threats associated with external validity is setting and environment. Complex applications take lot of time for the tester to understand its working in exploratory testing. To avoid this threat we have used simple application with a few web pages with seeded defects.

5.2.3 Construct validity

Construct validity is generalizing the results of the experiment to the concept or theory behind the experiment [13].

-Inadequate description of constructs: One of design threats related to our experiment is inadequate description of constructs used to measure effectiveness of ET. We have mitigated this threat by defining ET effectiveness as the number of defects found by the subjects compared to total number of defects at each level. We have also statistically analyzed by comparing this with the most significant value.

-Evaluation apprehension: One of the Social threats to our study is evaluation apprehension, which means fear of being evaluated. In order to mitigate this threat we have maintained anonymity of data received from the subjects. The results like defect recordings and data sets were neither changed nor revealed to anyone. Ethical values are considered sincerely during the experiment execution and afterwards.

5.2.4 Conclusion validity

Conclusion validity threats deal with the issues that affect the ability to draw correct conclusion from the treatment and the outcome [13].

-Low statistical power: Different statistical tests are used to test the significance value and evaluate the conclusion. However there may be chances of having low statistical power due to single statistical test, this can be seen as a limitation of this study.

-Random irrelevancies: There may be disturbances of noise from outside or some sudden interrupt while conducting experiment, to overcome these random irrelevancies in experimental setting subjects are invited to conduct experiment in lab environment and subjects are requested to switch off their mobiles during experimentation.
5.3 Evaluation of experimentation results

5.3.1 Mapping RQs to Obtained results

This section summarises and answers research question.

RQ: Can ET detect all the defects seeded at unit, integration and system levels of testing?

We have studied many exploratory testing practices available in the literature. From the analysis of study we framed our research questions successfully and tried to fill the research gap with the help of experiment. The experiment helped us to examine the effectiveness of ET in finding seeded defects. The results of research indicate that ET doesn’t detect all the seeded defects at each of the three levels. Alternate hypotheses at three levels were accepted in our experiment.
6. CONCLUSION

In this thesis, v-model and different software test levels more precisely unit, integration and system testing were studied using literature review and then we tried to check cause and effect of ET and its benefits into software test life cycle using experiment and tested ET’s effectiveness. Our main focus of experiment was effectiveness of ET in finding defects at three test levels. Effectiveness is measured by comparing the defects seeded with the defects detected after performing experiment. The results from the experiment revealed that ET didn’t achieve expected results at three test levels. We performed hypotheses testing with the help of statistical tests. The null hypothesis at three test levels is rejected, which draws a conclusion that there are some factors which influenced dependent variable. The cognitive factors that had influence on the dependent variable are cognitive sciences like technical experience, domain knowledge and interest of the subject towards the application which is a personal attribute. If these factors are controlled and implemented in unit, integration and system test levels then exploratory testing yields beneficial results. We conclude that exploratory testing needs more research in testing its effectiveness at different software test levels. One of the facts that we conclude with help of experiment is that, time spent for defect detection using ET is less and this reduces documentation related tasks during testing.

6.1 Future work

This section involves future work of ET in our understanding from experimental and literature studies. Following are the areas which should be concentrated in ET in the coming future.

- Our experiment results show the results of the limited environment, it may be more accurate if tested under real-time projects which give accurate results.
- Empirical investigation of ET in terms of cognitive sciences like technical, domain knowledge should be studied under different software test levels.
- Future studies should concentrate on how to cater cognitive factors that affect exploratory testing?
- ET results should be combined with defect severity.

One of the aims of the experimentation is replication [13]; this experiment should be replicated with more number of students and should be done with industry practitioners so that the results can be generalized.
7. REFERENCES


42. http://core.ecu.edu/psyc/wuenschk/Courses.htm, Date of visit 25.07.2012

43. http://www.stats.gla.ac.uk/steps/glossary/hypothesis_testing.html, Date of visit 26.07.2012

44. http://www.wellesley.edu/Psychology/Psych205/pairttest.html, Date of visit 26.07.2012


8. APPENDIX

8.1 Appendix A

8.1.1 Assessment form

Table 10: Pre Assessment form

<table>
<thead>
<tr>
<th>Pre Assessment Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of the subject</td>
</tr>
<tr>
<td>Previous testing experience?</td>
</tr>
<tr>
<td>Have you taken the course verification and validation?</td>
</tr>
<tr>
<td>Department</td>
</tr>
<tr>
<td>Do you have knowledge of test levels?</td>
</tr>
<tr>
<td>Do you know about ET</td>
</tr>
<tr>
<td>Do you have knowledge of .NET</td>
</tr>
</tbody>
</table>

8.1.2 Defect Count Report

Table 11: Defect count report

<table>
<thead>
<tr>
<th>Name of the tester</th>
<th>Date:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defect No</td>
<td>Defect description</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# 8.1.3 Defects Seeded Report

## Seeded Defects Report

<table>
<thead>
<tr>
<th>Defect No</th>
<th>Test Level</th>
<th>Summary</th>
<th>File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unit</td>
<td>The mobile test field is not taking user input, it is seeded as defect</td>
<td>registration.aspx.cs</td>
</tr>
<tr>
<td>2</td>
<td>Unit</td>
<td>The country text field is misplaced with state field data which is seeded as defect</td>
<td>registration.aspx.cs</td>
</tr>
<tr>
<td>3</td>
<td>Unit</td>
<td>regular expression for e-mail id is seeded as defect which is not taking email id in proper format</td>
<td>registration.aspx</td>
</tr>
<tr>
<td>4</td>
<td>Unit</td>
<td>button used for forgot password is visible but it not functional due to seeded defect</td>
<td>customerlogin.aspx</td>
</tr>
<tr>
<td>5</td>
<td>Unit</td>
<td>textbox is seeded with defect that it will take username that doesn’t exists</td>
<td>forgetpwd.aspx</td>
</tr>
<tr>
<td>6</td>
<td>Unit</td>
<td>user can navigate back to homepage and session is seeded as not to work even though code for session is there</td>
<td>creditcard.aspx.cs</td>
</tr>
<tr>
<td>7</td>
<td>Unit</td>
<td>textbox1 for taking input username is seeded as defect</td>
<td>forgetpwd.aspx.cs</td>
</tr>
<tr>
<td>8</td>
<td>Unit</td>
<td>password that is retrieved is seeded with another password which is incorrect</td>
<td>forgetpwd.aspx</td>
</tr>
<tr>
<td>9</td>
<td>Unit</td>
<td>defect is seeded as when the user clicks electronic items the webpage is redirecting to diamonds webpage</td>
<td>myshoppingcart.aspx</td>
</tr>
<tr>
<td>Unit</td>
<td>Description</td>
<td>Page/Link</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>button used for clicking ok for purchasing items is set as not to perform</td>
<td>tsharts.aspx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>performing calculations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>reset button is seeded as defect which is not resetting certain fields like</td>
<td>creditcard.aspx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>custormername, pid and present address on the webpage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>required field validator is seeded with defect that code for enable view state</td>
<td>creditcard.aspx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>is not functional</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Button used for going back is seeded as defect</td>
<td>creditcard.aspx</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>panel id is seeded with defect which makes the information on the webpage</td>
<td>aboutus.aspx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>not functional</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>code file is seeded with defect which makes it redirect to inappropriate</td>
<td>WebuserControl.ascx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>page</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>the label is seeded with defect which is used to show error message during</td>
<td>adminandsupplierlogin.aspx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>login is displayed without clicking login button</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>invoice page is not shown due to seeded defect at sqldatasource</td>
<td>supplier.aspx</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>gridview is seeded with defect which shows customer name instead of serial</td>
<td>supplier.aspx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>no</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>insert button in gridview is seeded with defect, there is no confirmation</td>
<td>supplier.aspx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>message after recordings are getting inserted</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>reset button is seeded as defect which is not functional</td>
<td>compose.aspx</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>defect is seeded as that ,inconsistent data is flowing from registration</td>
<td>Registration.aspx,admin.aspx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>page to admin page</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Admin unable to add supplier details and edit functionality, delete functionality is seeded with defect

user purchased products cannot be calculated in total due to inappropriate data from myshopping cart page to creditcard page

user cannot see continueshopping page due to defect that was made in creditcard module

admin and supplier unable to exchange email and same email page is opening to admin as well as supplier

User unable to purchase products from purchase management system

Admin and supplier should exchange information when needed

Supplier unable to send invoice related information to admin

Credit card information is not relevant and the fields are taking inconsistent data in the credit card textboxes

Administrator is unable to view consistent information about customer

Table 22: Seeded Defects Source code

In the below table, source code for the above defects along with their number are specified.
protected void Page_Load(object sender, EventArgs e)
{
    con = new SqlConnection("Integrated
Security=True;database=pms;server=madhu");
    da = new SqlDataAdapter("select * from register where
username=TextBox1.text", con);
    SqlCommandBuilder cmd = new SqlCommandBuilder(da);
    ds = new DataSet();
da.Fill(ds, "register");
dt = ds.Tables[0];
Label4.Visible = false;
Label5.Visible = false;
}
protected void Page_Load(object sender, EventArgs e)
{
    txtproductname.Text = Session["d"].ToString();
    txtpurchaseamount.Text = Session["f"].ToString();
    TextBox1.Text = Session["e"].ToString();
}
da = new SqlDataAdapter("select * from register where username='
TextBox1.text'");

protected void Button1_Click(object sender, EventArgs e)
{
    DataRow row;
    string n;
    n = TextBox1.Text;
    row = dt.Rows.Find(n);

    if (row == null)
    {
        Response.Write("username not found");
    }
    else
    {
        Label4.Text = row[2].ToString();
        Label4.Visible = true;
        Label5.Visible = true;
    }

&lt;asp:HyperLink ID="HyperLink4" runat="server" Font-Bold="True" Font-Size="X-Large"
Height="14px" NavigateUrl="~/Dimand.aspx" Style="z-index: 100; left: 23px;
position: absolute; top: 182px" Width="170px">Electronic Items&lt;/asp:HyperLink&gt;
protected void Button1_Click(object sender, EventArgs e)
{
    int a = Convert.ToInt32(TextBox2.Text);
    int b = Convert.ToInt32(TextBox3.Text);
    int c = a * b;
    TextBox4.Text = c.ToString();
}

protected void Button2_Click(object sender, EventArgs e)
{
    ddlcardtype.Text = "";
    txtcardno.Text = "";
    ddledate.SelectedItem.Text = "";
    ddlexpirymonth.SelectedItem.Text = "";
    ddlexpiryyear.SelectedItem.Text = "";
    txtnameoncard.Text = "";
    txtpurchaseamount.Text = "";
    txtproductname.Text = "";
}</asp:Button ID="Button2" runat="server" OnClick="Button2_Click" Style="z-index: 100; left: 282px; position: absolute; top: 354px" Text="Reset" Width="70px" BackColor="Gray" BorderColor="Red" ForeColor="White" />

<asp:Button ID="Button3" runat="server" BackColor="Gray" BorderColor="Red" ForeColor="White"
   PostBackUrl="/myshoppingcart.aspx" Style="z-index: 100; left: 513px; position: absolute;
   top: 353px; height: 26px;" Text="Back" Width="62px"
   onclick="Button3_Click" />

<uc1:WebUserControl ID="WebUserControl1" runat="server" EnableViewState="False" Visible="True" />
<asp:Button ID="Button2" runat="server" BackColor="Gray" BorderColor="#FF8080"
    OnClick="Button2_Click" Style="z-index: 102; left: 24px; position: absolute;
    top: 246px" Text="Reset" ForeColor="White" />
protected void Button2_Click(object sender, EventArgs e)
{
    txtfrom.Text = "";
    txtto.Text = "";
    txtsub.Text = "";
    txtbody.Text = "";
}

protected void Button1_Click(object sender, EventArgs e)
{
    if ((ddlgender.SelectedItem.Text != "---Select---") && (ddlstate.SelectedItem.Text != "---Select---") && (ddlcountry.SelectedItem.Text != "---Select---"))
    {
        DataSet ds = new DataSet();
da.Fill(ds, "register");
Response.Redirect("Home.aspx");
con.Close();
}
</asp:SqlDataSource>

<asp:SqlDataSource ID="SqlDataSource1" runat="server" ConnectionString="<%$ ConnectionStrings:pmsdbConnectionString7 %>">
    SelectCommand="SELECT * FROM [register]"
</asp:SqlDataSource>
protected void GridView1_RowEditing(object sender, GridViewEditEventArgs e) {
    GridView1.EditIndex = e.NewEditIndex;
    showdata();
}

protected void GridView1_RowCancelingEdit(object sender, GridViewCancelEditEventArgs e) {
    GridView1.EditIndex = -1;
    showdata();
}

protected void GridView1_RowCommand(object sender, GridViewCommandEventArgs e) {
    try {

        if (e.CommandName == "insert") {
            TextBox t1, t2, t3, t4, t5;
            // t = (TextBox)GridView1.FooterRow.FindControl("TextBox1");
            t1 = (TextBox)GridView1.FooterRow.FindControl("TextBox2");
            t2 = (TextBox)GridView1.FooterRow.FindControl("TextBox4");
            t3 = (TextBox)GridView1.FooterRow.FindControl("TextBox6");
            t4 = (TextBox)GridView1.FooterRow.FindControl("TextBox8");
            t5 = (TextBox)GridView1.FooterRow.FindControl("TextBox10");
            con.Open();
            string str = "insert into addanddelete supplier values(" + t1.Text + "," + t2.Text + "," + t3.Text + "," + t4.Text + "," + t5.Text + ");";
            SqlCommand cmd = new SqlCommand(str, con);
            cmd.ExecuteNonQuery();
            GridView1.EditIndex = -1;
            showdata();
            con.Close();
        }
    }

    catch (Exception ex) {
        Response.Write(ex.Message);
    }
}

protected void GridView1_RowDeleting(object sender, GridViewDeleteEventArgs e) {
    Label t = (Label)GridView1.Rows[e.RowIndex].FindControl("label1");
    con.Open();
    SqlCommand cmd = new SqlCommand("delete from addanddelete supplier where Sno =" + t.Text, con);
protected void GridView1_RowUpdating(object sender, GridViewUpdateEventArgs e) {
    TextBox t1, t2, t3, t4, t5;
    Label t = (Label)GridView1.Rows[e.RowIndex].FindControl("label1");
    t1 = (TextBox)GridView1.Rows[e.RowIndex].FindControl("TextBox1");
    t2 = (TextBox)GridView1.Rows[e.RowIndex].FindControl("TextBox3");
    t3 = (TextBox)GridView1.Rows[e.RowIndex].FindControl("TextBox5");
    t4 = (TextBox)GridView1.Rows[e.RowIndex].FindControl("TextBox7");
    t5 = (TextBox)GridView1.Rows[e.RowIndex].FindControl("TextBox9");

    con.Open();
    string str = "update addanddeletesupplier set supplierid =" + t1.Text + ", 
                Location=" + t2.Text + ", Address=" + t3.Text + ", Phoneno=" + t4.Text + ", EmailID =" + t5.Text + " where Sno =" + t.Text;
    SqlCommand cmd = new SqlCommand(str, con);
    cmd.ExecuteNonQuery();
    GridView1.EditIndex = -1;
    showdata();
    con.Close();
}

void showdata() {

    SqlDataAdapter da = new SqlDataAdapter("select * from addanddeletesupplier ", con);
    DataSet ds = new DataSet();
    da.Fill(ds, "addanddeletesupplier ");
    GridView1.DataSource = ds.Tables[0];
    GridView1.DataBind();
    //con.Close();
}
protected void Button1_Click(object sender, EventArgs e)
{
    int a = Convert.ToInt32(TextBox2.Text);
    int b = Convert.ToInt32(TextBox3.Text);
    int c = a * b;
    TextBox4.Text = c.ToString();
}

protected void Button10_Click(object sender, EventArgs e)
{
    Session["d"] = TextBox1.Text;
    Session["e"] = TextBox4.Text;
    Session["f"] = TextBox5.Text;
    Response.Redirect("creditcard.aspx");
}

protected void Button1_Click(object sender, EventArgs e)
{
    con.Open();
    if ((ddlcardtype.SelectedItem.Text != "---Select---") &&
    (ddledate.SelectedItem.Text != "---Select---") &&
    (ddlexpirymonth.SelectedItem.Text != "---Select---") &&
    (ddlexpiryyear.SelectedItem.Text != "---Select---"))
    {
        SqlDataAdapter da = new SqlDataAdapter("insert into creditcarddetails
values(" + txtcustomername.Text + "," + TextBox2.Text + "," +
ddlcardtype.SelectedItem.Text + "," + txtproductname.Text + "," + TextBox1.Text + "," +
txtcardno.Text + "," + ddlcardtype.SelectedItem.Text + "," +
ddlexpirymonth.SelectedItem.Text + "," + ddlexpiryyear.SelectedItem.Text + "," +
txtnameoncard.Text + "," + txtpurchaseamount.Text + ")", con);
        DataSet ds = new DataSet();
        da.Fill(ds, "creditcarddetails");
        Response.Redirect("continueshoping.aspx");
    }
    else
    {
        Response.Write(Label1.Text = "Invalid Select");
    }
    ddlcardtype.Text = ";
    txtcardno.Text = ";
    ddledate.SelectedItem.Text = ";
    ddlexpirymonth.SelectedItem.Text = ";
    ddlexpiryyear.SelectedItem.Text = ";
    txtnameoncard.Text = ";
    txtpurchaseamount.Text = ";
    txtproductname.Text = ";
    con.Close();
}
</form>
<form id="form1" runat="server"}
<asp:Label ID="Label1" runat="server" Font-Bold="True" Font-Size="X-Large" ForeColor="Fuchsia"
  Height="22px" Style="z-index: 100; left: 30px; position: absolute; top: 175px"
  Text="You have successfully buy the products If you want to continue plz Click here"></asp:Label>

&nbsp;
<asp:HyperLink ID="HyperLink1" runat="server" Font-Bold="True"
  ForeColor="Blue" NavigateUrl="~/myshoppingcart.aspx"
  Style="z-index: 101; left: 819px; position: absolute; top: 177px">Myshoppingcart</asp:HyperLink>
<asp:HyperLink ID="HyperLink2" runat="server" Font-Bold="True"
  ForeColor="Blue" NavigateUrl="~/home.aspx"
  Style="z-index: 103; left: 825px; position: absolute; top: 207px"
  Width="80px">Logout</asp:HyperLink>
</form>
protected void Button1_Click(object sender, EventArgs e)
{
    try
    {
        MailMessage me=new MailMessage();
        me.From=txtfrom.Text;
        me.To=txtto.Text;
        me.Subject=txtsub.Text;
    }
    catch (Exception ex)
    {
        lblmsg.Text = "Error:" + ex.Message;
    }

    SqlCommand cmd=new SqlCommand("insert into messages values('"+txtfrom.Text +","+ txtto.Text +","+ txtsub.Text +","+ txtbody.Text +"')",con);
    con.Open();
    cmd.ExecuteNonQuery();
    con.Close();
    Response.Write("");
    Label1.Text="Message has been send Successfully");
    Label1.Visible = true;
    //lblmsg.Visible = true;
    txtfrom.Text = "";
    txtto.Text = "";
    txtsub.Text = "";
    txtbody.Text = "";
    }
}
8.2 Appendix B

8.2.1 User Guide for Experiment

User guide for Purchase Management System is described under three software test levels. User guide provides the subjects with introduction to different software test levels and expected defects that may occur during testing.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Unit</th>
<th>Integration</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>People Involved</td>
<td>Programmers</td>
<td>Programmers &amp; Testers</td>
<td>Testers</td>
</tr>
<tr>
<td>Hardware &amp; OS</td>
<td>Programmers IDE</td>
<td>Programmers IDE</td>
<td>System Test</td>
</tr>
<tr>
<td></td>
<td>IDE (Integrated Development</td>
<td>IDE (Integrated Development</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Environment)</td>
<td>Environment)</td>
<td></td>
</tr>
<tr>
<td>Source of Test Data</td>
<td>Manually created</td>
<td>Manually created</td>
<td>Production &amp;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Manually created</td>
</tr>
<tr>
<td>Volume of Test Data</td>
<td>Small</td>
<td>Small</td>
<td>Large</td>
</tr>
<tr>
<td>Strategy</td>
<td>Unit</td>
<td>Group of Units</td>
<td>Entire System</td>
</tr>
</tbody>
</table>

Table 12: A brief representation of different software test levels involved in our experiment are tabulated and differentiated based on their attributes

Unit testing

Unit testing is the smallest testable piece of software [1].

Unit testing is defined as the lowest level of testing [2].

Examples of unit testing are classes, functions, interfaces etc. Unit testing concentrates on individual units of source code in software. The following are the expected Unit testing defects or this kind of defects are tested during unit testing.

- Incorrect data type
- Testing function procedures or event handling
- Variable declared but never used
- Mismatching of data like for integer type floating type data is given
- For loop(index issues)
- While loop(end condition might be wrong)
- In web applications unit testing follows like this

1. The data entered from front-end webpage should be stored in the back-end database but that functionality is not done.

2. The webpage or the server controls will be working but the functionality of the code is not working.

**Integration testing**

Testing correct interaction between System units [1].

Integration testing is intended to access whether the interfaces between modules are communicating properly. Integration testing is the responsibility of the members of the development team [2].

Integration testing is a systematic technique which aims at constructing the program structure besides testing the interfaces [3].

Integration testing mainly aims to test module interfaces to ensure that there are no errors during parameter passing when one module invokes another module. During integration testing, different units are integrated to test the interfaces among units.

The following are the expected defects that may arise during Integration testing.
- Correct call from each other functions
- Transfer right data
- Access interfaces at right time
- Check correct state of the component
- In web applications Integration Testing issues arise when user enters fields from front end, then the other module like admin or supplier in our application cannot see the user entered data or admin may view inconsistent data which is not required.
- When a web application is called it automatically redirects to other web application.
- Find dataflow between functions
- Integration issues with database when database connection is successfully connected.
System testing

In this, subsystems are integrated to make up the whole system [3]. In this type of testing, tester validates the web application with the functional and non-functional requirements of the system. Disparities between implementation and specification are found out [1].

System testing aims at 1. Test complete system & 2. Other performance testing

The following are the expected system defects that may arise during system testing.

- Spelling errors in pages
- Incorrect/absence of data
- Actual functionality of code is missing.
- Logic or calculation of data is incorrect or inconsistent.
- When the customer purchases a product from web application but he is unable to purchase or he is redirected to other web pages.

8.2.2 User guide References:


8.3 Appendix C

Figure 10: Screen shot of web application used in the experiment
Figure 11: Screen shot of web application used in the experiment
Figure 12: Screen shot of web application used in the experiment
Figure 13: Screen shot of web application used in the experiment