Pre-print article accepted for publication in Empirical Software Engineering Journal, June 2013.

DOI: 10.1007/s10664-013-9266-8
The final publication is available at link.springer.com
http://dx.doi.org/10.1007/s10664-013-9266-8

Are Test Cases Needed? Replicated Comparison between Exploratory and Test-Case-Based Software Testing

Juha Itkonen

Aalto University, Department of Computer Science and Engineering
juha.itkonen@aalto.fi

Mika V. Mäntylä

Aalto University, Department of Computer Science and Engineering / Lund University, Department of Computer Science
mika.mantyla@aalto.fi

Abstract

Manual software testing is a widely practiced verification and validation method that is unlikely to fade away despite the advances in test automation. In the domain of manual testing, many practitioners advocate exploratory testing (ET), i.e., creative, experience-based testing without predesigned test cases, and they claim that it is more efficient than testing with detailed test cases. This paper reports a replicated experiment comparing effectiveness, efficiency, and perceived differences between ET and test-case-based testing (TCT) using 51 students as subjects, who performed manual functional testing on the jEdit text editor. Our results confirm the findings of the original study: 1) there is no difference in the defect detection effectiveness between ET and TCT, 2) ET is more efficient by requiring less design effort, and 3) TCT produces more false-positive defect reports than ET. Based on the small differences in the experimental design, we also put forward a hypothesis that the effectiveness of the TCT approach would suffer more than ET from time pressure. We also found that both approaches had distinctive issues: in TCT, the problems were related to correct abstraction levels of test cases, and the problems in ET were related to test design and logging of the test execution and results. Finally, we recognize that TCT has other benefits over ET in managing and controlling testing in large organizations.

Keywords: Software testing, manual testing, test cases, exploratory testing, experiment, effectiveness, efficiency.

1 Introduction

Manual software testing is an important and a widely practiced verification and validation (V&V) method despite the advances of automated software testing. Evidence from software practitioners (Berner et al. 2005; Martin et al. 2007; Rafi et al. 2012) suggest that automated testing will never completely replace manual testing, and in a recent survey of software practitioners (Rafi et al. 2012), only 6% of the 115 respondents agreed that all manual testing will be eventually superseded by software test automation. Yet studies of manual software testing are scarce.

Software testing is commonly seen as a process of executing test cases that are carefully predesigned using test-case-design techniques (Beizer 1990; Kaner et al. 1999; Myers 1979). In this test-case-based testing approach (TCT), the goal is to document the required knowledge in the test cases. The actual test execution, even if performed as a manual activity, is considered a mechanical task. During execution, the predefined test cases are run and their output compared with the documented expected results. However, studies on industrial practice report that real-world testing is seldom based on rigorous, systematic, and thoroughly documented test cases (Andersson and Runeson 2002; Engström and Runeson 2010; Ng et al. 2004).

The exploratory software testing (ET) is recognized in the software engineering body of knowledge (Abran et al. 2004) as a manual testing approach that is not based on predesigned test cases. Instead, it is a creative, experience-based approach in which test design, execution, and learning are parallel activities, and the results of executed tests are immediately applied to designing further tests (Bach 2004). Practitioners find that exploratory aspects are fundamental to most manual testing activities (Itkonen and Rautiainen 2005; Kaner et al. 2002; Tinkham and Kaner 2003a; Whittaker 2009). ET is a practically relevant testing approach that has been claimed, in practitioners' reports, to be both effective in detecting failures and cost efficient (Bach 2004; Våga and Amland 2002; Wood and James 2003; Lyndsay and van Eeden 2003; Bach 2000). At the same time, scientific research on the ET approach is scarce and only a few empirical studies of ET or similar approaches exist, see, e.g., (Houdek et al. 2002; do Nascimento and Machado 2007; Pichler and Ramler 2008; Itkonen et al. 2007; Itkonen et al. 2009; Itkonen and Rautiainen 2005).

In this paper, one of the original studies of ET (Itkonen et al. 2007) is replicated. The results of the original student experiment were interesting and indicated that the effectiveness of ET could be as high as with the TCT approach, but with significantly less effort. In the original experiment, the test execution time was strictly restricted to two-hour controlled sessions, thereby limiting the generalizability of the findings to the context where a longer testing time would be available. In this replication, we have removed the time restriction to address this limitation. The high-level research question in this experiment is the same as in the original experiment: *What is the effect of using predesigned and documented test cases in the manual functional testing with respect to defect detection performance?* The detailed research questions are presented in Section 3.1.

In the next section, we review relevant related literature and present the design and results of the original experiment in detail. In the third section, we state the exact research questions and describe the research methods with the changes to the original experimental design. The results are described in Section 4 together with a discussion and comparison with the original experiment. Finally, in the last section, we present the conclusions and directions for future research.

2 Related Work

The motivation for this experiment has been the claims and reports related to the effectiveness and efficiency of the ET approach. In Section 2.1, we cover the ET approach and review the practitioner literature. We elaborate specifically on the existing scientific results on the effectiveness and efficiency of the ET approach in Section 2.2 and the TCT approach in section 2.3. In Section 2.4, we describe the original study for this replication by following the replication reporting guidelines by Carver (2010).

2.1 Exploratory Software Testing

The ET approach has been proposed in practitioner literature and identified in the software testing community during the last decade. Despite the somewhat persistent role of ET in the software development and testing practitioner community, scientific research on ET is still scarce. In this literature review, we first introduce the ET approach based on books and the practitioner literature. Second, we cover practitioner reports on experiences and claimed benefits.

The ET approach has been acknowledged in software testing books since the 1970s (Myers 1979), but it has been referred to mostly as an ad hoc approach or error guessing without any concrete description of how to perform such testing. The term ET has been used in software testing books since the late 1990s (Kaner et al. 1999). The ET approach has been covered in books by Kaner et al. (2002) and, more recently, by Whittaker (2009). Many other books on testing also cover the exploratory aspects to some extent, see, e.g., (Copeland 2004; Craig and Jaskiel 2002; Crispin and Gregory 2009; Page et al. 2008; Whittaker 2003). ET is defined in SWEBOK as: "*simultaneous learning, test design, and test execution; that is, the tests are not defined in advance in an established test plan, but are dynamically designed, executed, and modified*" (Abran et al. 2004). Popular sources for writings on ET are the web articles and blog writings of a few ET advocates (Bach 2003; Bolton 2005). Jonathan Bach (2000) published the first approach to managing ET, called "session-based test management." In addition, James Bach (1999) described an ET procedure for testing the functionality and stability of a software application for the "Certified for Microsoft Windows Logo." ET is often described as how experienced testers work. For example, Lyndsay and van Eeden (2003) wrote, "Session-based testing mirrors the activities of experienced testers, but is not the subject of a great many papers or books."

Practitioner reports on experiences of applying ET approaches in industry have claimed that ET is effective in detecting defects and is cost-efficient. Bach described his experiences of the efficiency of ET (Bach 2004; Bach 2003). Lyndsay and van Eeden (2003) reported many results and lessons learned from their exploratory session-based testing (ESBT) approach. The most important results were the ability to measure and control the ET process and the visibility of the testing work to the test manager. The testing team also felt in control of their work. They could see the size of their work, its status, and its progress. This improved visibility of the testing process improved the trust between the developers and the testing team. Finally, they reported that the tested product in this case was more stable and had fewer outstanding defects. Wood and James (2003) reported the benefits of applying ESBT in the medical software domain. They identified several problems compromising the effective testing of medical software, including highly

compartmentalized testing, excess “housekeeping” documentation at the expense of actual testing, an emphasized focus on requirements and code coverage instead of defect discovery, and repetitive testing. As a solution to these problems, Wood and James proposed complementing the standard testing methods with ESBT. They found ESBT to be an effective testing approach, especially if the testers were independent of the development and V&V activities. They suggested that ESBT is the most applicable in the areas where heavy user interaction exists and outcomes can be confirmed quickly.

Våga and Amland (2002) reported a case description involving the application of an ET approach in a very tightly scheduled acceptance testing project of a proprietary Web publishing system for a large multinational IT company. The documentation for the system was scarce, and the developers still changed and fixed the system frequently, even though it was going into production. They combined ET and pair testing in their two-day testing project. The reported project was an extreme case where the task was to test a completed Web system in two days using inexperienced testers, mainly end users, who had only two days of training in ET prior to the testing work. They had 14 testers who, during this two-day acceptance testing, reported approximately 150 defects. In this case, the testing effort prevented the release of an immature system with too many defects into production. It took three months to fix the 65 most critical of the found defects before eventually taking the system into production with limited functionality.

2.2 Scientific Research on Exploratory Testing

In the scientific research literature, only a few studies related to ET have been published. There is a body of empirical research on the effectiveness of numerous test case design techniques (Juristo et al. 2004), but no studies have compared the ET and test case-based approaches. In a related study, Houdek et al. (2002) studied defect detection effectiveness in an executable specification context by comparing systematic testing and experience-based ad hoc simulation. According to their results, ad hoc simulation required less effort than the systematic techniques of inspection and testing, and there was no difference in effectiveness (Houdek et al. 2002). This gives some support to the hypothesis that ET could be an efficient approach to detecting defects. However, it is hard to generalize their findings from the executable specification context to functional software testing.

In an industrial case study, Itkonen and Rautiainen (2005) summarized the definitions and characteristics of ET based on literature review and reported results on the perceived benefits and shortcomings of ET in three software product companies. The main motivations to apply the ET approach were related to the challenges of applying conventional TCT methods, the ability to test from the users’ viewpoint, and utilize testers’ experience and creativity. The most important perceived benefits were effectiveness and efficiency of the ET approach, its versatility, as well as its ability to provide better view of the overall quality. The rapid feedback from testing to development was also reported as a benefit. The most important shortcomings of the ET approach, according to Itkonen and Rautiainen (2005), were the management of coverage, tracking the testing activities, high dependency on the individual properties of the testers, and repeatability of the tests in some cases.

Do Nascimento and Machado (2007) compared ET and model-based feature testing in the mobile phone application domain. They found the ET approach to be as effective as, and even more efficient than, the other testing approaches. The results were, thus, similar to those previously reported by Houdek et al.

Other studied aspects of ET include the effect of the individual characteristics on ET. Tinkham and Kaner (2003b) used differences in learning styles to explain different styles of testing and, more recently, Shoaib et al. (2009) studied the effect of personality traits on ET performance and found that extroverted personality types might be more likely to be good at ET. Tuomikoski and Tervonen (2009) reported good experiences in integrating ET sessions with the agile Scrum methodology. They reported good results in terms of the number of revealed defects as well as the benefits of sharing knowledge and forming a common understanding about the actual quality level through the ET sessions. These few studies on ET give some support to the hypothesis that ET could be an effective and efficient testing approach in certain contexts.

As a summary of the review of related ET work, it can be stated that practitioners have promoted ET and positive experiences with the applicability and effectiveness of ET have been presented in the practitioner literature. Research on ET in scientific forums is emerging. Even though there are experimental research studies on testing techniques, the effects of the TCT in comparison to the ET have not been studied experimentally prior to the original experiment by Itkonen et al. (2007). These results support the hypothesis of the similar effectiveness and higher efficiency of the ET approach in comparison to TCT approaches. In addition, as the existing results on the effects of the test-case-design techniques are inconclusive, the questions about the effect of predesigned test cases are highly relevant.

2.3 Defect Detection Effectiveness of TCT Techniques

The effectiveness of test case design techniques has been compared in many empirical studies. The results on the effectiveness of the testing techniques for functional testing have been presented by, e.g., (Basili and Selby 1987; Kamsties and Lott 1995; Myers 1978; Wood et al. 1997). The empirical research on testing techniques was reviewed by Juristo et al. (2004). They presented the following recommendations regarding functional testing techniques based on the research. First, boundary value analysis was recommended over the sentence coverage technique if the testers are experienced and have enough time. On the other hand, for inexperienced subjects and time pressure, it would be better to use sentence coverage. It is preferable to use boundary value analysis as opposed to condition coverage, as there is no difference regarding the effectiveness and it takes less time to detect and isolate faults.

Juristo et al. also state that the subject (individual) affects technique application time, fault detection, and fault isolation. In addition, the program affects the number and type of faults detected and the testing time. They also point out that more faults are detected by combining subjects than by combining techniques. These results indicate that in testing activities other factors than the test design technique have significant effects on the results. These include the factors of who performs the testing and how the testing is performed and how the test design techniques are applied.

Runeson et al. (2006) have compared the effectiveness of testing and inspection techniques. Their conclusion was similar to the results of Juristo et al. stating that there are no clear answers to the question of which method to choose, but the effectiveness depends on the tested artifact and the types of the defects.

In a more recent study, Juristo et al. (2012) present a study of eight replicated student experiments comparing equivalence partitioning, branch testing, and code reading by stepwise abstraction. Their results indicate that there is no difference in effectiveness between the two studied dynamic testing techniques (equivalence partitioning and branch testing) and this is not dependent on the program. However, they found that fault detection effectiveness in general is program dependent, but independent of the fault type. They also conclude that these techniques are more effective when different individuals apply the same techniques in comparison to the same individual applying different techniques.

The previous research shows that there are many factors that affect the defect detection effectiveness of software testing methods. These factors, including individual differences and the effects of human factors, such as motivation and satisfaction, should be studied (Runeson et al. 2006). This research aims at investigating how the testing approach, i.e., how the testing is performed and test designs applied, affects the defect detection effectiveness and efficiency.

2.4 The Original Study

The original study of this replication was a controlled student experiment published by Itkonen et al. in a conference paper that includes the main results of the study (Itkonen et al. 2007). A more thorough description is provided in a thesis (Itkonen 2008)[1] where the original survey data is also presented. The research is a controlled experiment comparing the defect detection effectiveness of ET and TCT. In the experiment, 79 advanced software engineering students performed manual functional testing on an open-source text editor application[2] with authentic and seeded defects. Each student participated in two controlled sessions using ET in one and TCT in the other. This section describes the original study (Itkonen et al. 2007; Itkonen 2008) following the guidelines proposed by Carver[3] (Carver 2010). In particular, this section describes the original study from six viewpoints: research questions, subjects, design, artifact, context variables, and results.

2.4.1 Original Research Questions

The research problem in the original study was stated as: What is the effect of using predesigned and documented test cases in manual functional testing with respect to defect detection performance? The research questions of the original study were all interested in the effect that predesigned test cases have in manual software testing. This condition was compared against an alternative where no predesigned test cases were available and where the testers performed free-form ET. The research questions of the study were presented as follows:

Research question 1: How does using predesigned test cases affect the number of detected defects?

---

[1] Available at http://www.soberit.hut.fi/jitkonen/Publications/Juha_Itkonen_Licentiate_Thesis_2008.pdf
[2] jEdit, http://www.jedit.org/
[3] http://www.cs.ua.edu/~carver/ReplicationGuidelines.htm

Research question 2: How does using predesigned test cases affect the type of defects found?

Research question 3: How does using predesigned test cases affect the number of false defect reports[4]?

Research question 4: How does using predesigned test cases affect the perceived test coverage, quality, and problems of testing? (Itkonen 2008)

2.4.2 Original Subjects

The subjects of the original study were 79 master's students of the Software Testing and Quality Assurance course at the Helsinki University of Technology. The students had completed an average of 113[5] credits out of 160 required for the degree. Over half of the students had more than a year of software development experience from the industry, but software-testing experience was limited to only 21% of the subjects. For a more detailed description of the demographics, see Table 4 that presents the comparison between original and replication study subjects.

2.4.3 Original Experimental Design

The original study had a one-factor block design with a single blocking variable. An overview of the experimental arrangements is shown in **Table 1**. The subjects were randomly divided into two groups, both of which performed similar test sessions with and without test cases in a randomized order. The experiment consisted of three phases: preparation, including test case design; session 1; and session 2. Prior to experiments, subjects were taught typical test design techniques and the ET approach. Then in the preparation phase, all subjects applied these techniques to either feature set A or feature set B. All subjects, regardless of which testing approach they first utilized, designed, and submitted, their test cases according to the same schedule. The test case design phase was not observed or restricted, but the used effort was recorded.

Both of the testing sessions followed the same agenda and consisted of a short instruction phase, a time-boxed 90-minute testing session, and finally filling in the survey form and submitting the results.

Table 1 Original experiment arrangements

Phase	Group 1	Group 2
Preparation	Test case design for feature set A	Test case design for feature set B
Testing session 1	TCT Feature set A	ET Feature set A
Testing session 2	ET Feature set B	TCT Feature set B

The single factor, the applied testing approach, had two alternatives: TCT and ET. The only significant blocking variable, representing the undesired, uncontrollable variations in the experimental design, was the tested feature set that could not be kept the same for all elementary

[4] False defect reports refer to reported defects that cannot be understood, are duplicates, or report non-existing defects.

[5] In the original study it says the average was 107.9. However, when doing re-analysis we found that three missing data points (=student that had not answered this question) had turned to zeros. Thus, changing zeros to missing increased the average slightly.

experiments. The experimental design forced the use of different feature sets and different sets of seeded and authentic defects in the two testing sessions.

This study looked at the defect detection efficiency measured by the number of defects found during a fixed-length testing session. Additionally, more insight into the efficiency was gained by considering the proportions of different defect types and severities as well as the number of false defect reports produced during a testing session. The data was collected in three ways. First, subjects submitted the predesigned test cases in an electronic format. Second, in the testing sessions, the subjects filled in test logs and defect report forms. Third, after each session, the subjects filled out a survey questionnaire. The data collection forms and templates are presented in the appendices of (Itkonen 2008).

The number of defects detected in ET and TCT sessions were compared using the t-test. In addition, multi-factorial analysis of variance (ANOVA) was used to analyze the effect of the different feature sets, and the possible interactions between the feature set and the testing approach. The defect distributions of the ET and TCT groups in terms of defect types were presented and the significance analysis of the type differences performed using the non-parametric Mann-Whitney test. Finally, the number of false reports was compared between the two approaches using the Mann-Whitney test.

2.4.4 Original Artifacts

The experimental units, i.e., the target of testing, in the original experiment were two variants of version 4.2 of the jEdit text editor with different sets of seeded defects. The scope of the 90-minute test sessions forced us to select small sets of functionality to focus on in the experiment. Two distinct and restricted feature sets were created and different sets of artificial defects were seeded into the selected feature sets. Feature set A had 25 seeded defects and feature set B had 24 seeded defects. The variants with seeded defects were not available to the subjects before the test sessions. The normal open-source version of the software was available to the subjects beforehand and they could familiarize themselves with the features and utilize the software when designing their test cases.

2.4.5 Context Variables

We think that the most important context variables of the original study were the use of students as subjects in test execution, the use of jEdit text editor as the target of testing, subjects creating test cases themselves, and the limited time for testing. Use of students in the original experiment really meant that results could only be used to establish a trend (Tichy 2000) and could be only generalized to novice industrial testers. In the original study, the jEdit text editor was selected for the system under test as the concepts and features of a text editor were familiar to students through the prior use of jEdit or similar text and code editing software. This meant that subjects were familiar with the application domain and the software at least on a conceptual level. Subjects created the test cases themselves, which introduced variations in how they performed TCT. However, a similar variation of the testing performed by the subjects also existed in the ET condition making the conditions comparable. Finally, an important context variable and a potential

threat to validity in the original experiment was that subjects had strictly limited time to perform test execution. This was visible in the survey data as many of the subjects reported about insufficient time for test execution in the original experiment.

2.4.6 Summary of the Original Results

In the original experiment, the subjects found fewer defects when using predesigned test cases (6.37 vs. 7.04 number of reported valid defects per subject, p=0.088). Thus, the difference between the two approaches did not allow rejecting the null hypothesis that assumed there is no difference in the number of detected defects when testing with or without test cases. The study also reports that the subjects used seven hours on average for test case design activities. This means that testing with predesigned test cases in this study took on average 8.5 hours, whereas testing without test cases took on average 1.5 hours. Still, the defect detection rates of the two approaches were not different, which makes the ET approach much more efficient.

The analysis of the defect report quality showed that the TCT approach produced twice as many false defect reports than the ET approach, and the difference was statistically significant. In terms of the defect types, the subjects found more of both the most obvious defects and the ones most difficult to detect when testing without test cases. In terms of the technical type, the subjects found more user interface defects and usability problems with the ET approach. The differences in defect types were not significant.

Regarding the perceptions of the subjects in the original experiment, no statistical difference was found in the perceptions of the easiness of the approaches or perceived test coverage. When asked about the usefulness of the approaches, both approaches were perceived more useful for guiding the testing than for revealing defects. The differences between the approaches were small but significant indicating that the TCT approach was perceived more useful both for guiding testing and revealing defects.

The perceived problems in the TCT were mostly related to the quality and shortcomings of the predesigned test cases. The problems in the ET approach included the lack of preparation, problems in logging, challenges in recognizing defects, and difficulties in performing the low-level test case design “on the fly” during the test execution. In the original experiment, both approaches initiated many comments on insufficient time available for testing.

The original research pointed out that the subjects did not perceive many problems regarding coverage or reproducing the occurred failures in ET, even though these two are presented in literature as some of the most severe shortcomings of the ET approach. Instead, in the original study, the subjects reported more coverage-related problems related to TCT than ET in the open questions, which could have been caused by the limited time of the first experiment.

3 Methodology

In this research, we conducted a dependent replication (Shull et al. 2008) of an original experimental comparison of the ET approach and the TCT approach. The replication was performed by the same researchers in the context of the same master’s level university course as in the original experiment, but using different student sample, i.e., students from a subsequent year, as the subjects.

3.1 Research Questions

The research problem in this experiment is the same as in the original experiment (Itkonen et al. 2007): *What is the effect of using predesigned and documented test cases in the manual functional testing with respect to defect detection performance?* The research problem is further split up into five more detailed research questions. The first research question of the original study (Itkonen et al. 2007) was extended by including the efficiency viewpoint and the fourth question of the original study (Itkonen 2008) was divided up into two distinct questions, RQ4 and RQ5, in this study to make the analysis and results clearer.

- RQ1: What is the effect of using predesigned test cases in terms of actual and perceived defect detection effectiveness and efficiency?
    - H1a: The ET approach has higher defect detection effectiveness.
    - $H1a_0$: There is no difference in the defect detection effectiveness.
    - H1b: The ET approach is more efficient when total efforts are taken into account.
    - $H1b_0$: There is no difference in the defect detection efficiency.
- RQ2: What is the effect of using predesigned test cases on the validity of the test results in terms of the number of invalid defect reports?
    - H2: The TCT approach leads to a higher number of invalid defect reports.
    - $H2_0$: There is no difference in the number of invalid defect reports.
- RQ3: What is the effect of using predesigned test cases on the types of the detected defects?
    - H3: The defect type distributions are different between the ET and TCT approaches.
    - $H3_0$: There is no difference in the defect type distributions.
- RQ4: What is the effect of using predesigned test cases on the perceived easiness, coverage, and usefulness for guiding testing?
    - H4a: The perceived easiness is higher for the ET approach.
    - H4b: The perceived coverage is higher for the TCT approach.
    - H4c: The perceived usefulness for guiding is higher for the TCT approach.
    - $H4_0$: There are no differences in terms of the perceptions.
- RQ5: What kinds of problems are associated with ET and TCT testing approaches?

For the RQ5, we cannot state a quantitative hypothesis, as this research question is qualitative. However, we thought that some of the perceived problems would be specific to one of the studied approaches and others would be common for both approaches. In the literature, there is not enough knowledge for us to base our hypothesis on the problems of the ET approach.

3.2 Experimental Design

In this research, we have conducted a replication of an experimental study on the effects of using predesigned test cases in contrast to a free-form ET approach. The context of testing was manual system level functional testing through the graphical user interface of jEdit text editor (see **Fig. 1**

for an example). The replication was performed by the same research group and same primary researchers as the original study. The effect of the independence of the replicating researchers has been discussed in the literature and the strengths and limitations were reported for both independent and dependent replications (Kitchenham 2008; Shull et al. 2008; Tsang and Kwan 1999; Vegas et al. 2006). The benefits of independent replication are the ability to avoid the possible biases of the original researchers and the ability to provide a conceptual replication in a different context from the original study. On the other hand, a need for communication between the original and replicating researchers has been identified in order to ensure successful replication (Vegas et al. 2006).

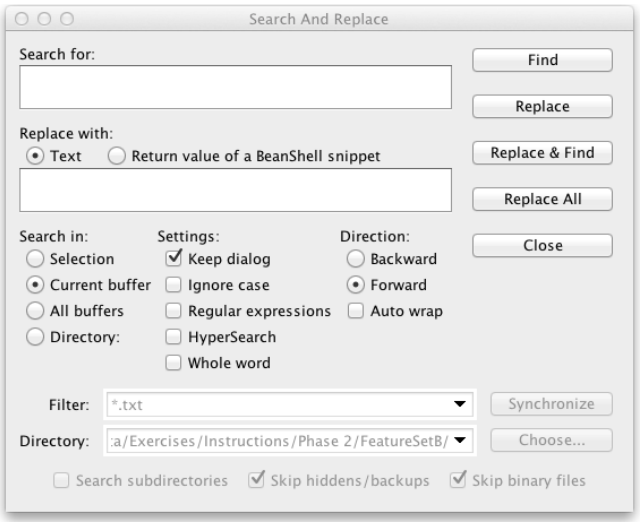

Fig. 1 Screen shot of one of the tested features of jEdit program (feature set B1)

We characterize our replication using the two-dimensional replication type classification by Tsang and Kwan (1999) as follows. The first dimension describes if the replication is performed using the same or different measurement and analysis. In this respect, our study falls in the category of same measurement and analysis. The second dimension divides the used data sets into three categories: same data set, same population, and different population. Our study falls in the same population category, since we used a sample of student populations taking the same undergraduate software testing course on subsequent years, i.e., we took a different sample from the same population.

The overview of the experimental design is described in **Table 2**. The experiment was designed as a balanced one factor design with a single blocking variable (Juristo and Moreno 2001). The studied factor was the applied testing approach with two alternatives: ET and TCT. The blocking variable was the tested feature set also with two alternatives FS B1 and FS B2. We used a paired design where all subjects performed both ET and TCT approaches, but with different feature sets. The subjects were randomly assigned to two groups that determined which one of the feature sets they used with each of the testing approaches. This way, the effect of the feature set was balanced.

The experiment activities were carried out as outlined in Table 2. In both approaches, the subjects performed manual testing. All subjects first designed the test cases for TCT approach in

the preparation phase and submitted their documented test cases. No predesign was required for the ET approach. In the preparation phase, the subjects had a deadline for submitting their test designs. The subjects had a few weeks of calendar time to perform the design tasks.

Table 2 Experimental design

Phase	Group 1	Group 2
Preparation	Test case design for feature set B1	Test case design for feature set B2
Testing[a]	TCT Feature set B1	TCT Feature set B2
	ET Feature set B2	ET Feature set B1

[a] The approach that was applied and the target feature set were controlled. The order of the testing activities was not controlled, but was recorded.

After the preparation phase, the subjects got the instructions for the testing (execution) phase and a deadline for submitting their results, including the defect reports, test logs, effort reports, and survey forms. Each subject performed both TCT and ET using the assigned feature sets as the target of testing. They had a few weeks of calendar time to accomplish the testing tasks. The order of the two testing approaches was not enforced, but most of the subjects (48/51) reported performing the TCT first, which is likely due to being familiar with the TCT feature set as they were previously forced to design the test cases for that feature set.

In TCT, the subjects used the test cases that they had designed and documented themselves at the previous phase to guide their test execution. For ET, the subjects were given a generic high-level test charter that is described in the section 3.2.1. Note that the groups were only used in experimental design for assigning the two feature sets randomly, but all the testing activities were individual work.

3.2.1 Testing Approaches and Training the Subjects

The instructions for the subjects were given on the course web pages. In addition, the arrangements were introduced during the course lectures.

The two alternatives of the studied factor, testing approach, were ET and TCT. The subjects were instructed to apply the functional test design techniques that they were taught during the course they were undertaking. These techniques were equivalence class partitioning (Myers 1979), boundary value analysis (Myers 1979) and combination testing using base choice (i.e., default) or pair-wise strategy (Cohen et al. 1997). These techniques were taught to the subjects in the course lectures and by going through few practical examples. The summary of the relevant teaching activities is provided in the **Table 3**. The course consisted of 13 lectures and the length of each lecture was 90 minutes. In addition to actual teaching sessions, the students were provided a 45-minute tutorial on the practical arrangements and guidance of each exercise phase.

Five of the course lectures were directly related to the testing activities of this experiment (see **Table 3**). After each lecture, the students were engaged in the individual study activities, meaning further studying the taught topics from the textbook[6] and teaching materials, guided by set of

[6] Burnstein I (2003) Practical Software Testing. Springer-Verlag, New York. (selected chapters)

provided questions to support their learning. The effort used for individual studies was not recorded.

Table 3 Summary of training given to subjects prior to the testing exercises

Teaching method	Topic	Duration
Lecture	Introduction and motivation to testing techniques	45 min
Individual study	Introduction and motivation to testing techniques	duration not available
Lecture	Black-box testing techniques	90 min
Individual study	Black-box testing techniques	duration not available
Lecture	Combination testing techniques	45 min
Individual study	Combination testing techniques	duration not available
Lecture	Practical guidance for test case design	45
Lecture	Exploratory testing	90 min
Individual study	Exploratory testing	duration not available

The testing techniques that were applied in this experiment were taught to students during three separate lectures and related individual studying. In addition to teaching the actual testing techniques in three separate lectures, the training included an introductory lecture to the TCT approach. This lecture covered the traditional concepts of describing test designs as test cases and motivation and reasoning for test case design and documenting tests.

The ET approach is testing where the individual tester has a lot of control and opportunities to guide the work during the actual testing activity, based on what the tester continuously observes and learns about the tested system. The ET approach has no detailed instructions on how to perform the actual testing activities. Subjects were trained by giving them a lecture on the ET approach, where the characteristics of ET were explained and its differences to TCT approach were elaborated.

The training and the lectures aimed at a neutral coverage of the two approaches, presenting the proposed strengths and weaknesses for both of the approaches. The training and the used experimental artifacts were the same as in the original experiment. Also, the teacher was the same person.

The ET testing was guided by the ET charter that described the high-level goal and guidance for the testing activity. The content of this charter included 1) the description of the tested areas, i.e., the list of the features to be tested; 2) a description of the high-level goals and viewpoint of testing, i.e., stating the typical end user's viewpoint and listing some guiding questions about what to pay attention to; 3) practical guidance on how to test the features, including instructions to use the user documentation as well as personal knowledge, and the basic testing techniques to be applied. The same document was used by the subjects to log their testing and findings during testing (the actual bugs were recorded also in the Bugzilla database, see Section 3.4). The charter did not detail any actual tests. This charter is included in the Appendix B.

3.2.2 Application of Techniques

Inside of the testing approaches ET or TCT, the students had the liberty to apply the different testing techniques described above as they saw fit. In this sense, this paper differs from many testing technique experiments, e.g., (Juristo et al. 2012; Mouchawrab et al. 2011) where the

subjects are forced to use a single testing technique such as equivalence partitioning. Such single technique studies offer good information comparing detailed testing techniques. Single technique design is more rigorous and offers more detailed information about individual techniques, whereas our approach offers information comparing higher-level testing approaches where the subjects could apply several different testing techniques. We think that our design is a good match with industrial practice where testers and test designers are likely to apply several different low-level techniques. We see that future works are needed to combine the different study types to understand how the detailed testing techniques work and are applied inside of the testing approaches. For example, we currently cannot offer data on how often each of the techniques was applied inside the approaches, if the techniques were combined, or which technique worked best for each approach.

3.2.3 Experiment Artifacts

The experimental artifact was version 4.2 of the jEdit text editor software with two different sets of seeded defects in two different feature sets. Two distinct feature sets were used for the testing assignments and different sets of artificial defects were seeded into the selected feature sets. The tested features in this experiment were same as the feature set B in the original experiment. The original feature set B was split into two feature sets B1 and B2 in this experiment (see **Fig. 1** for an illustration). The exact features that were included in each of the feature sets in the experiments are described in Appendix C.

The tested software version was a production release of this open-source project that was seeded with additional defects for the research purposes. The total number of identified unique defects in the feature set B was 48 of which 24 were artificially seeded. Thus, we think that the tested features were similar to the quality level of pre-release features of this type of software.

3.2.4 Subjects

The subjects were master's level students who were taking a software testing and quality assurance course as part of their software engineering studies in Aalto University, Finland. The number of subjects in the analysis was 51. The total number of students performing the assignments was 52, but the data of one subject had to be excluded from the analysis due to completing only part of the assignments.

We collected the demographic data of the subjects to characterize them in terms of their phase of M.Sc. studies and experience in software development and testing (see Table 4). The share of subjects with no professional programming experience was 29%, and 57% had no professional experience in testing. We can see that in the replication the subjects were a bit more experienced in terms of credits and professional experience. In the first year, the students had slightly higher passing rate for the course and they had spent more years studying in the university. We studied the statistical differences between groups (t-test for credits and Mann-Whitney test for others) and found that the only statistically significant difference was that the replication students had higher professional testing experience in average. The testing assignments that the subjects performed

were graded. The grading was based on the subjectively evaluated quality of their predesigned test cases and defect reports, and the number of defects they found and reported.

Table 4 Demographics of the subjects

	Original Study (n=79)			Replication (n=51)		
	mean	median	st. dev.	mean	median	st. dev.
Credits	113	110	36.80	124	120	33.9
Years of study	4.75	4	1.77	4.58	4	1.46
Passing grade (1-5)	3.48	4	1.10	3.50	4	1.40
Years of professional programming experience[a]	2.92	2	2.77	2.01	2	1.38
Years of professional testing experience[a]	2.07	1.5	1.36	1.36	1	1.19
% of individuals who passed the course	94%			88%		
% of individuals with professional programming experience	63%			71%		
% of individuals with professional testing experience	21%			43%		

[a] Individuals with zero years of experience were excluded from calculations

3.3 Changes to the Original Experiment

In this experiment, the original experiment (Itkonen et al. 2007) was replicated in the same context by applying the original experimental design with four alternations.

First, the time restriction for testing sessions of the original design was removed in order to get better understanding about the effort requirements and true effectiveness of the two approaches. In addition, this change in design allowed us to gain understanding about the effect of the time restriction on the testing approaches. Furthermore, this change also removed one potential threat to validity. The strict time limitation in the original experiment might have too severely limited some students testing activities, especially in TCT. If the time restriction is too hard, it can lead the TCT task to deviate too much from the task we are trying to study, e.g., if the time restriction leads to situation where logging the findings takes major share of the session time or the subjects are able to utilize only a small part of their planned test cases.

Second, in order to allow subjects use the time they needed, the testing activities were not performed in controlled classroom sessions as in the original experiment.

Third, we narrowed the scope of the testing task that was given to the subjects. This was implemented by splitting one of the two original feature sets (feature set B) to form the two feature sets for the replication. The target software application under test was the same and had the same defects; only the target feature set was limited to a subset of the original. The reason for narrowing the scope was to make the task smaller for the subjects. This ensured that in the non-time-restricted condition the subjects could perform the testing tasks thoroughly with reasonable amount of effort for the course context.

Fourth, the change in the design also restricted our ability to control the order of the treatments, which meant that the order of the two testing approaches was not enforced. Most of the subjects (48/51) reported performing TCT first, which is likely due to being familiar with the TCT feature set as they had designed the test cases for the feature set in the previous phase of the exercise.

However, we were able to keep all other aspects of the experiment as identical to the original experiment as possible, including the course context, student population, and training. The number

of subjects was lower than in the original experiment, and this was due to the decrease of the students on the software testing course that this experiment was part of.

3.4 Data Collection

The data collection methods were the same as in the original experiment. The data was collected in three forms. First, the subjects reported all their findings in the form of defect reports. Second, the subjects reported their used effort in each phase of the assignments. Third, the subjects filled a survey form after the testing assignments. All data was self-reported by the subjects and no direct observations were performed by the researchers. However, the students were required to keep a test log and write test notes of their testing and to submit their test cases. This was not used in the data analysis, although it allowed us to ensure that students had actually performed the tasks.

3.4.1 Defect Data

The dependent variables in this experiment were the number and types of the detected defects and the number of false defect reports. The subjects reported the found defects into a Bugzilla[7] defect database that was provided for them. The subjects were required to give a full report of the defects including steps to reproduce, expected and actual outcome, and severity assessment.

3.4.2 Effort Data

The effort data was collected from the subjects by using electronic forms. The subjects logged in the form the date and the amount of effort they used for each experiment task in work hours. The forms were filled during the assignment period by the subjects and submitted together with the test results and survey forms. The subjects were clearly informed that the effort data was collected purely for research purposes and was not used for grading the assignments.

3.4.3 Survey Form

A survey form was used to record the perceived effects of the testing approaches and the subjects' experience. The survey was constructed to measure the perceptions of the subjects regarding the easiness and usefulness of the two testing approaches. Closed form multiple-choice questions were used in order to record the perceptions of all subjects. The aim was to construct simple questions to get basic understanding if the perceptions would clearly differ between the approaches. The questionnaire was not based on any validated instrument, and thus we used straightforward ordinal scale questions. In addition, open questions were used to collect data on what shortcomings and problems the subjects faced when applying each of the approaches. An open question format was chosen because we did not want to restrict the subjects to any predefined set of typical problems. We also believe that the answers to open questions are true problems the subjects faced and important to them as they used the time to describe them in the form.

The first part of the survey collected data of the student subjects' phase of studies in terms of study credits and years, and experience in software development and testing in years. In the second part, the subjects assessed the coverage of their testing by assessing the coverage of each area

[7] http://www.bugzilla.org/

inside the feature sets on a four-step ordinal scale. The third part asked the subjects' perceptions of the easiness and usefulness of both testing approaches with a seven-step ordinal scale: 1) how easy it was to apply each testing approach, 2) how useful was each approach in finding defects, and 3) how useful were the documents of each testing approach in structuring and guiding test execution. In the TCT condition, the referred documents were the predesigned test cases and in the ET condition it was a high-level test charter provided by the course staff, see Appendix B. Finally, we used an open question to collect problems and shortcomings that the subjects faced when applying each of the testing approaches. The subjects filled the survey form after performing both testing assignments and submitted the form as part of their test results. The survey questions are presented in the Appendix A.

3.5 Data Analysis

In data analysis, we use both statistical analysis on the quantitative data and qualitative analysis to analyze the perceived challenges that were collected using open questions in the survey form. The raw data of the replicated experiment is uploaded to research data sharing service[8] where it is publicly accessible (Itkonen 2013).

3.5.1 Defect Data

The subjects record the defect reports into an online defect database (Bugzilla) during testing and submit a report listing all their defect reports after the testing activities. The defect data is coded so that each finding of the subjects is coded either with a defect ID representing a known (to the researchers) defect[9] or as a false defect. All known defects are classified by the researchers regarding their severity, technical type, and difficulty of finding. These classifications are used in the defect type analysis.

The defect data is presented by box-plots and distributions regarding type classifications. The defect defection effectiveness and efficiency are analyzed by comparing the means and using paired analysis with a t-test. The effect sizes for t-tests are calculated using Cohen's *d* (Ellis 2010), i.e., mean difference/sd difference. A Wilcoxon Signed Rank Test is used in analysis when normal distribution of the data can not be assumed. We use two-tailed tests and alpha level 0.05 for all statistical tests.

The power of the statistical tests affects the conclusion validity. We perform the power analysis of our statistical tests regarding the defect data. If the power of tests is low, it is possible that the test is not able to show a true difference in the data in case the effect size is small.

3.5.2 Survey Data

The survey data (RQ4) is presented as box-plots presenting the comparison between the testing approaches. Since the survey data was not presented in the original publication intended for wider audience (Itkonen et al. 2007) (although it was collected and presented in the thesis (Itkonen 2008)), we present the survey data for both the original experiment and the replication. The

[8] http://figshare.com/
[9] If a previously unknown, valid defect was reported a new known defect and ID was created.

statistical tests are performed with R[10] using paired analysis with Wilcoxon two-sided rank test, i.e., Mann-Whitney test. The Wilcoxon rank test is the correct choice for our data analysis as, unlike the t-test, the Wilcoxon test makes no assumption of the distribution of the data and it can be used for an ordinal scale data that was the output of our survey. We also calculate the effect sizes (Ellis 2010) for significant differences with *r* measure as instructed in (Field 2005; Yatani), i.e., $r = {Z}/{\sqrt{N}}$, where Z is the z-score of Wilcoxon test and N is the number of samples.

The open questions regarding the problems of the testing approaches (RQ5) are analyzed qualitatively by coding all open text answers with Atlas.ti program. Based on the coding, the main topics of the problems in both testing approaches were identified. The main topics are such problems that are identified at least by five subjects.

4 Results and Discussion

In this section, we present the results of the replication experiment regarding the defect detection effectiveness, efficiency, and the perceptions of the subjects. We present the comparison to the results of the original experiment along with the results. In the discussion, we also present possible hypotheses related to the effects of the existence of a time restriction in testing activities that was the main difference in the experimental design between the original experiment and the replication.

4.1 RQ1: Defect Detection Effectiveness and Efficiency——Measured and Perceived

In this section, we present the results regarding the first research question. The main response variable in this experiment was the number of defects a subject detected during the testing activities. In addition, the effort data was collected to analyze the efficiency aspect and, finally, the perceived usefulness for defect detection of the two approaches was measured using a survey. We used paired analysis in statistical tests when comparing the two approaches. We identified significant individual level correlations between the used effort in ET and TCT approaches and in the defect counts between the approaches. These characteristics of the data support the choice of paired analysis. This way the individual differences, e.g., in general motivation and testing skills and experience do not interfere as much with the effect of the two treatments.

4.1.1 Measured Defect Detection Effectiveness

The box-plot in **Fig. 2** illustrates that the defect counts are slightly lower and the direction of the difference between approaches opposite in the replication experiment (note: in all box-plots outlier dots are subjects who either performed exceptionally well or poorly, but they are included in the statistical analysis).

The defect count data of this replication is summarized in Table 5 and a comparison of statistical analysis is in Table 6. The number of found defects refers to how many different individual defects all subjects found together. The total numbers of unique defects all subjects detected together with ET and TCT approaches were 40 and 44, respectively. The absolute mean defect counts per subject for the ET and TCT approaches were 5.47 and 6.06, respectively. The

[10] *www.**r**-project.org*

difference between the ET and TCT approaches in terms of the absolute defect counts was 0.59 defects more in TCT. The difference is not statistically significant using paired analysis with two-tailed *t*-test (p=0.093). The result does not allow rejecting the null hypothesis $H1a_0$: There is no difference in the defect detection effectiveness.

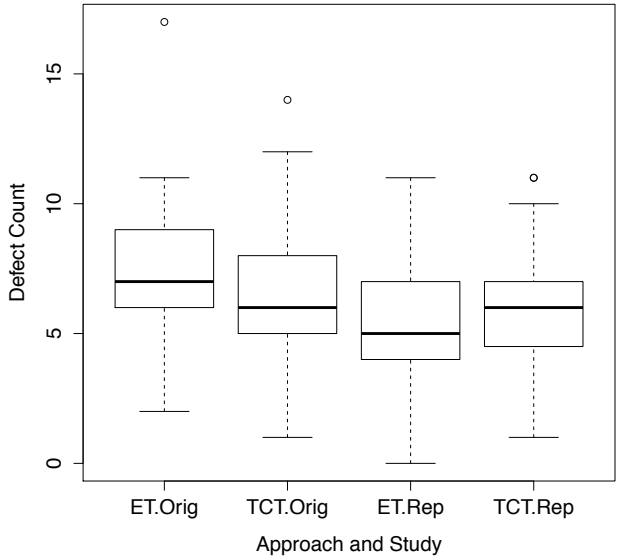

Fig. 2 Box-plot of the defect counts in the two approaches
(Orig = Original study (Itkonen et al. 2007), Rep = Replication presented in this paper)

Table 5 Defect count data of this study

Testing approach	Number of found defects	Found defects per subject	
		Mean	St. dev.
ET	40	5.47	2.148
TCT	44	6.06	2.310
Total	46	5.76	2.239

Table 6 Statistical analysis of defect count data in this replication and in the original study

Statistical analysis	Original study	Replication
Absolute effect size	ET finds 0.67 more defects	TCT finds 0.59 more defects
Standardized effect size: Cohen's d	-0.27	0.27
t-test	p=0.088	p=0.093

The absolute effect sizes in the original and replicated experiment were 0.67 and 0.59 defects, respectively. The standardized effect sizes using Cohen's d were -0.27 and 0.27 for the original and replication, respectively, indicating a small effect. Note that the absolute defect counts are not directly comparable between the experiments, because the target feature sets and the number and type of the actual defects were different. However, the replicated result supports the original conclusion that the difference between the approaches is small and not significant. The change in the direction of the effect can be related to the time restriction condition. In the original experiment, the test execution time was strictly limited, which may prevent testers to get the full

advantage of the predesigned test cases. In the replication, the subjects were allowed to use as much time as they needed to carry out the testing, which allows them to take full advantage of their test cases. Thus, we hypothesize that the time restriction condition could affect the applicability of a testing approach.

4.1.2 Defect Detection Efficiency

In this experiment, we did not restrict or control the amount of effort the subjects were allowed to use for the testing assignments. The instructions were that the subjects should aim at testing the given feature sets as thoroughly as possible to get good understanding of the actual quality level. The reported effort data is presented in **Table 7**. The test design and execution effort are separated for the TCT approach, but only the total effort is presented for ET approach since the test design and execution are parallel and inseparable activities in ET. We measured the defect detection efficiency in found defects per hours of testing effort. This metric was computed from the subjects' self reported effort data and numbers of reported valid defects. The subjects used less time for ET approach in total. Even if we consider only the test execution effort in the TCT approach, the mean total effort for ET was lower. The test design activity took, on average, almost three times more effort than the execution in TCT approach.

Table 7 Used effort and efficiency in this replication

	Testing effort (hours)				Defect detection efficiency
	Mean	Min	Max	Std. dev.	Defects/hour
ET total	4.58	1.00	13.00	2.283	1.49
TCT total	19.47	7.75	38.00	6.858	0.33
TCT execution	5.35	1.00	11.00	2.407	1.32
TCT design	14.12	5.00	30.00	5.659	—

The efficiency analysis of the test execution, excluding the TCT test design effort, results to 1.49 defects/h for ET, and 1.32 defects/h for TCT approach (see **Table 7** and **Table 8**). This means that even though TCT approach revealed more defects in absolute numbers, the ET approach was more efficient in revealing the defects during test execution. However, the difference in test execution efficiency is not statistically significant using paired analysis with 2-tailed t-test ($p=0.130$).

When the test design effort is accounted for in the case of TCT, the efficiency numbers naturally turn very clearly to favor the ET approach (see **Table 7** and Table 9). The mean efficiency for TCT when both test design and test execution effort were included was 0.33 defects/h. The difference to ET efficiency is 1.15 defects/h and statistically significant using paired analysis with 2-tailed t-test ($p=0.000$).

In the original study, the efficiency numbers of test execution are equal to the effectiveness numbers as the effort was kept constant. We cannot compare the absolute efficiency numbers between the original and replicated experiment since the target feature sets, and thus the number of existing defects to find, were different. However, we can compare the differences in the standardized effect size in each experiment (see Table 8 and **Table 9**).

Table 8 Efficiency when excluding test-case-design effort

Statistical analysis	Original study	Replication
Absolute effect size	ET finds 0.34 more defects per hour	ET finds 0.17 more defects per hour
Effect size: Cohen's d	-0.27	-0.19
t-test	p=0.088	p=0.130

The test execution efficiency, when excluding the design effort, is better in the ET approach both in this replication and in the original experiment, and the effect size is smaller in the replication than in the original experiment. Standardized effect sizes for the efficiency when excluding the design effort were -0.27 and -0.19 for the original and replicated experiment, respectively. These effect sizes indicate a small effect.

When accounting for the TCT design effort, the standardized effect sizes are -1.18 and -1.33 for the original and replicated experiment, respectively. These effect sizes indicate a large effect. In the original study, the effort used for test preparation was not analyzed. For this paper, we re-analyzed the data and the results are presented in Table 9.

Table 9 Efficiency when including test case design effort

Statistical analysis	Original study	Replication
Absolute effect size	ET finds 1.64 more defects per hour	ET finds 1.15 more defects per hour
Effect size: Cohen's d	-1.18	-1.33
t-test	p=0.000	p=0.000

In this replication, the used test execution effort in TCT and number of found defects correlate and the full TCT effort correlates with the number of found defects (see **Table 10**). Interestingly, in ET approach the correlation between the testing effort and the number of detected defects was very low (0.025; p=0.86). For the original study, such correlation cannot be counted as the test effort was fixed.

Table 10 Pearson correlation between effort used and number of defects found in this replication

Testing approach	Correlation coefficient	p-value
TCT execution effort	0.446	0.001
TCT total effort	0.383	0.006
ET execution effort	0.025	0.86

These results support the results of the original experiment and allow rejecting the null hypothesis $H1b_0$. Thus, the ET approach is more efficient when total efforts are taken into account.

These results indicate that when a more systematic approach with detailed test design and documentation is applied, the defect detection effectiveness is not significantly improved, but the used effort is highly increased. This result is similar to the earlier finding comparing boundary value analysis and a more detailed condition coverage testing technique. Juristo et al. (2004) concluded that it is better to use boundary value analysis than condition coverage, as there is no difference in effectiveness and it takes less time to detect and isolate faults.

4.1.3 Perceived Usefulness for Defect Detection

The results on the perceived usefulness of each testing approach for defect detection are in **Fig. 3** and the statistical analysis are in Table 11, Table 12, and Table 13. When comparing the approaches ET and TCT with the Wilcoxon paired two-sided rank test, we found that in the original study the TCT was perceived significantly more useful for defect detection with small effect size of 0.243 (see Table 11). However, for the replication, the Wilcoxon paired two-sided rank test showed that effect was reversed as ET was perceived more useful but with smaller effect size -0.149. When comparing between studies, i.e., the condition with the respect of having a fixed time box of 2h (original) vs. having no time restriction (replication), we found with the Wilcoxon two-sided rank test that both approaches are perceived more useful when more time was available; for ET the effect size was 0.580 and for TCT effect size of 0.134. Thus, we cannot conclude that there would be perceived difference between the approaches in relation to defect detection effectiveness as the two experiments offer conflicting results. However, both approaches were perceived more useful when more time was available for the subjects.

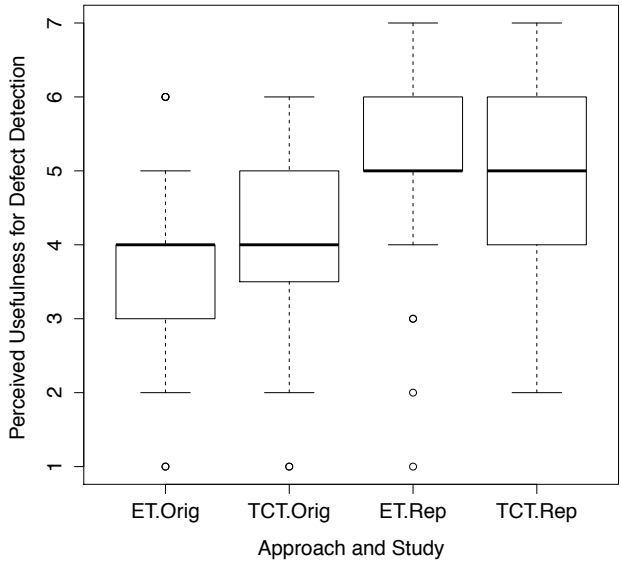

Fig. 3 Box-plot of the perceived usefulness for detecting defects
(Orig = Original study (Itkonen 2008), Rep = Replication presented in this paper)

Table 11 Perceived usefulness for defect detection between testing approaches

Statistical analysis	Original study	Replication
Direction	TCT perceived more useful	ET perceived more useful
Effect size: Pearson's r	0.243	-0.149
Wilcoxon test	p=0.001	p=0.0657

Table 12 Perceived usefulness for defect detection of testing approaches when more time available

Statistical analysis	ET	TCT
Direction	Perceived more useful when more time available	Perceived more useful when more time available
Effect size: Pearson's r	0.580	0.134
Wilcoxon test	p=0.000	p=0.0628

When we compare the perceived usefulness of each approach for defect detection with measured defect detection effectiveness, we find that in the original study TCT is perceived more effective but is measured as less effective than ET. In the replication, the situation is exactly the opposite, TCT is perceived less effective but it is measured as more effective than ET. When we study the perceived effectiveness of each technique at the individual level (see **Table 13**), we find that for TCT we have significant Kendall's Tau correlation between perceived TCT effectiveness and measured TCT effectiveness with value of 0.28 (p=0.001) in the original study and 0.39 (p=0.000) in the replication. However, for ET, there is no significant correlation in the original study (Kendall's Tau is -0.06), but in the replication there is a correlation of 0.24 (p=0.041).

Table 13 Correlation between perceived usefulness for defect detection and measured effectiveness

Testing approach	Original study	Replication
TCT	0.28 (p=0.001)	0.39 (p=0.000)
ET	-0.06 (p=0.527)	0.24 (p=0.041)

We conclude that there are no differences in the perceived usefulness for defect detection between the approaches. However, it seems that for TCT only there is an effect between the perceived usefulness for defect detection and the measured defect detection effectiveness. Finally, both techniques are perceived as more useful for defect detection when more time is available for the subjects.

4.2 RQ2: Quality of the Defect Reports

The quality of the test reporting was measured by the number of false reports (see **Fig. 4** and **Table 14**). With false reports, we refer to reported defects that cannot be understood, are duplicates, or report non-existing defects. These false reports were analyzed both as absolute numbers and as a proportion of all reported findings. As absolute numbers, the ET approach produced a mean value of 1.84 false reports per subject and the TCT approach produced mean value of 2.90 false reports per subject (see the box-plots in **Fig. 4**).

Since the false report data was not normally distributed, we used 2-tailed Wilcoxon Signed Rank Test for statistical analysis. The difference in absolute false report numbers is statistically significant (p=0.006). The false report percentages[11], called precision in (Baeza-Yates and Ribeiro-Neto 1999), for the ET and TCT approaches were 22.6% and 27.7%, respectively. The difference in the false report percentages was also significant (p=0.018). This result confirms the result of the original experiment and allows rejecting null hypothesis $H2_0$. Thus, we conclude the TCT approach leads to higher number of invalid defect reports.

[11] This is the number of false defect reports divided by all findings (both real and false defect reports).

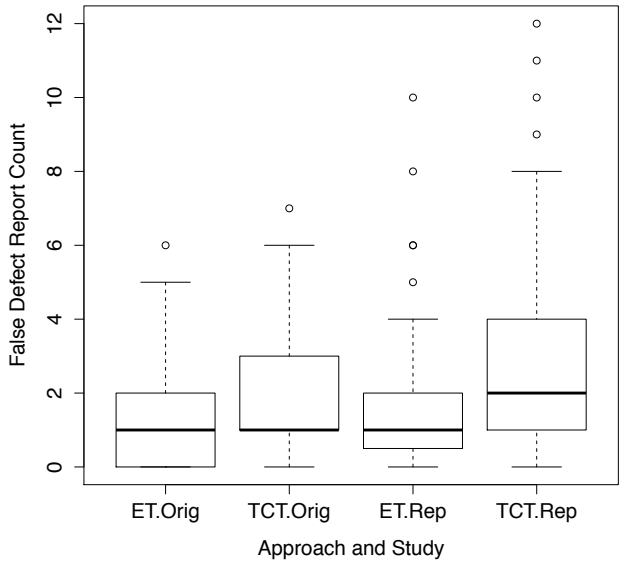

Fig. 4 Box-plot of the false defect counts

(Orig = Original study (Itkonen et al. 2007), Rep = Replication presented in this paper)

Table 14 False defects reports between ET and TCT

	False defects	False defect percentage
Absolute Effect size	TCT produced 1.06 more false reports	TCT had 5.12 percentage points higher false defect percentage
Effect size: Cohen's d	-0.50	-0.26
Wilcoxon test	p=0.006	p=0.018

The number of false reports was correlated (Kendall's tau) with the used effort (0.320; p=0.002) and the number of reported defects (0.348; p=0.001) in the TCT approach. The correlation in the number of false reports between the approaches was weaker (0.285; p=0.010). These results suggest that producing false reports is a somewhat individual characteristic and related to higher amount of effort and reports in general.

Regarding the number and proportion of false reports, both experiments reported significant difference showing more false reports in the case of TCT approach. The results strengthen the hypothesis that using test cases might affect the tester's ability to analyze the findings and understand the actual failure phenomenon; instead, the tester focuses on individual test cases and reports individual discrepancies regarding the documented test cases without considering the meaning of the symptoms. This might lead to high number of duplicate reports and even incorrect reports if faulty test cases are not recognized. The quality of reporting is an important aspect of the efficiency of testing. If the test results include large number of duplicate and incorrect findings it increases the costs of further analysis and filtering out the relevant information. The identified phenomenon is especially worrying, because in these experiments the test cases were designed by the same person who was executing the tests and thus, the test designs should be clear for the tester. In addition, the correlation between the number of valid and false reports indicates that this problem of report quality does not concern only poor performers and is not related to inability to understand the features under test or lack of testing skills.

4.3 RQ3: Types of the Defects

We further analyzed the types of the reported defects from three viewpoints. The defect types were analyzed using the same classifications that were used in the original experiment (Itkonen et al. 2007): first, the detection difficulty in terms of the number of interacting conditions that make the failure occur; second, the severity of the defect classified by the researchers; and third, the technical type of the defects. The results of this analysis are presented in Tables 15–17. Each of the tables presents the absolute numbers of defects in each category as well as the percentages for both of the studied approaches. From the percentages, it is clear that the defect distributions regarding these three dimensions do not differ between the ET and TCT approach and the null hypothesis, $H3_0$: There is no difference in the defect type distributions, cannot be rejected.

Table 15 Detection difficulty distribution

Mode	ET number	% of ET	TCT number	% of TCT	ET/TCT	Total number	% of all
0 = easiest	28	10.0 %	30	9.7 %	93.3 %	58	9.8 %
1	123	43.8 %	121	39.2 %	101.7 %	244	41.4 %
2	118	42.0 %	142	46.0 %	83.1 %	260	44.1 %
3 = hardest	12	4.3 %	16	5.2 %	75.0 %	28	4.7 %
Total	281	100.0 %	309	100.0 %	90.9 %	590	100.0 %

Table 16 Defect severity distribution

Severity	ET number	% of ET	TCT number	% of TCT	ET/TCT	Total number	% of all
Critical	31	11.0 %	30	9.7 %	103.3 %	61	10.3 %
Severe	81	28.8 %	89	28.8 %	91.0 %	170	28.8 %
Normal	114	40.6 %	128	41.4 %	89.1 %	242	41.0 %
Minor	50	17.8 %	55	17.8 %	90.9 %	105	17.8 %
Negligible	5	1.8 %	7	2.3 %	71.4 %	12	2.0 %

Table 17 Technical type distribution

Type	ET number	% of ET	TCT number	% of TCT	ET/TCT	Total number	% of all
GUI	44	15.7 %	52	16.8 %	84.6 %	96	16.3 %
Inconsistency	0	0.0 %	2	0.6 %	0.0 %	2	0.3 %
Manual	1	0.4 %	2	0.6 %	50.0 %	3	0.5 %
Missing function	68	24.2 %	76	24.6 %	89.5 %	144	24.4 %
Performance	13	4.6 %	10	3.2 %	130.0 %	23	3.9 %
Technical defect	19	6.8 %	22	7.1 %	86.4 %	41	6.9 %
Usability	7	2.5 %	7	2.3 %	100.0 %	14	2.4 %
Wrong function	129	45.9 %	138	44.7 %	93.5 %	267	45.3 %
Total	281	100.0 %	309	100.0 %	90.9 %	590	100.0 %

The results of the replication support the original results. Both experiments concluded that there are no significant differences between the approaches in terms of defect severity, type, or detection difficulty. The distributions of defects were even more similar between the approaches in

this replication experiment. This finding does not support the hypothesis that ET and TCT approaches would detect different types of defects, at least according to the used classifications in our data, and the $H3_0$ cannot be rejected. This is not very surprising finding if we consider that both approaches were manual, system level, functional testing where the input material and the knowledge and experience of the testers were identical. Furthermore, the actual test design techniques, e.g., boundary-value analysis, that were taught to the students were same in both approaches. There are existing results indicating that testing technique effectiveness is independent from the fault type (Juristo et al. 2012). However, one could assume that the ET approach to testing could be more suitable for detecting certain types of defects, regarding difficulty of finding or visibility on GUI level.

4.4 RQ4: Perceived Easiness, Usefulness for Guiding, and Coverage

We asked the subjects of the perceived easiness of applying each approach, the perceived guiding provided by each technique, and the perceived coverage.

The results of perceived easiness of each approach are in **Fig. 5** and in **Table 18**. For original study, the TCT is perceived easier than ET but statistical Wilcoxon test does not show significant different. However, for the replication the effect size was reversed; ET is perceived easier but the difference is not statistically significant. We conclude that there is no difference in the perceived easiness between approaches.

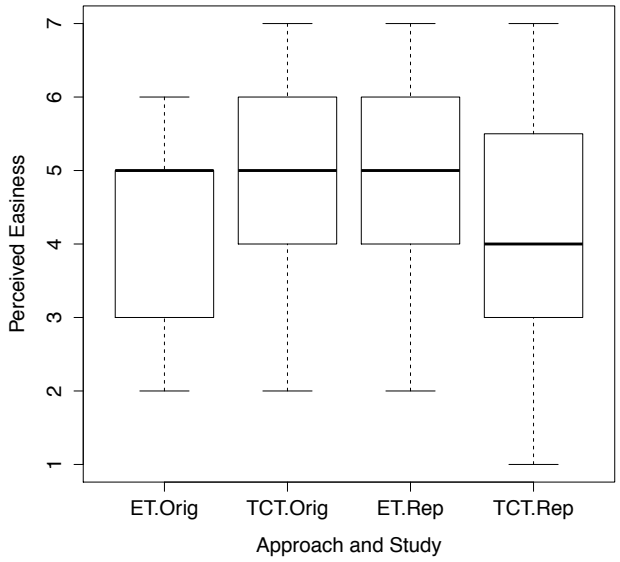

Fig. 5 Box-plot of the perceived easiness to apply each approach
(Orig = Original study (Itkonen 2008), Rep = Replication presented in this paper)

Table 18 Perceived easiness of testing approaches

Statistical analysis	Original study	Replication
Direction	TCT perceived easier	ET perceived easier
Wilcoxon test	p=0.095	p=0.218

The results of perceived usefulness for guiding and structuring software test execution of each approach are in **Fig. 6** and **Table 19**. From the figure, we can see that in both years TCT is

perceived better in guiding. In the original study and in the replication, the Wilcoxon paired two-sided rank test showed non-significant p-value. Since in both years the effect is on the same direction, i.e., TCT perceived better for guiding testing, we combined the data of both years and performed the Wilcoxon paired two-sided rank test. The results showed a significant p-value but small effect size of r=0.130. Thus, we conclude that TCT is perceived as better for guiding and structuring manual software test execution; the difference between approaches is small but significant.

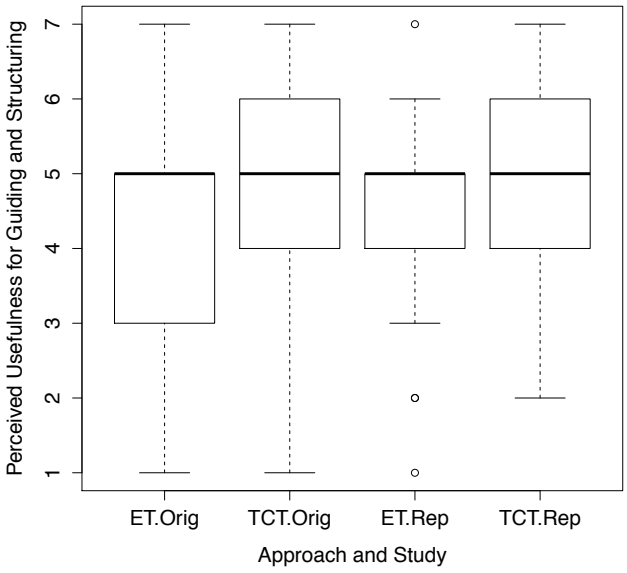

Fig. 6 Box-plot of the perceived usefulness for guiding and structuring testing
(Orig = Original study (Itkonen 2008), Rep = Replication presented in this paper)

Table 19 Perceived usefulness for guiding and structuring software test execution

Statistical analysis	Original study	Replication	Combined data
Direction	TCT perceived better	TCT perceived better	TCT perceived better
Wilcoxon test	p=0.062	p=0.141	p=0.017
Effect size Pearson's r	r=0.121	r=0.107	r=0.130

The results of the perceived coverage of each approach are in **Fig. 7** and in **Table 20**. The perceived coverage was measured by asking subjects how well each of the five to seven function areas was covered in their testing. To provide a single measure of each subject's coverage for each technique, we calculated the mean value[12] of the subject's responses. TCT is perceived to produce better coverage in both years. In both years, the Wilcoxon paired two-sided rank test showed statistically significant values with effect sizes of r=0.143 and r=0.305. Since in both years the effect is on the same direction, i.e., TCT is perceived better for coverage, we combined the data and calculated the effect size of r=0.221. Also whether individuals were under time restriction (Original) or not (Replication) produced a significant effect. We found with the Wilcoxon two-

[12] The authors are aware of the controversy of calculating the mean value from the ordinal data. However, we felt that using mean would be more accurate than median, e.g., if an individual respondent's coverage estimate median for ET and TCT coverage are both three, it could be result of ET coverage having a mean of 2.6, while TCT coverage would have mean value of 3.4. Obviously, in such a case, the respondent's intention would be that TCT provided better coverage, but it would not be visible in the perceived coverage measure using the median.

sided rank test that both approaches are perceived to produce better coverage when having no time restriction. For ET, the p-value was = 0.02246 (W = 2400.5) and the effect size was 0.175. For TCT, the p-value was 0.000 (W = 2874) and effect size was 0.344. Thus, we concluded that TCT produces better perceived coverage than ET and that both techniques produce better perceived coverage when more time is available for testing.

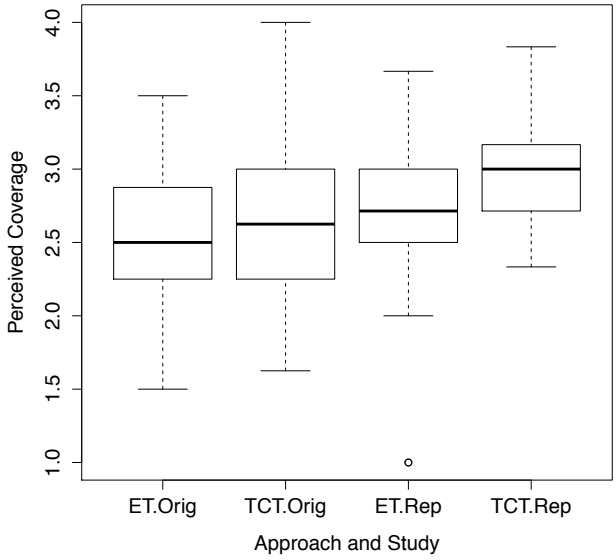

Fig. 7 Box-plot of the perceived coverage of testing
(Orig = Original study (Itkonen 2008), Rep = Replication presented in this paper)

Table 20 Perceived coverage

Statistical analysis	Original study	Replication	Combined data
Direction	TCT produced better perceived coverage	TCT produced better perceived coverage	TCT produced better perceived coverage
Wilcoxon test	p=0.035	p=0.001	p=0.001
Effect size Pearson's r	r=0.143	r=0.305	r=0.221

Table 21 Perceived coverage of testing approaches when more time available.

Statistical analysis	ET	TCT
Direction	Perceived higher coverage when more time available	Perceived higher coverage when more time available
Effect size: Pearson's r	0.175	0.344
Wilcoxon test	p=0.022	p=0.000

As a summary of the perception data, our results allow accepting hypotheses H4b: The perceived coverage is higher for the TCT approach, and H4c: The perceived usefulness for guiding is higher for the TCT approach, but regarding the perceived easiness, the null hypothesis cannot be rejected.

4.5 RQ5: Problems Related to ET and TCT

In both years, we asked the subjects to answer an open question in a survey regarding the shortcomings of each technique after they had completed the experiment (see Sections 3.4.3 and

3.5.2 for details of data analysis). In the original study, for both techniques, the largest share of comments was pointing out the limited time for testing rather than the techniques. The problems related to the time restriction used in the first experiment was the reason the experiment was replicated with no time restriction in the subsequent year. Next, we report the main topics of the problems that occurred related to ET and TCT. We include into this analysis the topics that were identified by at least five individual subjects. Note that although the number sounds low, we must keep in mind that typically the answers were between zero to two sentences long and the qualitative answers were distributed over the 44 topics identified in the answers where one answer could contribute to several topics.

In TCT, many answers were related to the level of detail in test cases. Interestingly, we found reports to both directions, i.e., some subjects complained that their test cases had too many details, while others complained that they should have had more details in the test case descriptions. In TCT, many subjects also reported that their test cases were too large, i.e., single test case would try to test too much, and that they should have created more but smaller test cases. On the other hand, some subjects reported that they had too many test cases that were partly overlapping and redundant. They thought they should have had bigger test cases to avoid overlapping sequences of same steps or actions in numerous test cases that only slightly differed from each other.

In ET, the biggest problems were reported in logging of the test progress and the creation of new tests on the fly. Many subjects felt that logging interrupted their testing work and it was also difficult to decide the level of detail the test log should be kept. Creation of tests on the fly was difficult for many subjects, e.g., they thought that figuring out the pair-wise coverage tests while testing was challenging. Furthermore, some subjects also reported that they felt that they were lost during the test session, as they did not know which areas to test next.

In ET, the number of subjects indicating that they would have needed better knowledge of the jEdit text editor was three to four times as high as it was for TCT. With some type preparation, these issues could have been avoided. Actually, many subjects using ET reported that they should have prepared better for the test session, i.e., as they were not forced to create test cases for ET they thought they should have studied the user manual more thoroughly before the testing session.

Finally, regarding both ET and TCT, a very frequently made comment was the lack of coverage of the testing. This comment could be found in both approaches, and 10%–20% of all respondents indicated that this was a shortcoming in their testing. However, as we could not make any distinction between the techniques based on the lack of coverage, we see that there is no difference between the techniques with respect to this viewpoint.

The perceived problems in this experiment were partly similar to the shortcomings of the ET approach identified in industry by Itkonen and Rautiainen (2005). Managing test coverage was identified as the most important shortcoming in the case companies. The logging problem manifested as reproducibility challenges in cases where logging practices were not detailed enough. The challenges in acquiring sufficient application domain knowledge were also identified by Itkonen and Rautiainen.

4.6 The Effects of Context Variables

In this section, we discuss the effects of the most important context variables on the result of this study. Based on the discussion, we identify context dependencies to be studied in the future.

The number of defects in the tested features is an important context variable that affects the generalizability of the results. Our target features had a high number of defects, but it was in line with the actual production quality (we seeded 24 defects and the application had 24 real defects). It is challenging to compare the defect density of the jEdit application with other similar applications because the defect data for the specific version used is not available. However, we were able to estimate the amount of source code related to the search-and-replace functionality that was one of the tested features in the experiment. The number of detected defects in this functionality was 19 and the feature involves 3063 lines of code. This gives us an estimate of roughly 6.2 defects/KLOC (thousand lines of code). This is very similar to reported industry defect densities, e.g., Shah et al. (2012) report defect densities between 0.8 and 6.0 delivered defects/KLOC based on 61 projects. Huang and Boehm (2006) presented on analysis from 161 projects where the nominal quality defect density was 14.2 delivered defects/KLOC and high-quality density was 7.5 delivered defects/KLOC. Additionally, according to (McConnell 2004) Microsoft application division experience 10–20 defects/KLOC during in-house application testing. Based on this limited comparison, it seems that the tested jEdit software was comparable to common references of pre-release or even post-release quality of industrial software products, in terms of defect density. Nevertheless, our results cannot be generalized to situations where the number of defects is very low. This is an interesting avenue for further work, i.e., determining which testing techniques work when there is a high number or a low number of defects. Perhaps, for an industrial practice, a combination approach would work for the best, e.g., Ramasubbu and Balan (2009) found that in a high-maturity (CMM level 5) context, augmenting the plan-driven processes with agile practices provides superior performance outcomes. One approach would be to use ET first to flush out the large majority of the defects and then use more rigorous approaches to uncover the final defects, which proportionally require much greater effort. These hypotheses need to be addressed in the future.

One also needs to take in to account the type of application under testing. We used a text editor that was tested through GUI. Thus, we cannot generalize our results to a very different context, such as, e.g., embedded ABS-breaking software that is tested through technical interfaces. On the other hand, we think that our results can be applied with caution to other applications that work with standard WIMP[13] GUI even when they are not text editors in particular. An interesting further work would be to study ET for applications that do not have GUI or its use in testing non-functional requirements. For example, our personal communication with industry practitioners revealed that even performance testing has exploratory aspects, i.e., system performance thresholds can be determined by exploring and only after that the documented performance tests are created.

[13] Windows-Icons-Menus-Pointer

The knowledge the subjects had for software under testing affects the results. We used jEdit text editor and, undoubtedly, all of our subjects had prior experience in working with some type of text editor, if not jEdit in particular. From the generalizability viewpoint, the prior knowledge is strength because industrial testers can be assumed to have knowledge about the application they are testing. Of course, the application knowledge of the industrial testers can vary dramatically. In (Itkonen et al. 2013), we present a study of highly knowledgeable industrial testers, and perhaps in the context globally distributed testing, there are testers who may possess a rather low level of knowledge of the system and application domain test as suggested in (Grechanik et al. 2010). We think that prior application knowledge might have helped the ET testing more than TCT testing, since with increasing knowledge there is less need for detailed documentation, e.g., Spolsky (2001) states that more knowledgeable cooks need less detailed recipes and we found that deployment experts did not need detailed installation manuals (Mäntylä and Vanhanen 2011). In the end, we think that the fact that our subjects had prior knowledge of the application domain makes the experiment more realistic.

In our studies, we have utilized a condition with no time restriction (replication) and one with time restriction (original study). Our prior work (Mäntylä and Itkonen 2013) indicates that a time restriction decreases testing effectiveness, but increases testing efficiency, and a group of time pressured individuals offer better effectiveness with equal effort. In this paper, we focused on the differences between ET and TCT. We extend (Mäntylä and Itkonen 2013) by finding that both ET and TCT are perceived as more useful for finding defects and achieving higher perceived coverage when more time is available (see Table 12 and Table 21). Regarding defect detection, we found that ET is more effective under time restriction, but TCT more effective when no time restriction is present, although neither of these differences is statistically significant. A similar result can be found in the domain of accounting where a controlled experiment of 179 professional staff auditors showed that the benefits of having structured accounting procedures diminish when time pressure increases (McDaniel 1990). The structured accounting procedures would correspond to the predefined test cases of our study. We state a hypothesis that the ET approach is more effective and applicable in comparison to the TCT approach when severe time restrictions apply. The successful experiences of Våga and Amland (2002) of applying ET in a testing project with severely limited calendar time support this hypothesis. Since time pressure is highly relevant to the software industry that is working under the harsh schedule rush (Glass 2002), future studies should further investigate the effects of time pressure on software testing.

One can question whether our comparison of effectiveness and efficiency based only on defect counts is fair and balanced. For example, organizations may benefit from test cases with other means, e.g., by having increased accountability, repeatability, tracking of coverage, contractual reasons, improved defect reporting, or improved division of labor between testers. Such benefits would typically increase as the size of organizations increases, as the number of regression testing rounds increases, and as the required formal certification and accountability for testing increases. For example, in a complex product line context, an analysis of regression test case executions of manual testing suggests that redundant test executions occur, unless the testing is properly managed and controlled (Engström and Runeson 2013). Thus, we recognize that the TCT approach

may have benefits over the ET approach that are not related to defect detection but to the management of testing. Regarding perceived measures, we found that using the TCT approach leads to higher perceived usefulness to guiding the testing and higher perceived coverage than the ET approach. This finding suggests that the TCT approach could indeed provide the benefits in the test management and accountability of testing. However, such organizational issues were outside of the scope of this experiment and should be addressed in future studies.

4.7 Validity threats

In this section, we discuss the threats to validity of this experiment. We used Wohlin et al. (2000) as a checklist for different possible validity threats. The main validity threats are 1) conclusion validity, 2) internal validity, 3) construct validity, and 4) external validity (Wohlin et al. 2000). Next, we evaluate the validity of this study under each of the four main types of validity threats.

Considering the conclusion validity, there is a threat to the reliability of treatment implementation. We were not able to completely control how the subjects performed the testing activities on a detailed level. The subjects of the experiment used the trained black-box testing techniques for the test case design and we verified this by checking and grading the test cases the students had created before the testing sessions. For the ET sessions, we could not determine if the subjects used the same testing principles that they used for designing the documented test cases or if they explored the functionality in pure ad hoc manner, as this topic was not part of our questionnaire. Controlling this reliably would have required using video recording and think-aloud protocol that could have been used to capture the students' actions and thought processes during the ET sessions. Furthermore, ET literature as summarized in (Itkonen and Rautiainen 2005) specially suggests that ET allows testers to be creative and utilize their knowledge, and thus enforcing a too strict form of control for subjects would have been against what ET suggests. Nevertheless, this might present a threat to validity, as it is possible that subjects applied the instructed testing techniques differently or took varying strategies in the exploratory approach. However, we think that these limitations do not affect the main research question, i.e., what are the differences in the results of testing with test cases in comparison to ET without test cases. On this level, we were able to control that the subjects properly designed and documented test cases for the TCT approach and, on the other hand, we have no reason to suspect that the subjects would have predesigned test cases in secrecy for the ET approach. Another threat to conclusion validity is the reliability of measures. The measures of used effort were based on student subject's self-reported data, which is somewhat unreliable, but we cannot estimate the size or even direction of the potential bias.

The power of the statistical tests affects the conclusion validity. We present the power analysis of the statistical tests of the defect data in **Table 22**. The power of tests for the effectiveness and execution efficiency tests was low, which means that it is possible that there is actually a small difference, but the power of our analysis due to the low number of subjects was not enough to identify it. However, the direction of the effect in the effectiveness was different between the experiments, which indicates that such effect would not be purely caused by the testing approach. The efficiency analysis with the total effort (test design effort included) had high power, in

addition to the statistical significance and large effect size. The false defect count analysis also had high power for the medium-sized, significant effect.

Table 22 Power analysis of the statistical tests (alpha level 0.05)

Test	Significance	Power
Effectiveness test	0.093	0.39
Efficiency test (execution effort)	0.130	0.33
Efficiency (total effort)	0.000	1.00
False defect count	0.007	0.79

An issue that presents a threat to internal validity is maturation in terms of the order of the treatments. We could not control the order of the treatments for the replication and, in practice, most of the subjects reported that they performed the TCT approach first. It is possible that the subjects used more time to get acquainted with the system in general during the TCT approach, even though the feature set was different for each of the approaches. However, it is also possible that students were more tired or had less time to perform the ET task since they had already spent effort on the TCT task. Finally, the subjects performed the test case design before the testing activities, which forced some familiarization with the product already at that phase.

A threat to construct validity is the mono-operation bias, meaning that by using a single case, subject, or treatment will cause the construct be under-represented. We used only one target application for testing. This includes the possible effects of the tested software application and the actual and seeded defects in it. It is obvious that, in software testing, the actual defects present in the tested application affect strongly the results in terms of the number of revealed defects. Similarly, the type of the actual defects might affect the relative effectiveness of different testing approaches. The effect of the testing approach was stronger in one feature set than in the other feature set, which was also the case in the original experiment. It could be possible to calibrate the target artifacts, but at this phase of research, we do not know what factors cause this difference between artifacts. We aimed at balancing the effect of the feature set by randomly assigning the feature set per testing approach for each subject. However, it is still possible that in the context of different systems with different distribution of faults, the results could be different. The relatively high defect density of the target features may benefit one of the studied approaches. We believe that the defect content was typical for the graphical highly interactive application on a low criticality application domain. However, in other domains, the defect densities can be much lower, and the results of this study cannot be directly generalized in such context because there is no knowledge on how the effectiveness of the ET approach is affected by the defect density. The effects of different types of software and defects could not be addressed in the scope of this single replicated experiment. Future studies in different contexts are needed to understand the effects of the ET approach in testing different types of software in the contexts of varying types and densities of actual defect.

Another threat to construct validity is evaluation apprehension, meaning that the students were evaluated based on their performance. The test cases and defect reports they produced were graded based on quality and clarity, and furthermore, points were given for the number of defects they found. The knowledge of being evaluated based on the number of findings might have affected the

number of duplicate and false reports. However, we think that had we not graded the students' performance, some students would still have been motivated while many others could have easily started to care very little of their performance. This would have naturally biased the results as well. Grading of the students may also introduce a threat of students cheating during the experiment. This threat was mitigated by course staff checking all the test logs, test cases, and defects reports, and we found no signs of cheating during this exercise. Furthermore, the students received only a small part of their course grade based on the experiment, thus the risk/reward ratio for cheating in the experiment was poor from the students viewpoint.

Finally, the main threat to the external validity of this experiment is the use of students as subjects introducing possible interaction of subject selection and treatment. It is not obvious how the results of a student experiment can be generalized to the industrial context, but the professional and academic experience of the subjects is presented in Section 3. The subjects' lack of testing experience probably affected the quality of their performance in ET tasks as well as the quality of the test cases they designed. However, the applied training in this study is a typical kind of testing training that software engineering professionals get during their studies before moving to professional work in industry and, therefore, we think the subjects were representative of novice testers with less than two years of experience in the industry.

5 Conclusions

In this paper, we have presented a replicated experimental study investigating whether it is better to perform manual software testing by using predesigned test cases or by following the ET approach where predesigned test cases are not used. Next, we present our conclusions that are summarized in **Table 23** and outline some directions for future research.

The two experiments reported mixed results regarding the absolute defect detection effectiveness, i.e., ET was found slightly more effective in the original experiment, whereas TCT was found slightly more effective in the replication, but the difference in neither of the experiments was statistically significant. Thus, we conclude that there is *no difference in the effectiveness in terms of the number of defects detected* between the ET and TCT approaches in the context of the experiment.

However, in terms of efficiency, *ET is more efficient as it requires less design effort*, i.e., the number of defects detected per hour were 1.49 for the ET and only 0.33 for the TCT approach in the replicated experiment. Similar findings were also made in the original study. This result is caused by the effort subjects used in designing the test cases in the TCT approach. The TCT approach required considerable upfront effort in designing test cases before the test execution, but also the test execution effort was higher in the TCT approach. It seems that our subjects were unable to get benefits from the predesigned test cases that would have led to higher defect detection effectiveness during test execution. In future studies, one should see whether similar results are found with professional testers.

We also found replicable results regarding false positive reports: the false report percentage was 5.12 percentage points higher for the TCT approach than for the ET approach. This finding was identified in both experiments with significant differences between the approaches. In Section 4.2, we discussed this in more detail and hypothesized that the reason could be related to reporting

defects strictly based on the difference between the observed actual result and the expected result in the test case, without considering the actual fault causing the failure symptoms. In future work, the effects of following different types of test documentation should be studied more carefully in order to understand how the documentation affects the tester's behavior and findings. We suggest that a practical test documentation approach could be found between the traditional detailed test cases and free-form ET, supporting both test management and exploratory aspects in the defect detection.

We found initial evidence that the TCT approach could be more effective when there is no time restriction and ET is more effective when working under time restriction, even though the statistical significance of the differences was not high. Additionally, the finding that the used effort correlates with the number of found defects in the TCT approach, but not in the ET approach, supports the hypothesis that the TCT approach requires sufficient amount of time in order to be effective.

When studying problems related to the techniques, we found that both techniques suffered from problems, but they were largely different from each other. In TCT, the most prevalent problem was the challenge with the level of details in the test cases, having either too many or too few details. Other TCT problems were related to having too large or too many (partly overlapping) test cases. In ET, the problems were related to the logging of testing, the repeatability of testing, and creating test cases on the fly. Important research topics related to these challenges are the methods for managing ET work and support for documenting exploratory tests. Furthermore, in ET, a far larger set of students complained about the lack of system knowledge than with TCT where system knowledge was acquired when creating test cases beforehand. These findings suggest that some preparatory activities are required also in the ET approach in order to acquire the required system and domain knowledge, due to the lack of detailed test case documentation activities.

The context of this experiment (see Section 4.6) was manual functional testing performed through GUI. The system under test was a text editor application that is a highly interactive system. The application domain was familiar to the subjects of this experiment. This context is well-suited for the ET approach. Our conclusions apply to similar contexts and we cannot generalize to very different types of testing or target systems. The study focused strictly on defect detection effectiveness and there are also other important benefits in both the TCT and ET approaches that were not studied in this paper.

Table 23 Main conclusions

• Effectiveness: There is no difference between ET and TCT.
• Efficiency: ET is more efficient as it requires less design effort.
• TCT produces significantly more false positives.
• Both ET and TCT have problems, but the problems are different; in TCT, it is the quality of the test cases, and in ET, it is managing the testing activities and reporting.

References

Abran A, Moore JW, Bourque P, et al. (2004) Guide to the Software Engineering Body of Knowledge 2004 Version. IEEE Computer Society, Los Alamitos, CA, USA

Andersson C, Runeson P (2002) Verification and validation in industry - a qualitative survey on the state of practice. Proceedings of International Symposium on Empirical Software Engineering. pp 37–47

Bach J (2004) Exploratory Testing. In: van Veenendaal E (ed) The Testing Practitioner, Second. UTN Publishers, Den Bosch, pp 253–265

Bach J (2003) Exploratory Testing Explained. http://www.satisfice.com/articles/et-article.pdf. Accessed 8 May 2013

Bach J (1999) General Functionality and Stability Test Procedure for Certified for Microsoft Windows Logo. http://www.satisfice.com/tools/procedure.pdf. Accessed 8 May 2013

Bach J (2000) Session-Based Test Management. Software Testing and Quality Engineering 2(6)

Baeza-Yates R, Ribeiro-Neto B (1999) Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc.

Basili VR, Selby RW (1987) Comparing the Effectiveness of Software Testing Strategies. IEEE Transactions on Software Engineering 13:1278–1296.

Beizer B (1990) Software Testing Techniques. Van Nostrand Reinhold, New York

Berner S, Weber R, Keller RK (2005) Observations and Lessons Learned from Automated Testing. Proceedings of International Conference on Software Engineering. pp 571–579

Bolton M (2005) Testing Without a Map. Better Software 7(1)

Carver JC (2010) Towards Reporting Guidelines for Experimental Replications: A Proposal. 1st International Workshop on Replication in Empirical Software Engineering

Cohen DM, Dalal SR, Fredman ML, Patton GC (1997) The AETG system: an approach to testing based on combinatorial design. IEEE Transactions on Software Engineering 23:437–444. doi: 10.1109/32.605761

Copeland L (2004) A Practitioner's Guide to Software Test Design. Artech House Publishers, Boston

Craig RD, Jaskiel SP (2002) Systematic Software Testing. Artech House Publishers, Boston

Crispin L, Gregory J (2009) Agile testing: a practical guide for testers and agile teams. Addison-Wesley, Boston

Ellis PD (2010) The Essential Guide to Effect Sizes: Statistical Power, Meta-Analysis, and the Interpretation of Research Results, 1st ed. Cambridge University Press

Engström E, Runeson P (2010) A qualitative survey of regression testing practices. Proceedings of International Conference on Product-Focused Software Process Improvement. pp 3–16

Engström E, Runeson P (2013) Test overlay in an emerging software product line – An industrial case study. Information and Software Technology 55:581–594. doi: 10.1016/j.infsof.2012.04.009

Field A (2005) Discovering Statistics Using SPSS, 2nd ed. Sage Publications Ltd

Glass RL (2002) Project Retrospectives, and Why They Never Happen. IEEE Software 19:112. doi: 10.1109/MS.2002.1032872

Grechanik M, Jones JA, Orso A, van der Hoek A (2010) Bridging gaps between developers and testers in globally-distributed software development. Proceedings of the FSE/SDP workshop on Future of software engineering research. ACM, New York, NY, USA, pp 149–154

Houdek F, Schwinn T, Ernst D (2002) Defect Detection for Executable Specifications — An Experiment. International Journal of Software Engineering and Knowledge Engineering 12:637–655.

Huang L, Boehm B (2006) How Much Software Quality Investment Is Enough: A Value-Based Approach. IEEE Software 23:88–95. doi: 10.1109/MS.2006.127

Itkonen J (2008) Do test cases really matter? An experiment comparing test case based and exploratory testing. Licentiate Thesis, Helsinki University of Technology

Itkonen J (2013) ET vs. TCT Experiment replication dataset. In: Figshare.com. http://dx.doi.org/10.6084/m9.figshare.689809. Accessed 29 Apr 2013

Itkonen J, Mäntylä MV, Lassenius C (2007) Defect Detection Efficiency: Test Case Based vs. Exploratory Testing. Proceedings of International Symposium on Empirical Software Engineering and Measurement. pp 61–70

Itkonen J, Mäntylä MV, Lassenius C (2009) How do testers do it? An exploratory study on manual testing practices. Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on. pp 494–497

Itkonen J, Mäntylä MV, Lassenius C (2013) The Role of the Tester's Knowledge in Exploratory Software Testing. IEEE Transactions on Software Engineering 39:707–724. doi: 10.1109/TSE.2012.55

Itkonen J, Rautiainen K (2005) Exploratory testing: a multiple case study. Proceedings of International Symposium on Empirical Software Engineering. pp 84–93

Juristo N, Moreno AM (2001) Basics of Software Engineering Experimentation. Kluwer Academic Publishers, Boston

Juristo N, Moreno AM, Vegas S (2004) Reviewing 25 years of Testing Technique Experiments. Empirical Software Engineering 9:7–44.

Juristo N, Vegas S, Solari M, et al. (2012) Comparing the Effectiveness of Equivalence Partitioning, Branch Testing and Code Reading by Stepwise Abstraction Applied by Subjects. Proceedings of Fifth International Conference on Software Testing, Verification and Validation. pp 330 –339

Kamsties E, Lott CM (1995) An empirical evaluation of three defect-detection techniques. In: Schäfer W, Botella P (eds) Proceedings of ESEC '95. Springer Berlin Heidelberg, pp 362–383

Kaner C, Bach J, Pettichord B (2002) Lessons Learned in Software Testing. John Wiley & Sons, Inc., New York

Kaner C, Falk J, Nguyen HQ (1999) Testing Computer Software. John Wiley & Sons, Inc., New York

Kitchenham B (2008) The role of replications in empirical software engineering—a word of warning. Empirical Software Engineering 13:219–221. doi: 10.1007/s10664-008-9061-0

Lyndsay J, van Eeden N (2003) Adventures in Session-Based Testing. http://www.workroom-productions.com/papers/AiSBTv1.2.pdf. Accessed 20 Jun 2012

Martin D, Rooksby J, Rouncefield M, Sommerville I (2007) “Good” Organisational Reasons for “Bad” Software Testing: An Ethnographic Study of Testing in a Small Software Company. Proceedings of International Conference on Software Engineering. pp 602–611

McConnell S (2004) Code complete. Microsoft Press

McDaniel LS (1990) The Effects of Time Pressure and Audit Program Structure on Audit Performance. Journal of Accounting Research 28:267–285. doi: 10.2307/2491150

Mouchawrab S, Briand LC, Labiche Y, Di Penta M (2011) Assessing, Comparing, and Combining State Machine-Based Testing and Structural Testing: A Series of Experiments. IEEE Transactions on Software Engineering 37:161 –187. doi: 10.1109/TSE.2010.32

Myers GJ (1979) The Art of Software Testing. John Wiley & Sons, New York

Myers GJ (1978) A controlled experiment in program testing and code walkthroughs/inspections. Commun ACM 21:760–768. doi: 10.1145/359588.359602

Mäntylä MV, Itkonen J (2013) More testers – The effect of crowd size and time restriction in software testing. Information and Software Technology 55:986–1003. doi: 10.1016/j.infsof.2012.12.004

Mäntylä MV, Vanhanen J (2011) Software Deployment Activities and Challenges - A Case Study of Four Software Product Companies. Proceedings of the 15th European Conference on Software Maintenance and Reengineering. pp 131 –140

Do Nascimento LHO, Machado PDL (2007) An experimental evaluation of approaches to feature testing in the mobile phone applications domain. Proceedings of the Workshop on Domain Specific Approaches to Software Test Automation. pp 27–33

Ng SP, Murnane T, Reed K, et al. (2004) A preliminary survey on software testing practices in Australia. Proceedings of the Australian Software Engineering Conference. pp 116–125

Page A, Johnston K, Rollison B (2008) How We Test Software at Microsoft. Microsoft Press

Pichler J, Ramler R (2008) How to Test the Intangible Properties of Graphical User Interfaces? Proceedings of 1st International Conference on Software Testing, Verification, and Validation. pp 494–497

Rafi DM, Moses KRK, Petersen K, Mantyla MV (2012) Benefits and limitations of automated software testing: Systematic literature review and practitioner survey. 2012 7th International Workshop on Automation of Software Test (AST). pp 36 –42

Ramasubbu N, Balan RK (2009) The impact of process choice in high maturity environments: An empirical analysis. Proceedings of 31st International Conference on Software Engineering. pp 529–539

Runeson P, Andersson C, Thelin T, et al. (2006) What do we know about defect detection methods? IEEE Software 23:82–90. doi: 10.1109/MS.2006.89

Shah SMA, Morisio M, Torchiano M (2012) The impact of process maturity on defect density. Proceedings of International symposium on empirical software engineering and measurement. ACM, New York, NY, USA, pp 315–318

Shoaib L, Nadeem A, Akbar A (2009) An empirical evaluation of the influence of human personality on exploratory software testing. Proceedings of IEEE International Multitopic Conference. pp 1–6

Shull FJ, Carver JC, Vegas S, Juristo N (2008) The role of replications in Empirical Software Engineering. Empirical Software Engineering 13:211–218. doi: 10.1007/s10664-008-9060-1

Spolsky J (2001) Big Macs vs. The Naked Chef. In: Joel on Software. http://www.joelonsoftware.com/articles/fog0000000024.html. Accessed 28 Jun 2012

Tichy WF (2000) Hints for Reviewing Empirical Work in Software Engineering. Empirical Software Engineering 5:309–312. doi: 10.1023/A:1009844119158

Tinkham A, Kaner C (2003a) Exploring Exploratory Testing. Proceedings of the Software Testing Analysis & Review Conference. p 9

Tinkham A, Kaner C (2003b) Learning Styles and Exploratory Testing. Proceedings of the Pacific Northwest Software Quality Conference

Tsang EWK, Kwan K-M (1999) Replication and Theory Development in Organizational Science: A Critical Realist Perspective. Academy of management review 24:759–780.

Tuomikoski J, Tervonen I (2009) Absorbing software testing into the scrum method. Proceedings of 10th International Conference on Product-Focused Software Process Improvement 32 LNBIP:

Vegas S, Juristo N, Moreno A, et al. (2006) Analysis of the influence of communication between researchers on experiment replication. Proceedings of the 2006 ACM/IEEE international symposium on Empirical software engineering. ACM, New York, NY, USA, pp 28–37

Våga J, Amland S (2002) Managing High-Speed Web Testing. In: Meyerhoff D, Laibarra B, van der Pouw Kraan R, Wallet A (eds) Software Quality and Software Testing in Internet Times. Springer-Verlag, Berlin, pp 23–30

Whittaker JA (2009) Exploratory Software Testing: Tips, Tricks, Tours, and Techniques to Guide Test Design. Addison-Wesley Professional

Whittaker JA (2003) How to Break Software A Practical Guide to Testing. Addison Wesley, Boston

Wohlin C, Runeson P, Höst M, et al. (2000) Experimentation in software engineering: an Introduction. Kluwer Academic Publishers, Boston, MA, USA

Wood B, James D (2003) Applying Session-Based Testing to Medical Software. Medical Device & Diagnostic Industry 25:90.

Wood M, Roper M, Brooks A, Miller J (1997) Comparing and combining software defect detection techniques: a replicated empirical study. ACM SIGSOFT Software Engineering Notes 22:262–277. doi: 10.1145/267895.267915

Yatani K Statistics for HCI Research: Mann-Whitney's U test. In: Statistics for HCI Research. http://yatani.jp/HCIstats/MannWhitney. Accessed 28 Jun 2012

Appendix A: Summary of the survey questions

Background

Years of university studies

Study credits

How many years of experience do you have on the following areas?

- Professional software development (any kind of role in development)
- Professional programming (as a developer, programmer or equivalent)
- Professional software testing (as a tester, developer, or equivalent)
- Other kind of experience in software development

Have you got any training on software testing before this course? (Yes or No)

- What kind of training?

Coverage

Assess the coverage of your testing on the following features

4-step ordinal scale: not covered at all – covered superficially – basic functions well covered – covered thoroughly

Exploratory Approach

How easy was the exploratory testing approach to apply in practice?

7-step ordinal scale: (1) difficult … (4) neutral … (7) very easy

How useful was the provided test charter for structuring and guiding your testing?

7-step ordinal scale: (1) hinder … (4) neutral … (7) very useful

How useful was the exploratory testing approach for finding defects?

7-step ordinal scale: (1) hinder … (4) neutral … (7) very useful

What problems or shortcomings did you experience in the exploratory testing approach?

Test-case Based Approach

How easy were your own test cases to execute in practice?

7-step ordinal scale: (1) difficult … (4) neutral … (7) very easy

How useful were your own test cases for structuring and guiding your testing?

7-step ordinal scale: (1) hinder … (4) neutral … (7) very useful

How useful were your own test cases for finding defects?

7-step ordinal scale: (1) hinder … (4) neutral … (7) very useful

What problems or shortcomings did your test cases have?

Which one of the two testing approaches (ET or TCT) gave you a better confidence to the quality of your testing, and why?

Appendix B: Contents of the ET Charter

1. What – tested areas

Select the correct description of tested features for your exploratory testing and remove the other one:

Feature Set B1
Search and replace (User's Guide chapter 5)
- Searching For Text
- Replacing Text
    - Text Replace
- HyperSearch
- Multiple File Search
- The Search Bar
+ Applicable shortcuts in Appendix A

Feature Set B2
Editing source code (User's Guide chapter 6)
(test for one edit mode, e.g. java-mode)
- Tabbing and Indentation
    - Soft Tabs
    - Automatic Indent
- Commenting Out Code
- Bracket Matching
- Folding
    - Collapsing and Expanding Folds
    - Navigating Around With Folds
    - Miscellaneous Folding Commands
    - Narrowing
+ Applicable shortcuts in Appendix A

2. Why – goal and focus

Perform testing from the viewpoint of a typical user and pay attention to following issues:

- Does the function work as described in the user manual?
- Does the function do any things that it should not do?
- From the viewpoint of a typical user, does the function work as the user would expect and want?
- What interactions the function has or might have with another functions, settings, data, or configuration of the application; do these interactions work correctly and as the user would expect and want them to work?

Focus into functionality in your testing. Try to test exceptional cases, invalid as well as valid inputs, things that the user could do wrong, and typical error situations. However, do not test external and environment related (e.g. hardware) errors and exceptions (such as very low memory, broken hard drive, corrupted files, etc.).

3. How – approach

Use the jEdit User's Guide as the specification for the features, and utilize also your own knowledge and experience since the User's Guide is neither comprehensive nor unambiguous. Use the following testing strategies for functional testing.

Domain testing

- equivalence partitioning
- boundary value analysis

Combination testing

- Base choice strategy
- Pair-wise (all-pairs) strategy

4. Exploration log

SESSION START TIME: 2006-mm-dd hh:mm

TESTER: _

VERSION: jEdit 4.2 variant for T-76.5613 exercise

ENVIRONMENT: _

4.1 Task breakdown

DURATION (hh:mm): __:__

TEST DESIGN AND EXECUTION (percent): _%

BUG INVESTIGATION AND REPORTING (percent): _%

SESSION SETUP (percent): _%

4.2 Test Data and Tools

What data files and tools were used in testing?

4.3 Test notes

- Test notes that describe what was done, and how.
- Detailed enough to be able to use in briefing the test session with other persons.
- Detailed enough to be able to reproduce failures.

4.4 Defects

Time stamp, short note, Bugzilla bug ID

4.5 Issues

Any observations, issues, new feature requests and questions that came up during testing but were not reported as bugs.

Appendix C: The target application and features

The target of testing in the both experiments was jEdit open source text editor, version 4.2. with seeded defects in the tested features.
The official version and documentation of the target software can be accessed at:
http://sourceforge.net/projects/jedit/files/jedit/4.2/
The user's guide used as the source documentation for testing can be accessed at:
http://sourceforge.net/projects/jedit/files/jedit/4.2/jedit42manual-a4.pdf/download

The target feature sets used in the experiments were the following:

Feature Set A (Used in the original experiment)

Working with files (User's Guide chapter 4, pp. 11-12, 17)

- Creating new files
- Opening files (excluding GZipped files)
- Saving files
- Closing Files and Exiting jEdit

Editing text (User's Guide chapter 5, 18-23)

- Moving The Caret
- Selecting Text
    - Range Selection
    - Rectangular Selection
    - Multiple Selection
- Inserting and Deleting Text
- Working With Words
    - What's a Word?
- Working With Lines
- Working With Paragraphs
- Wrapping Long Lines
    - Soft Wrap
    - Hard Wrap

And the applicable shortcuts (User's Guide Appendix A, pp. 46-50)

Feature Set B1 (Used in the original and replicated experiment)

Search and replace (User's Guide chapter 5, pp. 26-29)

- Searching For Text
- Replacing Text
    - Text Replace
- HyperSearch
- Multiple File Search
- The Search Bar

And the applicable shortcuts (User's Guide Appendix A, pp. 46-50)

Feature Set B2 (Used in the original and replicated experiment)

Editing source code (User's Guide chapter 6, pp. 30-36)

- Tabbing and Indentation
    - Soft Tabs
    - Automatic Indent
- Commenting Out Code
- Bracket Matching
- Folding
    - Collapsing and Expanding Folds
    - Navigating Around With Folds
    - Miscellaneous Folding Commands
    - Narrowing

And the applicable shortcuts (User's Guide Appendix A, pp. 46-50)