The Effect of Team Exploratory Testing– Experience Report from F-Secure

Paula Raappana, Soili Saukkoriipi
F-Secure Oyj
Helsinki, Finland
{paula.raappana, soili.saukkoriipi}@f-secure.com

Ilkka Tervonen, Mika V. Mäntylä
Department of Information Processing Science
University of Oulu
{ilkka.tervonen, mika.mantyla}@oulu.fi

Abstract— **Practitioners have found exploratory testing (ET) to be cost effective in detecting defects. The team exploratory testing (TET) approach scales exploratory testing to team level. This paper reports the effectiveness of (TET), and the experiences of the participants of TET sessions. The research was carried at F-Secure Corporation, where two projects were investigated. The results show that the TET sessions have good effectiveness and higher efficiency than other testing methods in the company measured in number of defects detected. Furthermore, the TET sessions found more usability defects that other methods. The session participants saw benefits in especially in the joint discussion and learning of the target application. However, with respect to test effectiveness and efficiency we should be cautions as further studies are needed to compensate the limitations of this work.**

Keywords— team exploratory testing, exploratory testing, software testing, experience report, case study, defect analysis, defect classification, qualitative survey;

I. Introduction

According to a recent industrial survey with over 1,500 respondents, the testing teams should expand their skills beyond manual testing and test automation, i.e. the testers are required to test not only functionality of the application, but the customer experience as well [1].

This article introduces a manual testing method called team exploratory testing (TET), which is defined as a way to perform session-based Exploratory Testing (ET) in teams [2]. The existence of TET is motivated by several factors. First, increasing number of people in quality assurance increases the number of the found defects [3]. Second, in the field, a wide variety of people report defects, not only the testers [4]. Yet, it is challenging to get non-tester-domain-experts involved in testing activities because of their busy schedules [4]. Third, with the TET session approach, non-testers can easily contribute in testing in a controlled way and this enables the easy inclusion of domain expert to software testing. The participation to TET sessions requires no preparation or post-work from the participants [2]. Furthermore, non-testers get support during TET sessions that improves the quality of the testing and defect reports yielding to highly useful testing [5].

TET has similarities to software reviews [6] and usability inspections [7]. All of them increase the effectiveness of defect detection by involving several people and all of them benefit from the knowledge transfer between participants. Prior work on TET and team based testing is limited while software reviews and usability inspections have numerous prior works. Our previous work created TET specification in collaboration with a service-focused organization in a big telecommunications company [2]. This paper utilizes those results in the context of F-Secure and seeks to answer two research questions.

RQ1: How does using the team exploratory testing approach affect the testing results?

RQ2: How are TET-sessions experienced by the participants?

In the next section, we review the prior work. In section III, we introduce the context, the research question and the defect classification. The results are described in Section IV and V. In section VI the discussion and limitations of the work are presented. In the last section, we present the conclusions.

II. Related Work

This section summarizes related work by looking at both practitioners reports and academic research articles.

A. Exploratory testing

In theory, ET is almost the opposite of the traditional test-case-based testing (TCBT) method. A tester does not have predefined test cases; instead, she uses her skills and imagination to find defects and weak areas of tested software. Some authors point out that the level of exploration is actually a continuum: *…the question is not when do you do exploratory testing, but rather how exploratory is the testing…*[8].

One beneficial situation in which to use ET is when the next test case to be executed cannot be defined in advance but depends on previous tests and their results [9]. ET is also good when (a) rapid feedback is needed, (b) requirements are vague, (c) the development is in an early phase, (d) the system is unstable, (f) a defect is detected as there is a need to explore the variations, size and scope of that defect to write accurate defect report about it for the developers. ET can also be a good addition to scripted testing at the point when using the scripts no longer leads to new defects being found.

ET is advocated by testing consultants and the topic has gained initial academic interest. Controlled experiments between ET and TCBT [10-13] have shown that the techniques are equally effective. However, ET benefits from superior efficiency as there is less effort spent in pre-design of the test

cases. Thus, it appears that pre-design of manual test cases produces no value with respect to defect detection effectiveness.

A case study in three companies highlighted other benefits of using ET such as experienced based test selection to help with the combinatorial explosion, ability to work with frequently changing specifications, enabling quick feedback to developers and allowing testing from the user's [14]. These findings were supported by another case study [15] that also proposed ET for testing intangible quality attributes in the graphical user interface. Thus, the case studies suggest that ET is beneficial when user's viewpoint is highly important. However, a recent industrial survey found that ET is also used in critical domains [16]. The use of ET in critical domains may be partly explained by the fact that ET allows an efficient use of domain and system knowledge in testing [17].

Shah et al. proposed a hybrid approach to solve the dilemma of choosing between ET and TCBT [18]. They conducted a literature study and practitioner interviews to identify the unique strength of each approach. In their approach, ET and TCBT are executed in parallel and TCBT provides accountability by guaranteeing that all requirements are tested with a documented way while ET offers flexibility, rapid feedback and higher efficiency.

B. Team Exploratory Testing and SBMT

SBTM approach helps track individual testers' ET progress [19]. The basic idea of SBTM is to divide the testing work into time-boxed sessions, and in this way manage the ET. SBTM gives a framework to a tester's ET so that results can be reported in a consistent, accountable way [20].

The first ideas of TET are by Bach [5] who claims that ET done by teams can be extremely powerful. In our past work [2], we present the TET session approach as a method for performing ET in teams, i.e., people with different backgrounds and perspectives come together to test a specific area of a system. Exploratory testing is done with a facilitator and a domain expert supporting the participants. The TET session has similarities to the SBTM approach, i.e., an uninterrupted, reviewable and chartered block of time for focused testing. Differing from Bach's examples [5] of team testing, as in working in pairs or one doing the testing and others watching and commenting, in TET sessions all the participants test together with their own devices.

The TET approach is also similar to defect bash or bug bashing. In a literature review of defect bash, we defined defect bash "as a spatially and temporally co-located testing session performed by a group of people" [21]. The review found that defect bash is often mentioned in the side notes of practitioner literature, like in the testing text books [22, 23], but empirical studies of it are lacking. The main difference between TET and defect bash is that TET approach is more focused and defined, see [2] for details, while defect bash can refer to almost any type of group based testing event.

III. Methodology

This research examines the TET sessions organized at the F-Secure software development organization. The research can be classified as a case study [24], but as two main authors are actually working in the company it might be fair to describe this as an experience report instead.

The following sections describe the context, the research question, the data collection and defect classification in more detail.

A. Case context

The research was carried out during the period March 2012 - April 2013 in F-Secure, a company producing security and data protection products and services both for traditional desktop computers and mobile devices. Figure 1 presents the timeline of the study and the data about the TET sessions. At the beginning, we were aware of the specifications of the sessions. Our previous research at another company [2] provided specification of the TET sessions, i.e., specifications of participant's roles, session parameters and session process. This information was valuable when organizing TET sessions at F-Secure. At the end, we knew more of the benefits of TET sessions, i.e., the amount and type of defects that were found in TET sessions. The participants felt like having also got personal benefits from the sessions as in learning the application better and improving their own testing.

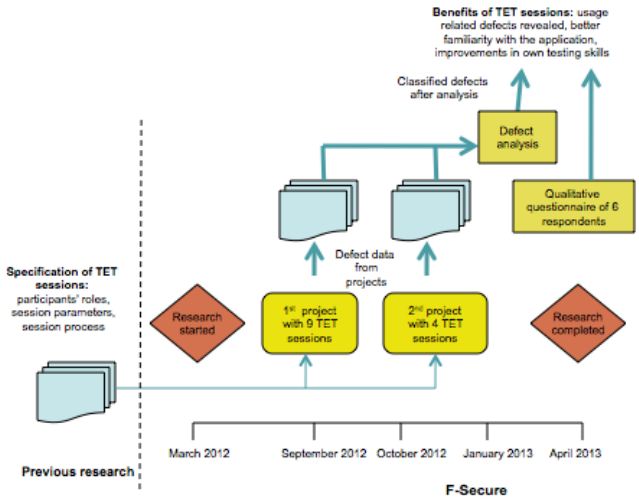

Fig. 1 Research process at F-Secure

The study consisted of two units of analysis, which were projects with targets to renew the user interface and improve the performance of a mobile application, respectively. The projects were carried out subsequently, developing the same application further. The first project was carried out between March 2012 and September 2012; nine TET sessions were carried out during the project. The second project started in October 2012 and lasted until January 2013, with four TET sessions held during this time period.

All the TET sessions were held at the end of development sprint before demoing the achieved functionalities. Reasoning behind the session schedule - the last day of the each iteration - was that at this point other iteration related testing was already done and aim was to find defects leaked from it as quickly as possible and to notice and discuss if any misinterpretations took place during the new feature implementation. The whole

project personnel, that is developers, testers, user experience designer, scrum master, technical product manager, product owner and occasional visitors from the other projects, were invited to participate and they were guided by the facilitator and domain expert. Domain expert could be for example product owner who knows what features are required in the product and what issues actually are defects. Information about the testing focus area i.e. new implementation and its current state including already found defects or other limitations was prepared by the facilitator and projected on the wall. This way the whole project personnel including managers were able to get hands on experience of the new features, project's current quality status, and fewer surprises were left to the next day's demo session.

Table 1 Context of our study described with the checklist of Petersen and Wohlin [25].

Product	Software application to protect smartphones and tablets, and the personal content on them, against all threats.
Process	The process was scrum with three week sprints. Each sprint produces demoable application.
Practices, Tools, Techniques	During the sprints the following testing took place: grooming, reviews, unit testing, integration testing, individual exploratory testing in SBTM style, automated smoke, regression and performance testing, re-testing, and in the end of the projects localization testing and beta testing inside the company. In addition to these different testing tools were applied and benchmarking was done time to time and during each sprint test planning and reporting took place. In the end of the each sprint TET session was organized.
People	Projects had technical product manager, product owner, scrum master, 1-2 quality engineers, 3-5 developers, and UX designer. Some of these people were working in other projects also and one of the developers acted also as an architect. Projects made close collaboration with the product's maintenance team.
Organization	F-Secure makes software security products and services to global markets. Company has long history with software process improvement. Organizational unit is team making the particular product.
Market	Product for global market

B. Data: Defect classification and Questionare

Following data collection method levels introduced by Runeson and Höst [24], the data collection in this research consisted of level-two and -three methods. In level two, the researcher collects the data, which in this case applies to one of the researchers acting as a facilitator in studied projects' TET sessions, and through that role collecting and reporting forward the findings by session participants. In level three, researcher independently analyzes the available data. In the research in question, another researcher performed the data analysis by collecting the recorded defects of the studied projects from the defect tracking tool without having been involved in the projects' TET sessions or in other testing work of the project.

Findings from all 13 TET sessions were included in the defect analysis as well as defects found by other testing means (called non-TET session defects in the research). The non-TET session defects include defects found by other than TET session testing approaches; for example, defects found by reviews, end-users or testing tools, see Table 1 for other practices.

Both the TET session and non-TET session defects were reported to a project tracking tool during the projects by the defect finders or TET session facilitators. We then carried out defect analysis by collecting the defects of the projects from the tracking tool. The defects were analyzed and labeled in defined defect type categories. In addition to categorizing defects in different types, we also analyzed the severity levels of the defects. The severity levels were set by defect reporters according to projects' conventions. Severities were set from the viewpoint on what the severity of the defect is for the customer. The severities are listed in Table 2.

Table 2 Defects' severity levels

Severity level	Definition
1	Show stopper / Unusable for over 10% of customers
2	Urgent / Unusable for some customers
3	High / Impairment, bad workaround
4	Medium / Workaround exists
5	Low / Inconvenience or annoyance
6	Small / Enhancement

The defect categorization used in the research for defect type analysis is based on Itkonen et al.'s [10] technical defect type categorization modified slightly to match the product and defects typically found from it, see Table 3. Other defect classifications such as Humphrey's [26] defect type standard and Chillarege et al.'s [27] Orthogonal Defect Classification (ODC) were studied when selecting the suitable classification. Both of these categorizations were seen as too code-based to be used when looking at the defects from the point of view of testing the product itself and not reviewing and checking the code.

Table 3 Defect type categorization used in the defect analysis.

Defect type category	Definition
Documentation and guidance	Issues with documentation and user guidance in the software; for example, incorrect information in help texts or shown dialogs, missing help where such would be needed or unclear or misleading guiding texts.
UI	Issues with user interface of the software; may cause difficulties or inconvenience to use the software.
Inconsistency	Issues with inconsistency; for example, in functions, graphics, texts or dialogs of the software.
Functionality	Issues with functionality of the software; for example, a feature does not work as end-user could assume, functionality is illogical or simply wrong.
Performance and reliability	Issues with performance and reliability; for example, slow performance speed, using a lot of device memory, software getting stuck or data getting lost during upgrade process.
Technical	Issues with technical details of the software, such as code-level issues which are not actually shown to end-user clearly or issues requiring actions with backend systems unseen by end-user.
Usability	Issues with the ease, comfort and clarity of how the software and its functions need to be used, make it difficult, complicated or inconvenient to use the software or its feature for the end-user.
Localization	Issues with software localization; for example, mistakes in used language, used words causing problems in the UI (need for revised localization) or missing localization.
Enhancement	Issues that could be done in a better way even though the current solution also works okay.

Additionally, to collect the qualitative data of the TET sessions a questionnaire was conducted in April 2013. The questionnaire was sent to the TET session participants by email. There were 14 participants for whom the questionnaire was sent and 6 answers were received and used in the analysis. The questionnaire is presented in online appendix[1].

Section V presents TET-session experiences that we collected with a qualitative questionnaire while the next session presents the results of the defect classification.

IV. Results – RQ1 Testing results

A. TET session effectiveness and efficiency

The overall number of found defects in TET sessions during the two projects was 115, see Table 4. In addition to this, 13 false defects and 3 duplicate defects were found. Other testing approaches resulted in 427 defects and 25 false defects and 3 duplicate defects being found. In Project 1 TET sessions, a total of 92 defects and ten false defects were found and two defects where duplicates, while other testing approaches produced 281 defects, 19 false defects and three duplicates. In Project 2, there were 23 defects, three false defects and one duplicate found in TET sessions, while 146 defects, six false defects and no duplicates were found by other testing approaches. To summarize, TET sessions revealed 21% of the defects while 79% were found with other testing methods.

To compare efficiency between TET sessions and other testing work we need effort data. Unfortunately, the company time tracking system did not record testing effort, thus, we are forced to use the estimates provided by the company. For TET sessions, the estimates are as follows. Each TET session was thirty minutes long. The TET session facilitator estimated it took about one hour to prepare a test session and four to five hours to finalize the session afterwards. The projects included a total of 90 participations (by 14 people) in thirteen sessions, and each session had five to ten participants. Taking the thirty-minutes times 90 participations and adding the preparation and finalizing times, it took 116.5 hours (13 sessions * 1 h preparation time + 13 sessions * 4.5 h completion time + 90 participants * 0.5 hours session length) to find and report the defects through TET sessions. This gives the efficiency of 0.987 found defects per hour (115 TET defects / 116.5 hours TET time).

For other testing work, we know that the projects had 1.5 testers working for the projects see also Table 1 for list of other used testing practices. The company calculates six hours efficient working time per a work day (this exclude time spent on overhead). This gives us effort spent in other type of testing (projects' data minus TET session effort spent by the testers): ((target projects' combined length, 7 + 4 months - 1.25 months for vacations) * 21 working days per month * 6 efficient working hours per day * 1.5 testers) - (13 sessions * 1 h preparation time + 13 sessions * 4.5 completion time + 13 sessions * 1.5 testers * 0.5 hours session length)). This gives us the efficiency of 0.242 defects per hour (427 non-TET defects / 1761.5 non-TET testing hours).

[1] http://mikamantyla.eu/The_Effect_of_TET_Appendix.pdf

This number is a rough estimate since it assumes that all project defects were found by the testers, and all testers' time was used in defect hunting. Of course in reality not all of the testers' time is spent in testing, but at the same time also software developers and other people performed testing in the project.

The results of other testing may also be skewed by two practices. First, the software developers don't always record the defects they found, but the bugs were patched without defect reporting. Second, the testers tend to use collected reports of defects with localization testing. Localization defects were reported by language such that each language specific report may include several different items. When an application is localized on 27 different languages, and we have 5 items (a rough estimate) by reported language in two projects, then we report maximum 54 defects instead of 270 defects. Given that in reality localization defects were not found from every language but tended to mount up a more precise estimation would be 25 defect reports instead of 125.

Comparison shows that TET sessions were more efficient than other testing (0.987 vs. 0.242 defects per hour), see Table 4. The difference is so large that even if the company's estimates for the time spent in other testing would be reduced by as much as 50% and we take the defect recording practices (discussed above) into account, still the TET session would be more effective than other type of testing. Furthermore, it should be kept in mind that the other testing was done during the whole development time, whereas TET sessions were only held in the end of each sprint so the majority of defects had and should have been found before the session. Therefore, we think this data shows that TET sessions were beneficial for the company.

In Table 4, we show data from two other industrial papers, to provide a reference point for our efficiency numbers.

Table 4 Testing effectiveness and efficiency

Source	F-Secure - TET sessions	F-Secure Other testing	ET test sessions Case A/B [14]	Sub-system and System testing in Testing phase T1/T2/T3/T4 [28]
Defects found	115	427	169 / 34	20/12/24/0
Effective testing hours	116.5	1761.5	36 / 4	32/570/3150/160
Efficiency (defects per hour)	0.99	0.24	4.8 / 8.7	0.63/0.021/0.0076/0.0

Itkonen and Rautiainen [14] studied ET sessions in two companies and report a defect detection efficiency of 4.8 and 8.7 defects per hour which is extremely high. Their study does not reveal what testing had been performed before the ET sessions. However, in the paper they mention that ET was used to give rapid feedback to the developers, thus, we can think that the ET sessions were one of the very first testing methods applied, which would explain the high efficiencies. Berling and Thelin [28] provided data of testing from a telecom company that we used to calculate the efficiencies of testing. The testing work was divided into different test phases T1-T4 where T1 is performed first and T4 is the last test phase. Their data shows that defect detection efficiencies decrease in orders of magni-

tude as the product progresses through testing phases. In the first sub-system testing T1 the efficiency is 0.63, in the next second sub-system level testing T2 the efficiency is 0.021, in system testing T3 efficiency is 0.0076 and in T4 final verification testing no defects were found. Given that in our case other testing was performed before TET sessions one would expect TET to produce far lower efficiencies than other testing. Since this was not the case we assume that TET sessions are effective, but also that other type of testing should have been more effective.

B. TET session defect severities

The defect analysis studied whether there are differences in defects found in TET sessions compared with defects found by other testing methods. The number of defects in this presentation are shown as percentage values. Table 5 shows all the TET session defects classified by their severity level from both studied projects. The table shows that 54% the defects found in TET sessions are of medium-severity whereas for non-TET session defects the share is 43%. High-severity defects had 21% share in TET session and 32% in other testing. For low severity defects, the shares were 21% and 18% in TET and non-TET respectively. With respect to showstopper and urgent defects TET session had 3.5% share whereas non-TET sessions 6% share. To summarize the table shows that non-TET session found more defects that were severe.

These findings can result from the situation that the TET sessions were pre-demo sessions, meaning that they were held when the iteration's development work was already ending. Thus the more serious defects with higher impact on the customers were likely to be found before the TET sessions by other testing means. On the other hand the limited time of TET session could also explain the lower share of Show stopper, Urgent and High defects in TET session. Given the situation in our case it is impossible to come up with a definite cause-effect relationship on this matter.

Table 5 Defects found in TET sessions classified by defect severities

Defect type	TET session	Other testing
Show stopper	0.9%	1.6%
Urgent	2.6%	4.4%
High	20.9%	31.9%
Medium	53.9%	42.9%
Low	20.9%	18.3%
Enhancement	0.0%	0.5%
Undefined	0.9%	0.5%

C. TET session defect categories.

Table 6 shows defects found in TET sessions and non-TET session by defect type. The table shows that there was a notable amount of user interface (UI) and usability related defects found. UI defects have a 28 percent share of all TET session defects and usability defects a 24 percent share. The numbers for non-TET were 18% and 14% respectively. These two together include over half of the found defects in TET sessions with a 52 percent share compared with 32% in non-TET testing. The third-biggest portion of TET session defects are functionality defects with a 22 percent share. For non-TET Functionality defects have the largest share with 40%. Other categories' defects have smaller shares of total findings. Notable differences in them are that in Performance and reliability non-TET has 8% defect share whereas TET has only 3%, and that TET-session has 9% share of inconsistencies while non-TET has only 2%.

The defect figures can also be looked at from the point of view of grouping the types into more end-user's application usage-related defects, and on the other hand, on the application's technical and functional related defects. Grouped like - this, enhancement, inconsistency, localization, documentation and guidance, usability and UI defect types form one group of "ease and comfort of usage" defects. Then functionality, performance and reliability and technical defects form a second group called "technical and function" defects. Such grouping reveals that the TET sessions provide more ease and comfort of usage-related defects (75%) than technical and function-related defects (25%). For Non-TET session the shares are 47% and 53% for ease and comfort of usage and technical and function defects respectively. Thus, the TET-sessions with a variety of participants could be more likely to catch usage defects that the developers may have gotten dazzled of.

V. RESULTS - RQ2: EXPERIENCES OF PARTICIPANTS

A qualitative questionnaire was used to find answers to the second research question - *How are TET-sessions experienced by the participants?* The aim was to gather TET session participants' opinions and views about TET sessions' usefulness or disadvantages and to learn if the participants find the usage of team in exploratory testing a good way of testing or not. Also, we wanted to know if the participants thought they personally gained something from participating in the TET sessions. In following subchapters, the answers to the questionnaire are presented and analyzed.

A. Recipients' background

The first question was about the role of the respondent in the software development process. The dispersion of the respondents was quite wide from this point of view, as out of six respondents there were two software engineers, one quality

Table 6 Defect categories and severities

Severity	All		High		Medium		Low	
Defect categories	**TET**	**non-TET**	**TET**	**non-TET**	**TET**	**non-TET**	**TET**	**non-TET**
Enhancement	4%	1%	4%	0%	2%	1%	12%	1%
Inconsistency	9%	2%	0%	1%	15%	1%	4%	5%
Localization	4%	7%	4%	0%	5%	8%	4%	20%
Documentation and guidance	5%	5%	4%	3%	6%	5%	4%	9%
Usability	24%	14%	17%	13%	2 %	16%	21%	16%
UI	28%	18%	8%	7%	29%	25%	46%	24%
Functionality	22%	40%	54%	56%	14%	34%	8%	18%
Performance and reliability	3%	8%	8%	13%	2%	5%	0%	3%
Technical	1%	6%	0%	7%	2%	5%	0%	4%

engineer, one software architect, one scrum master and one product owner who answered the questionnaire. Thus multiple kinds of perspectives were included in the answers.

The second question was also a background question asking the respondent's experience of testing – what kind of testing he or she has done in her work and for how long. Four respondents gave a presumption of their experience in testing in years, the shortest being three years and longest thirteen years. Thus out of these respondents it can be said that they are experienced testers in general. Two respondents did not mention their experience in years.

As all of the respondents were participants of TET sessions they all have at least that experience from exploratory testing. In addition to this, five out of six respondents mentioned having done exploratory testing in their work which is likely to refer in the individually done ET. Three of the respondents mentioned having done developer testing, two mentioned having done automated testing and one also mentioned performing manual and semi-automated test case based testing. Two respondents also added having done multiple kinds of testing during their many years of working life and thus being unable to list all of them, one respondent commenting:

- *I've probably done all kinds of testing during the last 13 years of my career - can't really make a list of them.*
- *Nowadays I usually write and execute unit- and integration tests during development as well as exploratory and ad-hoc testing.*

When it comes to the TET sessions, the answers to the third question showed that the respondents were also experienced TET session participants. Three of the respondents wrote having participated in over five TET sessions. One of the respondents also mentioned participated in more than 15 TET sessions. Only one participant wrote having taken part in only one TET session and one in three sessions.

B. TET sessions compared to individually done testing

The fourth and the fifth questions asked the respondent to elaborate about the possible benefits and the possible disadvantages correspondingly when comparing the TET sessions with individually done testing. All the respondents gave some answer to both questions.

Four respondents raised as a benefit the group discussion in some words. For example phrases like "*immediate discussions*", "*instant feedback*", "*I can ask*", "*colleagues collaborating in the same room"* and "*communality*" were used to describe the TET sessions Below are two citations from the answers addressing this item as a benefit of TET sessions.

- *Yes. In my opinion the benefits comes from the immediate discussions about the issues found during the sessions. From developer point of view it helps a lot finding the root-causes of the problem faster. Also, I have a feeling that occasionally the group synergy in the session helps finding more bugs and issues faster than during individually executed testing.*
- *At least meeting's communality is a benefit when comparing to individual testing. I mean that you tend to think aloud and can discuss about found issues with other participants. This is an advantage for example when other participant can immediately try if some particular issue can be repeated also by her, or you can discuss with others what could cause the issue or is it even an issue worth investigating.*

As other mentioned benefits of TET sessions compared with individually done testing was the sessions giving developers a broader look at the application and making it possible to test the application between the developments of new features. It was also mentioned that it is a good side of TET session that the process is such that the results will be recorded immediately after the session. Another mentioned benefit was the testers getting "*an insight from non-testers on how applications are used by other people*".

One of the respondents did not find the TET session beneficial compared with individually done testing but said that a person can do exactly the same testing without the session as it still is one person doing the testing.

As disadvantages of TET sessions there was no consensus as different people found different items as downsides of the sessions. Two of the respondents did not see any disadvantages in TET sessions compared with individually done testing.

One of the respondents raised the possibly vague descriptions of found issues as a disadvantage. The respondent explained that as the testing is done in the session in an ad-hoc manner and within the participants there are people who do not have testing experience, the issues are sometimes reported in an unclear and non-descriptive manner, which does not help to fix the issue or even to decide if the issue is functional or for example usability defect or defect at all. An example was provided:

- *For example at early TET sessions we've had issues reported from TET session saying nothing more than: "Home view does not look good, it should look better".*

Another respondent mentioned as a disadvantage of the sessions the fact that someone has to arrange the session and afterwards check the feedback gained from it, meaning there is extra work caused by the TET sessions. One respondent mentioned predefined test cases that should be tested in the sessions are not a good idea; the respondent wrote that this kind of testing is easier to do alone, with good concentration. The respondent also added that in that kind of testing it can be a disadvantage if the participants include people who do not have prior experience from testing.

The joint discussions were raised as commonly seen benefit in TET session but when asking for the possible disadvantages the discussions were also raised as a disruptive thing in the sessions. One respondent wrote that as people are telling each other about the defects they have found or other ways discussing in the session, the actual testing slows down and concentration is disrupted.

C. TET sessions' suitability in different testing areas

The next two questions – the sixth and the seventh – were about the testing areas where TET sessions especially fit, or in the other hand lack of fit, in the respondent's point of view. Regarding the testing areas where the TET session is seen as especially fitting testing approach, usability testing was mentioned by two respondents and one respondent phrased the answer as customer-centered exploratory testing where the usability testing also fits in.

One of the respondents said that TET sessions fit especially into testing full features. Another TET session participant saw the sessions being good in finding regression related defects and explained this view as cited below.

- *Regression related bugs are often found in these sessions. The exploratory testing flow is more random in the nature so it is not uncommon to make a discovery that would not have been made in actual structured regression testing.*

Regarding the non suitable testing areas for TET sessions an alignment was found in the answers: it is not a suitable way of performing the testing of functions that require a lot of steps to carry out, testing timely speaking long running functions or performing testing that requires backend, device side or remote service configurations. Each of these issues was mentioned in two or three answers.

One respondent did not have any comments on the question of testing areas where TET sessions are not suitable for and another wrote that no areas like this come to mind, following is a citation from that answer.

- *Can't really elaborate, just a feeling, but in my opinion TET sessions can be held at any time of the lifecycle of an application. It's always good to have testing, rather more than less. Also TET sessions don't "cost" that much resources compared to benefits gained from it.*

D. TET sessions' successfulness

The eighth question addressed if the respondents see the TET sessions they have participated in as having been successful. In a summary, all the answers were saying the sessions had indeed been successful. Three of the answers were purely commenting that the sessions were successful, one writing that as far as remembered right, there had not been any session where no defects were found and another phrasing that "multiple bugs were found during a short time period". The third answers simply said that the sessions had been "very helpful".

The other three answers included some hesitation though firstly finding the sessions successful. One respondent added that whereas some had been helpful, some had not been for example due to vague defect descriptions. Another respondent commented that at some sessions it would have been good to have a clearer list of items into which concentrate on as the application had not had clearly showing changes compared with an earlier session. One answer said the sessions had been helpful, but in respondent's opinion the found defects would have been discovered by other testing means as well.

E. TET sessions' benefits for individual participant

With the ninth question it was asked whether the respondent thinks he or she has got some personal benefits from taking part in the TET sessions. One respondent did not give any comments to the question, whereas the other five respondents wrote they had got benefits.

Two answers included comments about getting to learn the application better. This was due to taking time through the TET sessions in trying all the application's functions – or as a respondent put it "*I know the products better after having taken the time to try every knob and button, trying to find bugs*" - and due to testing the application with different devices.

In addition, in a product owner point of view comments about how to better define features helps in avoiding misunderstandings in the future development work as the definitions will become clearer. Personal benefits included also the fast introduction to the new features, and getting new test ideas, which was commented in two respondents' answers. They wrote that to see how other people use the application and to discuss the application helps thinking up new and better ways of testing both the application and the code inside.

F. Free word on using teams in (exploratory) testing

The tenth, and the last, point on the questionnaire was giving the respondents a free word on commenting on the usage of teams in testing, or more specific in exploratory testing. Three of the respondents did not give any further comments. The other three answers are fully cited below.

- *Teams have more eyes and fingers than one tester who probably does thing in a certain way every time and thus misses some bugs. During TET session's people have asked a lot of questions about the application and its functions and specifications that would otherwise have been never addressed.*
- *The only comment I have is that in my opinion this [TET] should be a mandatory practice to a team and especially to those who don't normally exercise testing in their daily job*
- *Considering the time it has taken to do this kind of testing (half an hour every sprint), my understanding is that we have been able to identify a good amount of bugs that might even have passed to production without these sessions.*

VI. Discussion

A. RQ1

The research question (RQ1) to be answered by the executed defect analysis was, "*How does using the team exploratory testing approach affect the testing results?"* With respect effectiveness we found that TET session found little over 20% of all defects. This is a significant contribution to the testing effort of the analyzed projects. TET sessions were also more efficient than other type of testing (0.987 vs. 0.242 defect per testing hour). However, this needs to be confirmed in the future studies due to the limitations in effort data.

The defects were analyzed by severity and type. With respect to severities no large differences were found between

TET and non-TET methods. In TET sessions, fewer high-level and than by other testing means. The differences, however, most likely depend on the fact that the TET sessions were held at the end of development sprints. This means that most high-level defects are already found and fixed during development sprints and therefore TET sessions held at the end of the sprints find fewer of such defects.

The study showed clear differences between what types of defects are found in TET sessions and what are discovered through other testing. Most TET session defects were of the user interface and usability types. This idea is also supported by recent paper suggesting that ET testers could be supported by HCI techniques [29]. In other testing methods, most of the defects were of the functionality type. Thus, the answer to RQ1 is as follows: TET affects the results of testing by providing good effectiveness and high efficiency, and reveals defects that relate to end-user's usage of the tested application. While also functionality and technical related defects were found, usage-related defects formed the majority of the defects.

It should be remembered that the two approaches are not thought here as competitors or as each other's substitutes but as complementing ways of testing. In TET sessions the only used testing technique was exploratory testing whereas other testing approach included also other techniques, see Table 1 for details, and also localization testing was done separately during the other testing. Other testing also was done during the whole development time, whereas TET sessions were only held in the end of each sprint so majority of defects should already have been found before the session. These factors lead to the conclusion that it is natural that the other testing methods at some level find defects in more wide variation, more defects from more various categories and TET sessions then complement the already done testing.

B. RQ2

The RQ2 was *"How are TET-sessions experienced by the participants?"* was studied by conducting the open ended questionnaire for the TET session participants. Summarized from the questionnaire results the participants saw the sessions as a good way of discussing the application in hand to understand its functionality and to know what noticed issues are actually defects. It was pointed out that it helps the developer to understand the causes of problems, to see how the application is used, and to discuss about the steps leading into the defect taking place. It was seen that TET sessions are especially suitable for usability testing. Vice versa, the approach was seen as not a good way to test functions that are time taking or require lots of test steps. The vague defect descriptions from non-experienced testers, time required from the session facilitator, and unclear definitions of what should be tested were seen as shortcomings.

All of the respondents of the questionnaire saw the session having been successful with slight hesitation caused by the above mentioned shortcomings. The respondents felt having got personal benefits from the sessions as in learning the application better and improving their own testing based on the comments and ideas got from the session. Overall the respondents saw that the usage of teams in testing is good because a team has more resources than only one individual tester, the required resources are low versus to the seen benefits, and the discussions are not likely to happen anywhere else than in the sessions. Thus, the found issues could have been left unaddressed without the sessions.

Whilst the overall tone of responses was very positive to the direction that the respondents saw the TET sessions very useful and good way of testing, there were the downsides of the sessions also mentioned and couple of answers also showed that not everyone finds the joint testing and discussion as useful but may find it disruptive also. Additionally, the organization of the session is important, it should have a mission what is tested and the mission should be made clear for each participant. It should also be taken care of that the defects are explained and reported in a descriptive enough way that enables the reproduction of the defect and the further investigation of it.

VII. Limitations of the Study

This section summarizes the limitations of the research. These limitations are discussed from the viewpoints of construct, internal, external validity, and reliability [30].

A. Construct validity

This aspect of validity reflects to what extent the operational measures that are studied really represent what the researcher have in mind and what is investigated according to the research questions [24]. We studied in our research the effectiveness and efficiency with a number of defects found measure. It is a commonly used effectiveness measure, thus, the threat to construct validity is small in that respect. For efficiency, we have a larger construct validity threat as the number of hours used in other type of testing was based on company estimates rather than actual effort data. Unfortunately, the company had no effort data available. Additionally, we studied the type of defects found in TET sessions and in other testing (conventional and ET testing) approaches. This measure gave us more detailed view of the effect of TET sessions than pure defects counts alone, thus, this lessens the threat to construct validity with respect to the effectiveness and efficiency of TET.

B. Internal validity

This aspect of validity is of concern when causal relations are examined [24]. We studied the causal relationship between the TET sessions and the effectiveness and efficiency of testing. However, it is possible that a latent third factor might have caused the good effectiveness and high efficiency of TET session. For example, the fact that all TET sessions were at the end of the sprint might have motivated the people to perform better testing on the eve of the product demonstration. Thus, the TET session might have no effect at all and the results could be caused by the upcoming product demonstration. Unfortunately, we had no way of investigating this matter further.

Furthermore, in software testing we have several other factors, which may cause the success of testing. These factors include e.g. expertise of participants, support and guidelines provided for testing, reserved time for a testing session, and available tools for testing. For example, time pressure has been shown to increase the efficiency in software engineering experiments and the limited time and possible time pressure of TET sessions might have affected the efficiency of TET [3].

C. External validity

This aspect of validity is concerned with to what extent it is possible to generalize the findings. In case studies, there is no population where a statistically representative sample would been drawn from. Instead, in case studies generalization happens to similar context only [24]. In Table 1 we have outlined our context variables following the guideline by [25]. In more detail, case studies rely on analytical generalization where the researchers try to identify the most relevant context variables to allow generalization to other cases that share the same characteristics [24]. Listing out all context variables in case studies is not needed nor desired as it hides the researchers' interpretations on what is relevant and simply leads to defining a number of context combinations that may exceed the number of atoms in the universe [31].

Next, by defining the most relevant context variables, we allow generalization and provide support for companies when they consider how to apply TET sessions to company's development process. We briefly discussed the main context variables in Table 1. Next, we discuss six context variables that we think can explain the results, and provide analytical reasoning as to why they are likely to do so. The six important context variables in our cases were (1) scrum way of working, (2) quality as every one's responsibility, (3) each sprint resulted in a working demo, (4) quality engineer's working in the team, (5) facilitator's experience and (6) testing process.

Scrum way of working: Flexible way of working that enabled fast reaction to the new requirements and change requests. Not to compromise on quality multiple testing approaches were applied starting from the grooming user stories and ending to beta testing.

Quality as every one's responsibility: Everyone was considered to be responsible of the end result and its quality. Needed information was got and decisions were made fast. People from all positions participated TET sessions and did at least some kind of testing also as part of their daily job. It is possible that TET sessions encouraged usage of the target application also outside of the sessions since during the sessions it was able to learn its usage and possible restrictions, get personal guidance and give feedback.

Each sprint resulted a working demo: Project members and all stakeholders were invited to the demo session. Based on the demo it was decided if the sprint was successful or not. TET sessions were held before each demo to catch all the defects not found during the user story testing. Also TET sessions ensured that all project members were able to use and try the very latest version of the target application and this way understand what new user stories mean in reality and what is the quality status.

Quality engineer's working in the team: Dedicated quality engineers were working in the team even though the quality was every one's responsibility. Quality engineers made sure new user stories were testable, all the needed testing tasks were done - also those not execute by themselves, quality status including risks was understood and reported to all stakeholders and needed correcting steps were taken if needed.

Facilitator's experience: Successful TET session requires experience and expertise from the facilitator and domain specialist who support participants and improve the process. For example during the study sessions were improved by projecting session focus area and its known issues on the wall. Also facilitator gave more attention to participants' difficulties to repeat the found issues and report them properly. Domain specialist enabled fruitful discussion and feedback giving.

Testing process: Testing was included to the requirement life cycles from the very beginning (grooming) until the very end (beta testing). All the steps didn't aim to find defects but prevent them and all were meant to complement each other not to waste time or resources. TET sessions completed user story testing, made sure all project members had hands on knowledge and experience of the target application's current situation and enhanced information sharing.

D. Reliability

This aspect is concerned with to what extent the data and the analysis are dependent on the specific researchers. Hypothetically, if another researcher later on conducted the same study, the result should be the same [24]. In this research, there was only one person doing the defect categorization research, which may influence categorization results, since it is a single person's view. Some defects in studied projects' applications inevitably were such that could fit into two or even into more categories; thus, another researcher could decide otherwise while scaling the decision of which category to select for such a defect. We tried to minimize the effect of this limitation by going through the classification three times and by creating a clear definition of categories and problematic situations referring to it. Although we studied the types and severity of defects only in one company, we have organized TET sessions in two companies. Our previous research in a service-focused organization at a large telecommunications company provided valuable guideline material for TET session organization purposes and would help redoing the study in other companies [2].

VIII. CONCLUSION

This paper reports how the use of a TET approach affects the results in the software testing. TET session had high effectiveness as they found 21% of all defects even though they were the last testing methods performed. With respect to efficiency, we found that TET sessions produced higher efficiency (0.99 vs. 0.24 defect per hour) than other testing methods on average. Unfortunately, our efficiency comparison was hampered by limited effort data collection as the company did not track hours spent on different testing methods. Comparison of the defect type detected revealed that the TET sessions are more likely to detect usability and user interface-related defects while other testing means detect more functionality related defects. These findings can result from the situation that the TET sessions were organized as pre-demo sessions when the iteration's development work was already ending. Finally, the participants of the TET sessions experienced them positively and highlighted the benefits with respect to learning about application under test, the instant investigation of defect reproducibility by other participants, and the ability to immediately reflect upon the findings with other participants.

As is typical in a case study or an experience report, the results are not generalized for wider use but might be applicable to similar projects. The results imply that the TET sessions can be a useful form of testing when complementing other testing. In addition to providing the concrete results of found defects, the sessions seem to bring new perspective to application development, and better understanding about the application for the participants. Defects related to the end-users usage of the application were found in the analyzed TET sessions, and thus the results of the sessions were valuable to the target organization since the usability is likely to affect the end-users' satisfaction with the application.

Further research is needed to study the effects of team exploratory testing when testing other types of projects or in other organizations. Another idea would be studying the differences between TET session results and the results developed through other testing by having one project performing only TET session testing and one only individual testing. This could be a more precise way to see what effects are due to using a team exploratory testing and what factors are purely consequences of how the TET sessions are scheduled within the other testing.

REFERENCES

[1] Buenen,M. and Walgude,A., "World Quality Report 2015–2016", *Capgemini, Sogeti and HP,* 2015,

[2] Saukkoriipi,S. and Tervonen,I., "Team exploratory testing sessions", *ISRN Software Engineering,* vol. 2012, 2012,

[3] Mäntylä,M.V. and Itkonen,J., "More testers – The effect of crowd size and time restriction in software testing", *Information and Software Technology,* vol. 55, no. 6, 2013, pp. 986-1003.

[4] Mäntylä,M.V., Itkonen,J. and Iivonen,J., "Who Tested My Software? Testing as an Organizationally Cross-Cutting Activity", *Software Quality Journal,* vol. 20, no. 1, 2012, pp. 145-172.

[5] J. Bach. , "Exploratory Testing Explained," 2003, Accessed 2015 6/1 http://www.satisfice.com/articles/et-article.pdf

[6] Wiegers,K., E., *Peer Reviews in Software,* Boston: Addison-Wesley, 2002.

[7] Nielsen,J., "Usability inspection methods", *Conference companion on Human factors in computing systems,* 1994, pp. 413-414.

[8] J. Bach. , "Where does exploratory testing fit?" 2001, Accessed 2013 3/25 http://www.stickyminds.com/s.asp?F=S2680_COL_2

[9] Copeland,L., A practitioner's guide to software test design, Artech House, 2004.

[10] Itkonen,J., Mäntylä,M.V. and Lassenius,C., "Defect Detection Efficiency: Test Case Based vs. Exploratory Testing", *First International Symposium on Empirical Software Engineering and Measurement,* 2007, pp. 61-70.

[11] Itkonen,J. and Mäntylä,M.V., "Are Test Cases Really Needed in Manual Software Testing? – Replicated Comparison between Exploratory and Test-Case-Based Testing", *Empirical Software Engineering,* vol. 19, no. 2, April. 2014, pp. 303-342.

[12] Afzal,W., Ghazi,A.N., Itkonen,J., Torkar,R., Andrews,A. and Bhatti,K., "An experiment on the effectiveness and efficiency of exploratory testing", *Empirical Software Engineering,* vol. 20, no. 3, 2014, pp. 844-878.

[13] Prakash,V. and Gopalakrishnan,S., "Testing efficiency exploited: Scripted versus exploratory testing", *3rd International Conference on Electronics Computer Technology (ICECT),* 2011, pp. 168-172.

[14] Itkonen,J. and Rautiainen,K., "Exploratory testing: A multiple case study", *Proceedings of the International Symposium on Empirical Software Engineering,* 2005, pp. 84-93.

[15] Pichler,J. and Ramler,R., "How to test the intangible properties of graphical user interfaces?", *1st International Conference on Software Testing, Verification, and Validation,* 2008, pp. 494-497.

[16] Pfahl,D., Yin,H., Mäntylä,M.V. and Münch,J., "How is exploratory testing used?: a state-of-the-practice survey", *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement,* 2014, pp. 5.

[17] Itkonen,J., Mäntylä,M.V. and Lassenius,C., "The Role of the Tester's Knowledge in Exploratory Software Testing", *IEEE Transactions on Software Engineering,* vol. 39, no. 3, 2013, pp. 707-724.

[18] Shah,S.M.A., Gencel,C., Alvi,U.S. and Petersen,K., "Towards a hybrid testing process unifying exploratory testing and scripted testing", *Journal of Software: Evolution and Process,* vol. 26, no. 2, 2014, pp. 220-250.

[19] Bach,J., "Session-based test management", *Software Testing and Quality Engineering Magazine,* vol. 2, no. 6, 2000,

[20] Crispin,L. and Gregory,J., Agile testing: A practical guide for testers and agile teams, Pearson Education, 2009.

[21] Zhou,X. and Mäntylä,M.V., "Defect Bash - Literature Review", Proceedings of the 8th Evaluation of Novel Approaches to Software Engineering (ENASE), 2013, pp. 125-131.

[22] Desikan,S. and Ramesh,G., *Software testing: principles and practice,* Pearson Education India, 2006.

[23] Vander Mey,C., Shipping Greatness: Practical Lessons on Building and Launching Outstanding Software, Learned on the Job at Google and Amazon, " O'Reilly Media, Inc.", 2012.

[24] Runeson,P. and Höst,M., "Guidelines for conducting and reporting case study research in software engineering", *Empirical Software Engineering,* vol. 14, no. 2, 2009, pp. 131-164.

[25] Petersen,K. and Wohlin,C., "Context in industrial software engineering research", Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement, 2009, pp. 401-404.

[26] Humphrey,W.S., *Introduction to the personal software process,* Addison-Wesley Professional, 1997.

[27] Chillarege,R., Bhandari,I.S., Chaar,J.K., Halliday,M.J., Moebus,D.S., Ray,B.K. and Wong,M.-., "Orthogonal defect classification-a concept for in-process measurements", *IEEE Transactions on Software Engineering,* vol. 18, no. 11, 1992, pp. 943-956.

[28] Berling,T. and Thelin,T., "An industrial case study of the verification and validation activities", *Proceedings of the Ninth International Software Metrics Symposium.,* 2003, pp. 226-238.

[29] Borg,A., Porter,C. and Micallef,M., "Is Carmen better than George?: testing the exploratory tester using HCI techniques", *Proceedings of the 37th International Conference on Software Engineering-Volume 2,* 2015, pp. 815-816.

[30] Yin,R.K., Applications of case study research, Sage Pubns, 2002.

[31] Dybå,T., Sjøberg,D.I.K. and Cruzes,D.S., "What works for whom, where, when, and why?: on the role of context in empirical software engineering", *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement,* 2012, pp. 19-28.