<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper published in *Information and Software Technology*. This paper has been peer-reviewed but does not include the final publisher proof-corrections or journal pagination.

Citation for the original published paper (version of record):

Asplund, F. (2018)

Exploratory Testing: Do Contextual Factors Influence Software Fault Identification?

Information and Software Technology

<https://doi.org/10.1016/j.infsof.2018.11.003>

Access to the published version may require subscription.

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-239580>

Exploratory Testing: Do Contextual Factors Influence Software Fault Identification?

Fredrik Asplund^{a,*}

^a*KTH Royal Institute of Technology, Department of Machine Design, Division of Mechatronics, Brinellvägen 83, 10044 Stockholm, Sweden*

Abstract

Context: Exploratory Testing (ET) is a manual approach to software testing in which learning, test design and test execution occurs simultaneously. Still a developing topic of interest to academia, although as yet insufficiently investigated, most studies focus on the skills and experience of the individual tester. However, contextual factors such as project processes, test scope and organisational boundaries are also likely to affect the approach.

Objective: This study explores contextual differences between teams of testers at a MedTec firm developing safety-critical products to ascertain whether contextual factors can influence the outcomes of ET, and what associated implications can be drawn for test management.

Method: A development project was studied in two iterations, each consisting of a quantitative phase testing hypotheses concerning when ET would identify faults in comparison to other testing approaches and a qualitative phase involving interviews.

Results: Influence on ET is traced to how the scope of tests focus learning on different types of knowledge and imply an asymmetry in the strength and number of information flows to test teams.

Conclusions: While test specialisation can be attractive to software development organisations, results suggest changes to processes and organisational

*Corresponding author

Email address: fasplund@kth.se (Fredrik Asplund)

structures might be required to maintain test efficiency throughout projects: the responsibility for test cases might need to be rotated late in projects, and asymmetries in information flows might require management to actively strengthen the presence and connections of test teams throughout the firm. However, further research is needed to investigate whether these results also hold for non safety-critical faults.

Keywords: Exploratory testing, Knowledge management, Test management

1. Introduction

Exploratory Testing (ET) has had a place in the practitioners' toolbox for a long time [1], but has so far received little attention from the academic community [2, 3, 4]. The definition of ET is thus still not set in stone: in SWEBOK
5 ET is simply defined as "simultaneous learning, test design, and test execution"
[5], while other formative sources define it as tied to the existence of a test charter — "a short declaration of the scope of a short (1 to 2 hour) time-boxed test effort, the objectives and possible approaches to be used" [6]. Regardless, the approach implies that testers minimize planning, maximize test execution
10 and leverage on their knowledge to decide what parts of the software to spend time testing. Focus is on using the skills and experience of the individual tester quickly and to best effect [7]. Test cases, i.e. specifications for the inputs, conditions and expected outcomes of tests, are usually neither defined prior to test execution nor meticulously recorded [8]. Due to the resulting opaqueness
15 in regard to what is tested and why, ET can be seen as more of a complement to other types of testing [6]. However, evidence suggests that ET can be more effective than testing using documented test cases [9].

Leaning on the existence of test charters in ET, Ghazi et al. describes five levels of exploration, each defined by a different test charter type [10]. At high
20 levels of exploration the test charter only provides a test object or a few high level goals for testing, while test charters at low levels of exploration include more or less completely defined test cases. The latter has, in earlier discourse, rather

been contrasted to ET as Test-Case Based Testing (TCBT) or Scripted Testing, i.e. software testing based on documenting knowledge in test cases that are later
25 executed by testers in a "mechanistic" fashion [11, 2, 12, 13, 9]. While this study adheres to the commonly used terminology, the study by Ghazi et al. [10] does highlight two important issues: studies on ET have not necessarily considered influences from sources external to the tester on ET or made much distinction between differences in the techniques used in "traditional" types of testing, such
30 as how test cases are designed or whether their execution is automated. This implies the need for further studies on the contextual influences on ET and contrastive studies between ET and traditional testing techniques. This study is of the former type, seeking to broaden the knowledge on how differences between testers — such as project processes, test scope and contacts across
35 organisational boundaries — influence ET.

The paper starts by outlining academic and industrial work related to ET. Different parts of the research design are described in detail, including a motivation for the use of a mixed model study, the study subject (in the form of a project at a MedTec firm) and limitations of the study (such as only using
40 using safety-critical software faults for the sake of validity). Two hypotheses are defined based on interactions with employees at the firm: ET should on average identify faults later than TCBT; and distributions of faults should not differ between ET and TCBT for testing activities that otherwise closely resemble each other. This is followed by quantitative results from the study based on a
45 statistical analysis of safety-critical faults, which suggest that the two hypotheses do not hold. These quantitative results are then discussed and related to qualitative findings from interviews with employees throughout the company. Analysis suggests that the scope of tests in different testing activities can influence learning at different levels of exploration by focusing it on different types
50 of knowledge, and implies an asymmetry in the strength and number of information flows to test teams. The paper ends by concluding that while test specialisation can be attractive to software development organisations, results suggest changes to processes and organisational structures might be required

to maintain test efficiency throughout projects: the responsibility for test cases
55 might need to be rotated late in projects, and asymmetries in information flows
might require management to actively strengthen the presence and connections
of test teams throughout the firm.

2. Related Work

Software testing in industrial contexts usually does not rely on the existence
60 of a complete set of well-defined test cases [13]. Testers tend to operate within a
continuum between mechanistic execution and disregard of test cases [1]: signs of
software error are pursued even if found by chance and outside the scope of test
cases. It has been suggested that this type of pursuit in its most ad hoc form is
the most widely practiced software testing [5]. However, by adding structure to
65 this approach it can be optimized through test management, testing strategies,
tools and training [14, 1, 7]. The resulting test techniques have been captured
under the term Exploratory Testing (ET), emphasizing the identification of
faults in the moment based on experience, skills and knowledge in favour of pre-
generated test cases [8]. One of the more well-known variants is Session-Based
70 Testing (SBT), which collects ET into sessions with clear time boundaries, focus
areas and documented results [15]. The areas, approaches and limitations for
testing in SBT sessions are given by test charters, which helps share experience,
knowledge and priorities among testers as well as other roles [16]. Practitioners
have reported SBT from as varied domains as web applications and MedTech
75 [17, 14, 18]. While ET is widely used by practitioners, the associated techniques
can be considered understudied: there is only a small set of associated studies
[2, 4], and a lack of comparative [9] and secondary studies [19].

The existing evidence concerning the efficiency of ET is still contradictory.
As an example, while Itkonen et al. found that TCBT led to significantly more
80 false positives than ET but did not find any difference in fault identification
efficiency or the types of faults identified [11, 13], Afzal et al. arrived at diamet-
rically opposed results [9]. Even though it is not straight-forward to statistically

prove differences in outcomes between e.g. experts and novices in this regard [9], the reliance on the individual tester in ET has led several studies to focus inwards: perhaps unsurprisingly the experience and knowledge of testers do seem to be important to identifying faults [12], albeit through a complex relationship between multifaceted phenomena. For example, a difference can be drawn between generic, testing, domain and system knowledge [7, 2], and considering more aspects of the output than fault identification, such as coverage of functionality, might rather suggest that the successful combination of TCBT and ET is more important to achieve optimal testing [12].

Itkonen et al. studied the application of ET, identifying how focused domain and system knowledge is used as a test oracle, while holistic domain and system knowledge is used for on-the-fly test design [20]. Results by Gebizli and Szer suggest that the level of domain and system experience influences the efficiency and ability to identify critical software faults [4]. Shah et al. argues that it is not the level but the higher reliance on knowledge that is decisive, since the requirements associated with certain types of faults will more frequently be implicit [21]. That the interaction between knowledge and type of fault is important has also been suggested in relation to testing knowledge: although formally trained testers are in general more efficient in ET, this does not hold for content, logical or functional UI defects [3]. This implies an underlying complexity in managing ET, since testers will differ in how and when they evolve different types of knowledge. This can result in a dependence on external consultants or even the hobbies pursued in the testers' own time [7]. Furthermore, ET itself has been identified as a way of learning about the system, even for supporting TCBT [22] or purposes other than testing [8].

However, the ways in which differences in individuals matter are not solely internal - testing often has a cooperative side to it [23]. The human behavioural aspects of software engineering are to date not well studied, from the individual to the organisational level [24]. Shoaib et al. identified an extrovert personality type as an asset when conducting ET, but without pinpointing the reason beyond the relationship [25]. ET conducted by teams has been found to be

more efficient than individually performed ET [26]. This might imply that the
115 interaction between ET and a tester’s context is an important part to explain-
ing its success. This is further supported by practitioners’ reports on the tight
integration established between testers and test managers to facilitate ET [17],
and on the ability of testers employing ET to draw on experience from others
[1]. A few studies that involve ET but do not explicitly separate it out also
120 report results that tentatively support this idea: software developers often feel
defect reports do not include a clear description of the steps required to trig-
ger a failure [27], a problem that is more pronounced in ET than in TCBT
[10]; at the same time, written communication between developers and testers
frequently has to be augmented by informal communication to e.g. overcome
125 differences in experience [28]. The efficiency of ET might thus in part be due
to an increased need to communicate. In a wider sense, this can be connected
back to the sharing of knowledge in regard to complex systems: testing might
in itself be an activity that reveals how the system should be designed and thus
be dependent on a connection to as many roles and perspectives as possible
130 [29, 30].

3. Research Design

This section starts with presenting the background, structure and hypotheses
of the study. It ends with a subsection providing further details on issues of
validity and reliability.

135 3.1. Background

The Firm studied is a large international company, which develops MedTech
products. Access to the Firm’s fault data and employees was offered providing
that liability issues were considered by not further disclosing the company name
or the magnitude of the fault numbers. These requirements were due to the
140 Firm developing safety-critical devices, which is by its very nature a sensitive
endeavour. The intent of the study was to explore how the context of testers

can influence the outcomes of ET, and what associated implications there might be for test management. To address this requires answers to several questions, including:

- 145 1. Which are the outputs of ET that can be affected by the context of testers?
2. What can be expected of this output if the effect of contextual factors is not decisive?
3. If there is an effect, how can it be best explained?
4. What are the likely implications for test management of an effect or the
150 absence of an effect?

Arguably, in regard to an understudied phenomenon that requires exploration, these questions are best answered initially through qualitative means. However, answering the third and fourth question would be substantially strengthened by quantitative measurements proving an actual effect. This motivates a
155 mixed model study that emphasizes the use of quantitative and qualitative approaches in tandem to increase the overall strength of the study [31]. In contrast to e.g. case studies the emphasis is not primarily on achieving internal validity through the properties of the study subject, for instance through it being critical or extreme [32, 33]. However, while the properties of the study subject were
160 not decisive for internal validity, they do have implications for external validity that will be discussed in Subsection 3.3.2. The research was thus designed as a *sequential mixed model study* [34]. A development project was studied in two iterations, each consisting of a quantitative phase involving statistical analyses of faults and a qualitative phase involving interviews. The quantitative phases
165 involved analysing several thousand defects, of which about one tenth were included in the statistical analyses. The qualitative phases involved interviewing 9 employees of the Firm.

The first iteration considered the time between the establishment of an initial stable baseline of artefacts and a complete design specification being in
170 place. The outcomes were an understanding of the situation at the company and hypotheses regarding what would be found in future phases. Later, when

Figure 1: Study Phases

the project had finished, these outcomes supported a second iteration in further analysing the time between the end of the first iteration and the last time a test case was executed. A general introduction to the statistical concepts of the quantitative phases is provided by Keppel and Wickens [35], with more specific choices described in Section 4. During the qualitative phases interviews were transcribed, coded and analysed using content analysis [36]. Coding was first descriptive [37], and then focused on identifying patterns [38]. The latter also involved studying the proximity of codes based on related interview questions. Categories were then identified inductively [39]. The structure of the study is visualized in Figure 1.

To avoid comparability issues the study was limited to *software* faults, since different engineering disciplines did not use the same testing approaches and techniques. In this paper the focus is further limited to *safety-critical* software faults. This is due to the emphasis put on safety-critical faults, which ensured that they were stringently handled. Testers had some leeway to ignore other faults, which could bias the study. Furthermore, the Firm separated testing activities into *Function Test* (FT) and *System Test* (ST). Therefore, these activities were each analysed in isolation in regard to TCBT and ET. FT and ST followed the same process and were organized through the same Change Control Board (CCB), but focused on functionality and the whole system respectively. FT was thus related to functions of the system rather than functions in code. Both FT and ST used the same method of pre-generating test cases based on product requirements, i.e. both activities used the same TCBT approach. However, the test cases were defined and executed by different testers belonging to

Figure 2: Testing Activities and Approaches, Process Flow

different parts of the organization. Several standards and guidance documents were adhered to, since the Firm exports products to multiple markets. Due to the auditing the Firm was subjected to a good summary of the ideas underlying the testing activities can be found in guidance provided by the US Food and Drug Administration [40], especially in regard to the emphasis placed on "specification-based" (i.e. functional) testing and traceability. ET in the form of SBT was conducted in parallel to both activities. Common SBT foci included functionality identified as being error prone, aspects of requirements not fully covered by test cases, and functionality closely related to currently active FT and ST test cases. The process flow for the testing activities and approaches are visualized in Figure 2.

3.2. Hypotheses

With pre-generated, specification-based test cases traced against changes for the sake of meeting regulations, the active pursuit of faults was not the primary objective of the TCBT in the way it was for the ET. TCBT rather focused on establishing a base of proven functionality, and then ensuring that it remained free of faults. Furthermore, safety-critical functionality was emphasized by the Firm, and often found at the core of functionality or even implemented as dedicated safety functions. Safety-critical functionality was thus exercised frequently even before being released to testing activities. Therefore, TCBT was expected to find safety-critical faults early, but then mostly act as proof of the non-existence of this type of fault. At the end of the first iteration this led to the formation of a first hypothesis for this case:

Hypothesis 1. *For the studied type of fault the average time of identification should be less for TCBT, than for ET.*

It might be easier to define a comprehensive set of test cases for functions than for systems, as functions can be more limited and have clearer boundaries. This might suggest that ST would rely more on ET than FT does. However, as the certification of safety-critical functionality requires hazards and their mitigation to be unambiguously identified, ST aimed at finding safety-critical faults cannot avoid spending the required effort on TCBT by referring to FT. This, coupled to the fact that FT and ST were based on the same process, led to the formation of a second hypothesis for this case:

Hypothesis 2. *The difference between distributions of the studied faults identified by TCBT and ET should not differ between FT and ST.*

Phrased differently, the second hypothesis can be stated as there not being any interaction between the activities (FT and ST) and the approaches (TCBT and ET).

These hypotheses are motivated by the aforementioned intent behind the study: to explore how the context of the individual tester can influence the

outcomes of ET, and what associated implications there might be for test management. If the context does not significantly affect the testing activities, then the hypotheses can be expected to hold; if the hypotheses do not hold, then the circumstances at the firm can be used to identify explanations that have a bearing on ET and, by extension, test management.

3.3. Ensuring Validity and Reliability

The overall approach to ensure the validity of the study is discussed in the first subsection, the two subsequent subsections deal with concerns related to the separate phases, and the last subsection summarizes by relating the actions taken to explicit threats to validity.

3.3.1. A Mixed Model Design

Part of the intent of the mixed model design was to triangulate methods to improve validity [41]: the opinions of interviewees regarding activities and approaches were supported or refuted by fault identification statistics. The quantitative results also supported the planning of the qualitative phases [41]: as an example, that FT and ST were identified as important testing activities ensured that the interviews were neither off-topic, nor based on misunderstandings. The largest validity concern with this sequential mixed model approach is that not all options for following up on quantitative results are properly considered [41]. To mitigate this concern a well-defined taxonomy was used during the quantitative phases. The choice fell on IBM's Orthogonal Defect Classification (ODC) [42]. This taxonomy was updated iteratively to capture pertinent information on faults and associated activities gathered throughout the study. In this way descriptive statistics could be generated continuously to support the generation of questions for the interviews. Furthermore, to ensure continuity throughout the study, a field journal was used to document observations, questions, etc.

3.3.2. The Quantitative Phases

During the quantitative phases several concerns related to the quality of the gathered data were considered. As mentioned in Subsection 3.1, for reasons of

265 comparability and reliability, only safety-critical software defect reports were
analysed. Mislabelled defect reports that e.g. dealt with feature requests were
removed, and duplicate defect reports were removed with faults attributed to the
first fault mitigation activity to identify them. The decision to remove duplicates
was partly related to the small amount of faults thus being removed, which was
270 explained by the interviewees as a sign of diligence on part of the testers in not
reporting previously identified faults. The general belief was that if function and
system testers did not check each others' fault reports for duplicates the fault
databases would quickly overflow. Trying to account for this effect would only
have served to make the statistical tests exceedingly subjective. Furthermore,
275 the logic behind both Hypothesis 1 and 2 suggests that including or removing
duplicates should not affect the associated statistical tests. The reliability of
the data was considered in regard to observer bias: to ensure that assessments
could be consistently replicated, employees at the Firm continuously reviewed
observations. A few objections were raised and learnt from, mainly in the first
280 quantitative phase with regard to the assessment of the criticality and effect of
faults.

To ensure that the gathered data was representative the sampling of the
study was considered. Three contexts were deemed important, namely those of
the project, the company and the domain.

- 285 • The project was chosen because it involved a major change to an existing
product. The project should thus be representative for projects that in-
volve substantial changes to the hardware and software of a product. It
should be noted that it might not be representative in regard to projects
developing new products or only introducing small changes.
- 290 • The choice of company was limited to those to which contacts were already
established. While geography might have introduced a selection bias in
the study, the fact that the Firm adheres to standards and guidance for
multiple markets should ensure at least a common denominator to many
other MedTech firms.

295 • The MedTech domain is characterized by its own best practice and standards. While companies are usually anonymized in literature, there are early examples from the MedTech industry in the ET discourse [14]. It should be noted that findings might not carry over to domains which do not have a history of using ET.

300 The implied limitations to generalizability have been considered in the discussion and conclusions.

3.3.3. *The Qualitative Phases*

The choice of interviewees was based on the intent to cover as many perspectives as possible, and included employees with experience of working as developers, project managers, managers, testers, designers and support environment
305 customizers. In total this meant 9 employees were interviewed for an average of 2 hours each. Table 1 provides the profiles of the interviewees in regard to their primary role at the time of the interviews, their secondary skills from working with other functions of product development, the years spent at the Firm and
310 the total number of years working with relevant product development. As can be seen in the table the interviewees had spent considerable time both at the Firm and in relevant domains of product development. Most interviewees had been directly involved in testing activities and the others had interacted with testers. It was an explicitly stated aim of the Firm that all testers should hold
315 at least one International Software Testing Qualifications Board (ISTQB) certification. While most of the 35-40 testers involved in the studied project did hold such certificates, they did not focus exclusively on ISTQB. As an example, the interviewee primarily working with software testing held a certified tester certificate from ISTQB, but rather than holding an agile tester ISTQB certificate
320 was certified as a Scrum Master.

To ensure continuity the main investigator was present throughout all interviews, but was a few times supported by additional interviewers. A detailed interview script was used, as part of rigorously adhering to the procedure defined

Table 1: Profiles of Interviewees

Primary Role	Secondary Skills	Years at Firm	Years of Experience
Configuration Management	Function Testing, Software Development	10	30
Management	Function Testing, Software Architecture, System Architecture, System Testing	7	17
Management	Project Management, Software Development, System Architecture	4	9
Project Management	Management, Quality Management	3	18
Requirements Engineering	System Architecture	4	39
Software Development	Embedded System Design	30	30
Software Testing	Function Testing, System Testing	6	15
System Architecture	Function Testing, System Testing, Software Development	7	20
System Architecture	Embedded System Design, System Testing	7	16

by Kvale and Brinkmann [43]: from *thematizing* and *designing* the investigation; through *conducting*, *transcribing* and *analysing* the interviews; to *verifying* and *reporting* the results.

This procedure involved identifying potential reasons why interviewees might provide biased or incorrect information. The strongest reason was thought to be that the interviewees might not want to voice opinions that might negatively affect their career. To avoid this issue, all interviewees were given the chance to remove parts of their interview transcript, although ultimately none did so. Resistance to change was also considered a likely reason for interviewee bias. To avoid this, different ways in which it could manifest itself were enumerated and a more confrontational interview technique adopted. To limit the risk of interviewer bias, an observer was present during the first interviews to provide feedback on the process.

3.3.4. Summary

As outlined in the previous subsections several actions were taken to ensure the validity and reliability of the study. This subsection uses the classification scheme by Runeson et al. to summarize and elaborate both on these and on the further actions taken [44]. This serves to make explicit the threats the actions were intended to address.

The *construct validity* of a study is the extent to which what is measured really represents what the researcher had in mind to measure. The largest threat in this regard was that either the activities conducted or the faults generated at the Firm would be misunderstood. This could for instance mean that faults would be mislabelled in regard to type or criticality, or that important questions would be omitted from interviews. The primary mitigation of this threat was to continuously describe the activities and faults using descriptive statistics based on the ODC taxonomy. This for instance allowed a thorough set of questions to be asked during the interviews.

The *reliability* of a study is the extent to which it depends on the researchers that conducted it, for instance if other researchers would be able to easily repli-

cate it. The primary threat in this regard was that the fault data would be
355 biased by the habits of the testers, for instance by testers choosing to ignore
faults that they did not consider important enough to merit attention. This was
mitigated by limiting the study to safety-critical faults, which were stringently
handled due to their implications for customers. The quantitative data was also
not accepted at face value, but checked and discussed with employees at the
360 firm to remove mislabelled and duplicate defect reports. The removal of a small
number of duplicate defect reports was based on it not negatively affecting the
testing of the hypotheses whilst making the used data set more uniform. Over-
all, there were few objections raised by the employees, but those that came up
were noted in a field journal and learnt from. Reliability was also considered
365 during the interviews: a few of the interviews were attended by an additional
interviewer that considered interviewer bias, for instance that the interviewer
would forget parts of the interview script or ignore obvious follow-up questions.
To avoid the possibility that important parts of the investigation were not car-
ried out completely and consistently, all issues that needed further attention
370 were noted in the field journal.

The *internal validity* of a study is the extent to which it minimizes systematic
error. As the purpose of the study initially suggested a qualitative approach,
the largest threat in this regard was that the interviewees would have a biased
perception of the state of activities at the Firm. Often repeated statements
375 can become established truths even when there is no basis in reality. Several
measures were taken to mitigate this possibility, including choosing interviewees
from varying backgrounds and with many years of direct experience of testing
at the Firm. However, the primary mitigation of this threat was the triangula-
tion of the quantitative results with the qualitative investigation. Rather than
380 assuming that the interviewees' assumptions in regard to the hypotheses were
true, they were tested and the results brought back to the interviewees. This,
combined with a more confrontational interview technique, meant that the in-
terviewees could be challenged for details concerning their assumptions and then
asked about alternative explanations. To avoid systematic error in the analysis

385 of the results, the study also made use of member checking and an audit trail
[45]. Member checks were extensively used to support the content analysis, with
some of the interviewees asked to comment on interpretations and conclusions
throughout the study. Two interviewees were also asked to comment on the
end results. A professor in software testing, external to the study, was asked to
390 act as an auditor. Comments from member checks and the auditor have been
included throughout the paper.

The *external validity* of a study is the extent to which its findings can be
generalized to other contexts. By studying an industrial context the external
validity in regard to production environments is maximized. Furthermore, with
395 the limitations imposed by considering reliability, the studied activities at the
Firm were strictly guided by safety-related regulations, which should make it
easier to transfer the findings to domains with a similar approach to safety. No
specific limitations that might jeopardize such a transfer were noted in the field
journal during the study. However, the literature does suggest that ET has a
400 long history in MedTec. Transferring the results to a context in which ET is a
novel testing technique might therefore be unfeasible.

4. Quantitative Results

This section provides the quantitative results from the complete study and
discusses how these were tied to the qualitative results. The first subsection gives
405 a summary for the reader only interested in the results, the second provides the
details of the quantitative analysis, and the third discusses the link between
the quantitative and qualitative phases. In discussing the link between the
phases, the third subsection also motivates primarily using information on e.g.
fault types for generating interview questions, rather than when conducting
410 statistical tests.

4.1. Summary

The medians for the TCBT and ET fault distributions are visualized in
Figure 3. Observations indicate that Hypothesis 1 did not hold: TCBT did not

Figure 3: TCBT and ET, Fault Distributions

Figure 4: TCBT and ET across FT and ST, Fault Distributions

on average uncover faults earlier than ET. The separation of faults identified by
 415 FT and ST is visualized in Figure 4. Observations indicate that Hypothesis 2
 did not hold: the difference between distributions of faults identified by TCBT
 and ET differ between FT and ST.

4.2. A Detailed Analysis

As mentioned, a general introduction to the statistical concepts of this sub-
 420 section are given by Keppel and Wickens [35]. Analyses are all based on the
 time, in days, it took for FT and ST to identify faults.

Inspection of the data set revealed outliers, as assessed by inspection of a
 boxplot. Furthermore, the data was neither normally distributed, as assessed by
 Shapiro-Wilk's test ($p < .05$), nor did it support the assumption of homogeneity
 425 of variances, as assessed by Levene's test for equality of variances ($p < .05$).
 Therefore, based on Fey's recommendations, an Aligned Ranks Test (ANT) was

used [46], calculated according to the procedure outlined by Leys and Schumann [47]. ANT was conducted on the influence of the two independent variables - activity and approach - on the time to fault identification, measured in days.

430 Activity included two levels (FT and ST) and approach included two levels (TCBT and ET). ANT indicated a medium, statistically significant interaction between approaches and activities, $F(1, df.Res) = 8.491$, $p < .01$, $\eta^2 = 0.069$.

An interaction contrast was run to compare the "difference between the differences" in time to fault identification between approaches (TCBT and ET) 435 during FT and between approaches during ST. Holm's correction for multiple comparisons was used. Adjusted p-values are presented. This test of difference of differences indicates that the difference between approaches during FT was significantly different than during ST: the difference between the differences was 44.491, $p < .01$.

440 Simple main effects were also calculated using ANT as outlined by Leys and Schumann [47]. There was a statistically significant difference between TCBT and ET in FT, $F(1, df.Res) = 4.830$, $p = .03$; the median score was higher for TCBT (202.5) than ET (142.5). There was also a statistically significant difference between TCBT and ET in ST, $F(1, df.Res) = 16.9$, $p < .01$; the median 445 score was lower for TCBT (130.0) than ET (135.5). Understanding whether these results signal a real difference between populations can be done by estimating effect sizes using the Hodges-Lehmann estimator ($HL\Delta$) on the cohorts in question [48]. $HL\Delta$ is originally only intended to be used for distributions with similar shapes. However, it has been shown that $HL\Delta$ can be used in the 450 case of symmetric distributions [49]. Inspection of a boxplot and comparing medians to means indicate that, apart from a few outliers, the time to fault identification measured in days is fairly symmetrical in each cohort. $HL\Delta$ is estimated to 46.6 (95%: 11.0, 100.0) for time to fault identification in FT, indicating that the results signal a real difference between TCBT and ET in 455 FT. However, $HL\Delta$ is estimated to -20.0 (95%: -42.0, 12.0) for time to fault identification in ST.

To summarize: TCBT identified faults significantly later than ET during

FT; it was not possible to establish whether the identification of faults across time differed between TCBT and ET during ST; and the difference in time
460 between identification of faults by TCBT and ET during FT was significantly different from the difference in time between identification of faults by TCBT and ET during ST.

4.3. *The Link Between Phases*

As described in Subsection 3.3.1, the quantitative results guided the qualitative phases. This was based both on results from feasible statistical tests, and
465 precautions taken when statistical tests were deemed inappropriate.

Firstly, interviews were strongly guided by the quantitative results indicating the opposite of what was expected based on the logically deduced hypotheses. Rather than taking the statements of the interviewees at face value the study
470 could follow up on likely explanations for the unexpected outcomes. Indeed, the chosen research design is known to be especially strong when unexpected outcomes arise during quantitative analysis [41].

Secondly, less advanced statistics provided support by describing important parts of the testers' contexts. For instance, the dominance of FT and ST among
475 testing activities, as shown in Figure 5, was confirmed during the first quantitative phase. Based on such descriptive statistics, the success and failure of other activities in identifying faults could also be discussed with the interviewees: the fault identification during requirements reviews was deemed to support both TCBT and ET, as requirements were both the base for test cases and for in-
480 formation sharing between new and old employees; similarly, interviewees did not believe the results were skewed by tests prior to FT and ST, such as testing software function calls and components, as they were not systematic and rather carried out to support code development.

Thirdly, while clear quantitative results help refute invalid assumptions, they
485 also risk skewing interview questions solely towards the hypothesis testing. To ensure that no options for following up on quantitative results were missed, the Firm's fault mitigation process was analysed using a well-known taxonomy, i.e.

Figure 5: FT and ST dominance

ODC [42]. The extended version of ODC was used to profile faults, but even with the basic version each fault can be profiled according to, for instance, one of twenty-one conditions that could trigger faults, one of thirteen impacts and one of seven types of correction. While appropriate for complex products, ODC illustrated the diversity of faults found in the Firm’s products and the resulting difficulty in ensuring the power and assumptions of most statistical tests. Even using only three of ODC’s categories in their basic version and hypotheses testable with e.g. a straight-forward Chi-Square Goodness-of-Fit-Test would most likely require tens of thousands of faults. In fact, that not all types of faults are equally likely to occur makes many statistical tests inappropriate in regard to the Firm’s complex products. Therefore, descriptive statistics were used to generate questions regarding areas difficult to evaluate using statistical methods.

For example, to investigate bias among the safety-critical faults identified by FT and ST, graphs were generated that showed the time at which specific types of faults were identified (see Figure 6). These were then used to generate the content of the interview scripts, e.g. a generic statement on the difficulty of han-

Figure 6: Examples of Types of Fault Distributed Across Time

Figure 7: Example of Relative Sizes of Different Fault Types

505 dling many hardware configurations followed by a question regarding whether
this could lead to a late identification of faults.

Descriptive statistics, such as graphs showing the relative sizes of different
fault types, were also generated across all faults to identify factors that could
have an indirect effect (see Figure 7). These were then used to generate questions
510 regarding e.g. common fault types not identified during FT and ST.

Subsection 5.1 and 5.2 thus build on questions primarily prompted by the
hypothesis testing, while Subsection 5.3 builds on questions generated from the
descriptive statistics.

5. Qualitative Results and Discussion

515 The hypotheses from Subsection 3.2 did not hold.

This section offers a number of possible explanations for this by linking the results from the interviews to different discourses: subsection 5.1 presents and discusses qualitative findings in relation to an issue already central to the ET discourse — that of the knowledge of the individual; subsection 5.2 takes
520 an organisational perspective centred on knowledge management. Literature external to the ET discourse are introduced as it becomes important in these subsections. Qualitative findings, identified throughout the interviews through coding and categorization, are *exemplified* in emphasized paragraphs after the discussion of associated circumstances at the Firm.

525 The section ends with Subsection 5.3, which underlines that more research is needed to generalize the findings to non safety-critical faults.

5.1. The Knowledge of the Individual

Factors beyond test case design, such as domain knowledge, can influence test results [11, 7]. Itkonen et al. found that the majority of failures that
530 testers identify through exercising their domain knowledge during ET are easy to provoke [20]. This ease begs the question why developers did not identify these faults during their pre-release testing. One explanation can be a too narrow focus of the developers' own tests, another that those without sufficient domain knowledge are blind to certain faults even if they manage to provoke
535 them.

"A: I work very close to the hardware ... There is no special unit test design process, I rather focus on testing the functionality together with the hardware."

*"A: Defects that are reported late? ... some type of mismatch
540 between the requirements and what the developer deemed necessary to do for practical reasons, and perhaps that has not been fed back completely."*

545 *"A: In my case I, well, there is no one else who works with this
functionality, so I have to write the unit tests myself. I do not think
this is the best solution."*

Function testers focused on testing techniques and worked closer to the software code, which meant that they regularly extended their knowledge in both testing and the technical aspects of the systems. System testers, on the other hand, focused on the overall system, which meant that they extended their
550 domain knowledge.

*"A: So, to summarize the system testers held more experience of the system as a whole, while function testers were more aware of test techniques? B: Likely to be the case, especially since my time in testing. I also believe that function testers worked "closer" to the code and developers, which can require a more in-depth technical
555 knowledge of the system."*

With faults requiring different levels of domain knowledge to identify, system testers were on average in a position to find more faults than function testers. One example of faults that would usually fall at the higher end of the scale
560 were those related to medical aspects indirectly connected to long-term patient survivability. These were only well understood by those with a high level of domain knowledge.

*"A: However, it is highly probable that if we did a study on 1000 patients that are hospitalized for more than 14 days, then this would
565 lower the risk of [adverse effect] with a few per mille ... B: And that understanding is linked to domain knowledge. A: Yes, definitely."*

System testers, being in possession of more and continuously growing domain knowledge, should thus achieve better late ET outcomes as they become less blind to faults. This corroborates the finding by Ghazi et al. that identifying
570 critical faults with ET relies strongly on the tester having sufficient knowledge to be able to evaluate the explored behaviour correctly [10].

However, faults remaining until late should on average be more complex to provoke and require a higher level of domain knowledge to identify. At a certain point this complexity and knowledge will be difficult to routinely recall, making
575 test cases a more certain way to capture the associated faults.

*"A: I believe that there have been problems related to ... unique components. Those which are ... the core of the company. They cannot be bought on the market ... They are also difficult. It is also a good thing that they are difficult, because then no one can
580 copy them. But they are also difficult to handle, and quite a lot is required to keep track of them ..."*

Differences could thus also be explained by a stronger system knowledge among function testers leading to more advanced test cases, making late TCBT outcomes better during FT. Indeed, TCBT is often perceived as producing better test coverage than ET [13, 10]. These results might suggest that this is not
585 simply a question of traceability, but also of externally capturing and evolving relevant knowledge from the testers. Ramler and Wolfmaier suggested that to decide whether to automate a test, one has to acknowledge that manual and automated testing address different types of risk [50]: automated testing addresses
590 regression risk, while manual testing is useful for exploration. Similarly, when differing between different levels of exploration, it might make sense to consider the way a test case could consistently capture and combine the knowledge of many expert testers — acknowledging that even experts might need guidance in regard to very complicated faults. This would corroborate the logic of the
595 experts interviewed by Ghazi et al. in that high levels of exploration might be particularly useful early in the development lifecycle as it leads to scripted tests useful in later phases [10].

Alternatively, the capture of domain knowledge in test cases could simply overall be more difficult in ST.

600 5.2. Organisational Implications

Ahonen et al. observed how different organisational structures could have an effect on testing in software engineering as these structures either facilitate or hinder the spread of improvements [51]. Rooksby et al. noted the same effect due to the day-to-day organisation of resources, and also that failures are attributed with significance through conversations between testers and other roles [23]. As mentioned in Section 2, this is mirrored in the results reported for ET in regard to the importance of the context of testers. In a broader sense these issues are connected to the concept of knowledge management within firms, a topic that has been approached within software engineering from several theoretical perspectives [52]. Studies have e.g. emphasized the importance of system and domain knowledge, as well as the tacit nature of knowledge in software engineering [53]. The main perspectives of knowledge management in software engineering, using Earl’s taxonomy [54], are the Systems and Engineering schools [52]. Both schools are focused on different uses of information technology throughout software development, which fits well with safety practices in MedTech such as the traceability between specification and code [40]. Unsurprisingly this perspective was also not far from the minds of the interviewees.

620 *”A: Requirements handling is in itself problematic for several reasons ... For historical reasons we are very focused on requirement documents and that to some extent locks us into certain practices when working with requirements ... Requirements are essentially only a part of Systems Engineering and it is actually our information model that is not thought through correctly ... It is a database, but still not. It is rather locked into the document structure and you cannot see... you see a hierarchy, but if you want to see things from another perspective it is not easy with today’s [technology].”*

However, most of the implications from the interviews were linked to the behavioural school of knowledge management, which focuses on how organiza-

630 tional structures influence the sharing of knowledge [52]. Several reasons for
 the stickiness, i.e. difficulties in the transfer, of knowledge have been identified
 within software engineering contexts, such as a lack of absorptive capacity or
 access to personnel with the right tacit knowledge [55, 56]. The CCB at the
 Firm was the central forum for people from different roles to discuss faults. The
 635 CCB thus pooled experience and information from across the organisation to
 identify areas to update and test. While the CCB was common for both FT and
 ST, all faults were not necessarily delegated to both parts of the organisation
 and updates to test cases within either group did not necessarily propagate to
 the other. This relates to the notion of absorptive capacity, i.e. the ability to
 640 recognize the value of, assimilate and apply new information [57]. The task
 of translating information into a form that is understandable by an intra-firm
 group (i.e. to be an interface between this group and the external environment
 or other intra-firm groups) is assumed by so called *organisational liaisons* and
gatekeepers [57, 58]. It has been suggested that the outward aspect of gate-
 645 keeping has all but disappeared with the widespread use of the Internet, while
 the intra-firm aspect of knowledge transformation remains relevant [59]. In the
 studied Firm the test leader assumed the role of knowledge transformer as part
 of the CCB, but there was still a frequent need for the test teams to investigate
 further or consult with other experts. This suggests an imperfect absorptive
 650 capacity or transformation of knowledge, which opens up for an asymmetric
 information flow to FT and ST: when receiving information about erroneous
 output from well-defined functionality it would e.g. be easier to identify the as-
 sociated functional tests than any impact on system tests; as a result the CCB
 would offer more support to functional test cases than system test cases.

655 More thorough test cases could identify more complex faults, making late
 TCBT outcomes better in FT.

*"A: Managers usually do not participate in the CCB, but system
 staff, system architects, test staff, software architects. Usually the
 project manager takes part."*

660 *"A: I guess not all of you sit in the CCB? B: No, the test leader
among others, [Name]. In this way it ... falls to a smaller group,
but the test lead has an overall ... is the one who spreads the infor-
mation in those cases which it is ... needed. However, in addition, I
mentioned that we speak to someone with a in-depth understanding
665 of the system, but we also do a check in the defect database for sim-
ilar issues. So, it is more of an active search for information, then,
something handed to you on a ... platter."*

*"A: In theory the system tests could be affected by changes to the
function tests, but it was not assured."*

670 In close alignment with the concept of absorptive capacity we find the idea of
cognitive distance: that peoples' mental categories are more or less unaligned,
and that a firm's primary function is to reduce this "distance" to enable its
employees to work towards a common goal [60]. Skilled function testers look
beyond the functionality they are responsible for and identify faults in parts of
675 the system that their functionality interacts with [61]. However, even if FT can
thus have a broader impact on the system, the reliance on interactions implies
that the nature of FT is more narrowly focused than ST. Function testers can
be expected to direct questions to a narrow set of developers, while system
testers have to talk to more developers and other roles to get answers. System
680 testers are thus exposed to knowledge from across the organisation, and - more
critically - system testers will have more possibilities to narrow the gap between
their own understanding of the world and that of employees elsewhere in the
Firm.

System testers should, by amassing more knowledge from and a greater
685 understanding of other parts of the Firm throughout a project's lifetime, be
more adept at identifying problem areas and directing late ET.

*"A: You will get more contacts in more areas within the firm if
you work with system testing. And to that you can add that several*

*system testers are socially competent, and have therefore ended up
with large networks.”*

690

Just as the knowledge of individuals interacts with the organisational level, so do the relationships between individuals: work-related relationships and friendships between individuals establish social network structures between teams, which influences team efficiency positively by increasing their ability to acquire and apply resources such as knowledge [62]. Software development teams have
695 been shown to rely on these relationships in particular when solving complex and unfamiliar tasks [63]. The influence of networks is not necessarily tied to tight and frequent collaboration: strong arguments exist for weak ties being critical in ensuring the propagation of knowledge, even if recent research indicates that
700 this depends on the environment and complexity of the information [64, 65]. Faults identified in the field were investigated by project leaders from a part of the organization dedicated to customer issues. These project leaders built their own task forces by choosing experts from across the organization. Through the system testers’ interactions with more developers during day-to-day testing they
705 had a higher visibility as experts.

System testers were thus exposed to knowledge of potential value to ET late in the project, when field issues were raised. Indeed, even when not directly involved in solving a field defect, chances were that system testers would hear about it through informal discussions. This would suggest that interactions
710 with other experts is not only important to novice testers in need of a mentor to show them what to explore and test [10], but also to experienced testers to direct ET in the moment.

*”A: If it concerned field defects, then it is the personnel at the customer complaints department who keep things together. They bring
715 in experts from different parts of the organisation ... But then they ask for help based on the need, so if it is something with a high priority they simply go out and pick the people they want.”*

”A: It is the customer complaint officers ... They have an under-

*standing of who works with what, and if they do not know then they
go to the managers and ask.”*

720

There is nothing to suggest that function testers would not acquire a substantial amount of domain knowledge eventually. However, it would seem that system testers are positioned to acquire domain knowledge more quickly and better understand which parts of it are most relevant at that particular moment. All three observations fit well with the idea that testers make use of knowledge on faults originally experienced by others to guide their ET [1].

725

5.3. Late Features

Certain product features were developed late in the project, and considered by interviewees as problematic to test. Graphical user interfaces was one example, since it took time to acquire language translations. Other examples were software installation and maintenance support. None of these features were over-represented in any part or all of the studied fault distributions.

730

”A: Language. Everything that has to do with language is found late and that is of course connected to the translations being delivered late.”

735

”A: Language, service and installation of software and such things, which are introduced late into projects ... yes, this [faults] is found late in the project due to that.”

However, the handling of active configurations, versions and variants were one such late feature, and enabling a new variant would only be indirectly associated to faults. Furthermore, coverage is mentioned as problematic to achieve through ET [8], especially for tests requiring consistency across configurations [10]. Differences could thus be explained by test cases ensuring consistent coverage across product variants. However, while this might explain the late TCBT outcomes during FT, it does not explain the variation between FT and ST.

740

745

Table 2: Explanations Summarized

	Positive	Negative
FT	<p>Easier to acquire and understand the relevance of system knowledge (TCBT).</p> <p>Less absorptive capacity needed to identify the implications of erroneous functionality on test cases leads to more support and direction in developing thorough test cases (TCBT).</p>	None identified.
ST	<p>Easier to acquire and understand the relevance of domain knowledge (ET).</p> <p>More exposure to other roles leads to a narrowing of the cognitive distance to others in the organisation (ET).</p> <p>More exposure to other roles leads to a wider formal and social network with access to more distant knowledge (ET).</p>	More difficult to capture domain knowledge in test cases (TCBT).

These features could have been misconstrued as problem areas, but more likely they resulted in faults that were primarily non safety-critical. The focus on safety-critical faults ensured the validity of the results; however, these qualitative results and the indications mentioned in Section 2 that link differences in ET efficiency to types of faults suggest that further research is needed to generalize the findings to non safety-critical faults.

6. Implications

Table 2 summarizes the identified explanations for the differences in outcome between TCBT and ET during FT and ST. Most of the identified explanations were framed as factors that have a positive influence on different approaches in the different activities. These factors were driven by differences in the scope of tests, but related to the knowledge and contacts of the different types of testers.

Even if the findings might suggest that testers should change positions and work tasks frequently, this could be difficult to achieve. Furthermore, specialisation is not necessarily bad, and exposing testers to more varied knowledge and contacts might be detrimental in the short run if it implies a longer time to reap the gains identified in Table 2. However, the list below outlines less intrusive ways of leveraging on these results identified in dialogue with the studied firm:

- With the tester and not only the test case being of importance, then late regression tests should sometimes be executed by a different set of testers. System testers could e.g. guide their ET by use of function tests.
- The often vague relationship between functional failures and system tests suggests more system testers should attend central forums for discussing faults. This would increase the chance of a sufficient level of absorptive capacity and narrow enough cognitive distance to transform information into knowledge usable for ST.
- Structured forms of ET, such as SBT, hold the potential to mitigate asymmetric information flows. Session foci could be sought from a wider group, e.g. by asking developers and field service to regularly vote on where to focus testing efforts. All ET need not be focused in this way, but it might offer a wider, more supportive perspective.
- Social networks can to some extent be engineered. By organising forums for information exchange that are not based on software architecture or functionality, FT teams could eventually find it easier to acquire and apply information. An example could be to organise cross-firm expert groups.

Testing is not necessarily structured to be as technically efficient as possible, but rather driven by what makes sense to the organisation as a whole [66]. Different roles will have a perspective based in their day-to-day responsibilities, rather than best practice in regard to software process improvement [67]. Indeed, if software testing is perceived as acceptable, then chances are that suggestions for improvements to test processes will be discarded [68]. Other work

practices and organisational structures could thus introduce, if not the exact, at least the same type of problems as found at the studied firm in the form of biased learning and networking. Arguably, these suggestions should therefore
790 be valuable outside the studied firm by emphasizing the need to understand the context of a tester involved in ET.

7. Conclusions

This study suggests that the context of the individual tester is important to safety-related ET in the sense that the scope of tests focuses learning on
795 different types of knowledge and implies an asymmetry in the strength and number of information flows to test teams. While test specialisation can be attractive to software development organisations, these results suggest changes to processes and organisational structures might be required to maintain test efficiency throughout projects: the responsibility for test cases might need to
800 be rotated late in projects, and asymmetries in information flows might require management to actively strengthen the presence and connections of test teams throughout the firm.

Acknowledgments

Special thanks go to Vicki Derbyshire for her help in proofreading; and
805 Professor Karl Meinke for auditing this study.

References

- [1] A. Tinkham, C. Kaner, Exploring Exploratory Testing, in: Proceedings of the 2003 STAR East Conference, 2003.
- [2] J. Itkonen, Empirical Studies on Exploratory Software Testing, Ph.D. thesis,
810 sis, 2011.
- [3] M. Micallef, C. Porter, A. Borg, Do Exploratory Testers Need Formal Training? An Investigation Using HCI Techniques, in: Proceedings of the

2016 IEEE Ninth International Conference on Software Testing, Verification and Validation Workshops, 2016.

- 815 [4] C. S. Gebizli, H. Sözer, Impact of Education and Experience Level on the Effectiveness of Exploratory Testing: An Industrial Case Study, in: 2017 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), 2017.
- [5] IEEE Computer Society, Chapter 4, Software Testing, in: P. Bourque, R. E. Fairley (Eds.), SWEBOK V3.0, Guide to the Software Engineering Body of Knowledge, 2014.
- 820 [6] D. Graham, E. van Veenendaal, I. Evans, R. Black, Test design techniques, in: Foundations of Software Testing, ISTQB Certification, chap. 4, Cengage Learning Emea, ISBN 978-1844809899, 2008.
- [7] A. Beer, R. Ramler, The Role of Experience in Software Testing Practice, in: Proceedings of Software Engineering and Advanced Applications, 2008. SEAA'08, 34th Euromicro Conference, 2008.
- 825 [8] J. Itkonen, K. Rautiaine, Exploratory Testing: A Multiple Case Study, in: 2005 International Symposium on Empirical Software Engineering, IEEE, 2005.
- 830 [9] W. Afzal, A. N. Ghazi, J. Itkonen, R. Torkar, A. Andrews, K. Bhatti, An Experiment on the Effectiveness and Efficiency of Exploratory Testing, Empirical Software Engineering 20 (2015) 844–878.
- [10] A. N. Ghazi, K. Petersen, E. Bjarnason, P. Runeson, Levels of Exploration in Exploratory Testing: From Freestyle to Fully Scripted, IEEE Access 6 (2018) 26416–26423.
- 835 [11] J. Itkonen, M. V. Mäntylä, C. Lassenius, Defect Detection Efficiency: Test Case Based vs. Exploratory Testing, in: Proceedings of International Symposium on Empirical Software Engineering and Measurement, 61–70, 2007.

- 840 [12] S. M. A. Shah, U. S. Alvi, C. Gencel, K. Petersen, Comparing a Hybrid Testing Process with Scripted and Exploratory Testing: An Experimental Study with Practitioners, in: Proceedings of the International Conference on Agile Software Development, Springer, 187–202, 2014.
- [13] J. Itkonen, M. V. Mäntylä, Are Test Cases Needed? Replicated Comparison Between Exploratory and Test-Case-Based Software Testing, Empirical
845 Software Engineering 19 (2014) 303–342.
- [14] B. Wood, D. James, Applying Session-Based Testing to Medical Software, Medical Device and Diagnostic Industry 25 (2003) 90–103.
- [15] J. Bach, Session-Based Test Management, Software Testing and Quality
850 Engineering Magazine 2 (2000) .
- [16] A. N. Ghazi, P. G. Ratna, K. Petersen, Checklists to Support Test Charter Design in Exploratory Testing, in: Proceedings of Agile Processes in Software Engineering and Extreme Programming, 18th International Conference, XP 2017, 2017.
- 855 [17] J. Våga, S. Amland, Managing High-Speed Web Testing, in: Software Quality and Software Testing in Internet Times, 2002.
- [18] J. Lyndsay, N. V. Eeden, Adventures in Session-Based Testing, techreport, Workroom Productions Ltd., 2003.
- [19] V. Garousi, M. V. Mäntylä, A Systematic Literature Review of Literature Reviews in Software Testing, Information and Software Technology 80
860 (2016) 195–216.
- [20] J. Itkonen, M. V. Mäntylä, C. Lassenius, The Role of the Tester’s Knowledge in Exploratory Software Testing, IEEE Transactions on Software Engineering 39 (2013) 707–724.
- 865 [21] S. M. A. Shah, D. Sundmark, B. Lindström, S. F. Andler, Robustness Testing of Embedded Software Systems: An Industrial Interview Study, IEEE Access 4 (2016) 1859–1871.

- 870 [22] K. Frajtak, M. Bures, I. Jelinek, Model-Based Testing and Exploratory Testing: Is Synergy Possible?, in: Proceeding of IT 2016 6th International Conference on Convergence and Security (ICITCS), 2016.
- [23] J. Rooksby, M. Rouncefield, I. Sommerville, Testing in the Wild: The Social and Organisational Dimensions of Real World Practice, *Computer Supported Cooperative Work* 18 (2009) 559–580.
- 875 [24] P. Lenberg, R. Feldt, L.-G. Wallgren, Towards a Behavioral Software Engineering, in: Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering, 2014.
- [25] L. Shoaib, A. Nadeem, A. Akbar, An Empirical Evaluation of the Influence of Human Personality on Exploratory Software Testing, in: Proceedings of the IEEE 13th International Multitopic Conference, 2009, 2009.
- 880 [26] P. Raappana, S. Saukkoriipi, I. Tervonen, M. V. Mäntylä, The Effect of Team Exploratory Testing - Experience Report from F-Secure, in: Proceedings of the 2016 IEEE Ninth International Conference on Software Testing, Verification and Validation Workshops (ICSTW), 2016.
- 885 [27] N. S. M. Yusop, J. Grundy, R. Vasa, Reporting Usability Defects: Do Reporters Report What Software Developers Need?, in: Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering, 2016.
- 890 [28] D. S. Cruzes, N. B. Moe, T. Dybå, Communication between Developers and Testers in Distributed Continuous Agile Testing, in: Proceedings of the 2016 IEEE 11th International Conference on Global Software Engineering (ICGSE), 2016.
- [29] D. Martin, M. Hartwood, R. Slack, A. Voss, Achieving Dependability in the Configuration, Integration and Testing of Healthcare Technologies, *Computer Supported Cooperative Work* 15 (2006) 467–499.

- 895 [30] M. V. Mäntylä, J. Itkonen, J. Iivonen, Who Tested My Software? Testing as an Organizationally Cross-Cutting Activity, *Software Quality Journal* 20 (2012) 145–172.
- [31] J. W. Creswell, Chapter One, The Selection of a Research Design, in: *Research Design*, Sage Publications, Inc., ISBN 978-1-4522-2610-1, 2014.
- 900 [32] P. Runeson, M. Höst, A. Rainer, B. Regnell, Design of the Case Study, in: *Case Study Research in Software Engineering, Guidelines and Examples*, John Wiley & Sons, Inc., ISBN 978-1-1181-0435-4, 2012.
- [33] B. Flyvbjerg, 17. Case Study, in: N. K. Denzin, Y. S. Lincoln (Eds.), *The SAGE Handbook of Qualitative Research*, chap. 17, SAGE Publications, Inc., 4 edn., ISBN 978-1-4129-7417-2, 301–316, 2011.
- 905 [34] A. Tashakkori, C. Teddlie, 8 Extended Examples of Mixed Model Designs, in: C. D. Laughton, E. Carr, W. Westgate, D. Santoyo (Eds.), *Mixed Methodology, Combining Qualitative and Quantitative Approaches*, Sage Publications, 1998.
- 910 [35] G. Keppel, T. D. Wickens, Part III, Two-Way Factorial Experiments, in: *Design and Analysis, A Researchers Handbook*, Pearson Prentice Hall, 4th edn., ISBN 0-13-127776-6, 193–288, 2004.
- [36] C. Adams, et al., *The SAGE Encyclopedia of Qualitative Research Methods*, vol. 1 & 2, SAGE Publications, Inc., 2008.
- 915 [37] J. Saldaña, First Cycle Coding Methods, in: *The Coding Manual for Qualitative Researchers*, 2009.
- [38] J. Saldaña, Second Cycle Coding Methods, in: *The Coding Manual for Qualitative Researchers*, 2009.
- 920 [39] L. Cohen, L. Manion, K. Morrison, 23 Content analysis and grounded theory, in: *Research Methods in Education*, Routledge, 6 edn., 2007.

- [40] U.S. Department of Health and Human Services, Food and Drug Administration, General Principles of Software Validation; Final Guidance for Industry and FDA Staff, Tech. Rep., U.S. Department of Health and Human Services, Food and Drug Administration, 2002.
- 925 [41] J. W. Creswell, Chapter Ten, Mixed Methods Procedures, in: Research Design, Sage Publications, Inc., ISBN 978-1-4522-2610-1, 2014.
- [42] R. Chillarege, I. S. Bhandari, J. K. Chaar, M. J. Halliday, D. S. Moebus, B. K. Ray, M.-Y. Wong, Orthogonal Defect Classification - A Concept for In-Process Measurements, IEEE Transactions on Software Engineering 18 (11) (1992) 943–956.
- 930 [43] S. Kvale, S. Brinkmann, InterViews, Learning the Craft of Qualitative Research Interviewing, SAGE Publications, Inc., 2 edn., ISBN 978-0-7619-2542-2, 2009.
- [44] P. Runeson, M. Höst, A. Rainer, B. Regnell, Data Analysis and Interpretation, in: Case Study Resesarch in Software Engineering, chap. 5, John Wiley & Sons, Inc., ISBN 978-1-118-10435-4, 61–76, 2012.
- 935 [45] J. W. Creswell, D. L. Miller, Determining Validity in Qualitative Inquiry, Theory in Practice 39 (2000) 124–130.
- [46] J. Feys, Nonparametric Tests for the Interaction in Two-way Factorial Designs Using R, The R Journal 8 (2016) 367–378, ISSN 2073-4859.
- 940 [47] C. Leys, S. Schumann, A Nonparametric Method to Analyze Interactions: The Adjusted Rank Transform Test, Journal of Experimental Social Psychology 46 (4) (2010) 684–688.
- [48] M. Hollander, D. A. Wolfe, E. Chicken, Chapter 4, The Two-Sample Location Problem, in: Nonparametric Statistical Methods, John Wiley & Sons, 945 2014.

- [49] G. K. Rosenkranz, A Note on the Hodges-Lehmann Estimator, *Pharmaceutical Statistics* 9 (2010) 162–167.
- [50] R. Ramler, K. Wolfmaier, Economic Perspectives in Test Automation: Balancing Automated and Manual Testing with Opportunity Cost, in: *Proceedings of the 2006 International Workshop on Automation of Software Test*, ACM, 85–91, 2006.
- [51] J. J. Ahonen, T. Junttila, M. Sakkinen, Impacts of the organizational model on testing: Three industrial cases, *Empirical Software Engineering* 9 (2004) 275–296.
- [52] F. O. Björnson, T. Dingsyr, Knowledge Management in Software Engineering: A Systematic Review of Studied Concepts, Findings and Research Methods Used, *Information and Software Technology* 50 (2008) 1055–1068.
- [53] I. Rus, M. Lindvall, S. S. Sinha, A State of the Art Report: Knowledge Management in Software Engineering, techreport, 2001.
- [54] M. Earl, Knowledge Management Strategies: Toward a Taxonomy, *Journal of Management Information Systems* 18 (2001) 215–233.
- [55] G. Szulanski, Exploring Internal Stickiness: Impediments to the Transfer of Best Practice Within the Firm, *Strategic Management Journal* 17 (1996) 27–43.
- [56] S. M. Mitchell, C. B. Seaman, Could Removal of Project-Level Knowledge Flow Obstacles Contribute to Software Process Improvement? A Study of Software Engineer Perceptions, *Information and Software Technology* 72 (2016) 151170.
- [57] W. M. Cohen, D. A. Levinthal, Absorptive Capacity: A New Perspective on Learning and Innovation, *Administrative Science Quarterly* 35 (1990) 128–152.

- [58] M. L. Tushman, Special Boundary Roles in the Innovation Process, *Administrative Science Quarterly* 22 (4) (1977) 587–605.
- 975 [59] E. Whelan, B. Donnellan, W. Golden, Knowledge Diffusion in Contemporary R&D Groups: Re-Examining the Role of the Technological Gatekeeper, in: *Knowledge Management and Organizational Learning*, 2009.
- [60] B. Nooteboom, Learning by Interaction: Absorptive Capacity, Cognitive Distance and Governance, *Journal of Management and Governance* 4 (1-2)
980 (2000) 69–92.
- [61] C. Andersson, P. Runeson, Investigating Test Teams Defect Detection in Function Test, in: *Proceeding of the First International Symposium on Empirical Software Engineering and Measurement*, 2007.
- [62] P. Balkundi, D. A. Harrison, Ties, Leaders, and Time in Teams: Strong
985 Inference about Network Structure’s Effects on Team Viability and Performance, *The Academy of Management Journal* 49 (2006) 49–68.
- [63] D. Šmite, N. B. Moe, A. Šāblis, C. Wohlin, Software Teams and Their Knowledge Networks in Large-Scale Software Development, *Information and Software Technology* 86 (2017) 71–86.
- 990 [64] M. Granovetter, The Strength of Weak Ties: A Network Theory Revisited, *Sociological Theory* 1 (1983) 201–233.
- [65] S. Aral, The Future of Weak Ties, *American Journal of Sociology* 121 (6) (2016) 1931–1939.
- [66] D. Martin, J. Rooksby, M. Rouncefield, I. Sommerville, Good Organisational Reasons for Bad Software Testing: An Ethnographic Study of Testing in a Small Software Company, in: *Proceedings of the 29th international conference on Software Engineering*, 2007.
- 995 [67] N. Baddoo, T. Hall, Practitioner Roles in Software Process Improvement: An Analysis using Grid Technique, *Software Process: Improvement and Practice* 7 (2002) 17–31.
1000

- [68] J. Kasurinen, Software Organizations and Test Process Development, *Advances in Computers* 85 (2012) 1–63.