Communication between Developers and Testers in Distributed Continuous Agile Testing

Daniela S. Cruzes
SINTEF-ICT
Trondheim, Norway
daniela.s.cruzes@sintef.no

Nils B. Moe
SINTEF-ICT
Trondheim, Norway
nils.b.moe@sintef.no

Tore Dybå
SINTEF-ICT
Trondheim, Norway
tore.dyba@sintef.no

Abstract— **Software developers and testers have to work together to achieve the goals of software development projects. In globally distributed software projects the development and testing are often scattered across multiple locations forming virtual teams. Further, the distributed projects are so complex that none of team members can possibly possess all the knowledge about the project individually. During testing in such teams, developers and testers need to coordinate and communicate frequently. However, coordination is affected by the availability of the project information, which is distributed among different project members and organizational structures. Many companies are facing decisions about how to apply agile methods in their distributed projects. These companies are often motivated by the opportunities of solving the coordination and communication difficulties associated with global software development. In this paper we investigate the communication between testers and developers in two teams from two software companies performing continuous agile testing in a distributed setting. We describe four communication practices used by the team: handover through issue tracker system, formal meetings, written communication and coordination by mutual adjustment. We also discuss communication between testers and developers in collocated versus distributed testers and developers. We have found that early participation of the testers is very important to the success of the handover between testers and developers. The communication between developers and testers is not sufficiently effective through written communication and that it changes depending on the type of the tasks and experience of the testers.**

Keywords—Continuous Agile Testing; Global Software Development; Communication;

I. INTRODUCTION

Global software development has matured considerably since its inception and has become an integral part of the information technology landscape. Now, many companies are facing decisions about how to apply agile methods in their distributed projects [14] [26]. One of the main challenges is the disconnection between important activities, such as planning, analysis, design, programming and testing. A number of recent trends have focused on the need to transform organizations to deliver software continuously [12], where a tight connection between development and testing is required to ensure errors are detected and fixed as soon as possible. Agile methods supports continuous development and enable complex and business and safety-critical software to be developed. To develop high-quality software in distributed teams, it is essential to use software testing methods and tools effectively and efficiently.

The current software testing practices are far from satisfactory, as indicated in various studies on software testing practices [16]. Besides, while many companies are facing decisions about how to apply agile testing methods in their distributed projects, there is a lack of empirical studies related to processes, methods and tools for testing and quality assurance in global software development [25][27].

Communication in distributed settings in a main challenge for teams' effectiveness. Alzoubi et al. [1] performed a systematic review of communication challenges in empirical studies of GDAD (Globally Distributed Agile Development), and found out that the knowledge about GDAD communication is limited. The review showed that study results are scattered, inconclusive, and ambiguous, and scarcely any studies opened up the communication process or focused on the social interaction and behavior of teams as part of the research. Hummel et al. [15] also investigated the role of communication in agile software development by conducting a structured, systematic literature review. The authors found that despite its acknowledged importance, the knowledge of communication and agile software development is limited because hardly any studies open up the communication process. The findings highlight a vital research gap that needs more attention, namely, how efficient and effective GDAD communication software teams are achieved in practice and what techniques are used to enhance GDAD communication.

GDAD also creates a number of challenges in relation to alignment between developers and testers. Recently, Barney et al. [3] were the first to explicitly study alignment in a global setting. The goal was to understand the levels of alignment between key stakeholder groups within a company on the priority given to aspects of software quality developed as part of an offshoring relationship. They found that communication and coordination are important factors for misalignment between developers and testers, but did not investigate the issue further.

In this paper, we focus on the communication that happens

2329-6313/16 $31.00 © 2016 IEEE
DOI 10.1109/ICGSE.2016.27

between testers and developers, to capture the complex nature of these team members working together in a continuous testing process in GDAD. Our research questions are:

1) *How do developers and testers communicate in distributed continuous agile testing?*
2) *How is communication impacted by types of team configuration between testers and developers in distributed continuous agile testing?*

The remainder of this paper is organized as follows. Section II we summarize the literature in alignment between development (mostly requirements) and testing phases. We also present the literature on distributed continuous agile testing. In Section III we describe the methodology followed for this research and details of the context in which we base our results. Section IV presents the results, and in Section IV we provide a discussion of the results with implications to research and practice and limitations. Finally in the conclusion we provide the final remarks of the study with the plans for future research.

II. TESTING IN GEOGRAPHICALLY DISTRIBUTED AGILE DEVELOPMENT (GDAD)

A. Brigding the Gaps Between Development and Testing

Studies of the gaps and challenges between development and testing have mostly focused on collocated projects [3][4][29][33][35]. In addition, only one article explicitly mentions that some of the projects followed the agile methodology[4], but not focusing on specific challenges of distributed agile software development (Table 1). Barney et al. [3] do not say explicitly that the two projects are agile, but we inferred them to be agile since Ericsson is known by their long-term usage of agile.

Some studies indicate that communication processes tend to have the capacity to mediate the interactions and misalignment between developers and testers [3][4][29][35]. In addition, Taipale and Smolander [29] defined process improvement propositions, including adjusting testing according to the business orientation of the organizational unit, enhanced testability of software components, early involvement of testing, and use of risk-based testing. Zhang et al. [35] affirms that developer–tester conflict occurs because they do not document their work or follow standardized procedures; they also engage in strongly negative behaviors and consequently develop negative perceptions of each other. Bjarnason et al. [4] adds that cooperation and requirements engineering practices are also a critical basis for alignment. Barney et al. [3] were the first to explicitly study alignment in a global setting and in addition to communication. The main reasons for misalignment were found to include cultural factors, control of quality in the development process, short-term versus long-term orientations, understanding of cost-benefits of quality improvements, and coordination. Uusitalo et al. [33] found a number of practices that increase the communication and interaction between requirements and testing roles, namely early tester participation, traceability policies, consider feature requests from testers, and linking test and requirements people. Linking people or linking artifacts were seen as equally important for the communication effectiveness.

B. Continuous Agile Testing

Continuous agile testing is one approach that is increasingly being adopted by software companies [7]. This approach does not just mean testing on agile projects, but testing an application with a plan to learn about it and let the product information and customer feedback guide the testing. Continuous agile testing, involves immediately integrating changes into the main system, continuously testing all changes and updating test cases to be able to run a regression test at any time to verify that changes have not broken existing functionality [12][21]. For achieving the goals of continuous agile testing it is essential that team members work effectively as a team, and that developers and testers are integrated as a team [17][29]. It is in general challenging to implement this kind of approach in many contexts because it requires more interactions between the developers and the testers, especially for global teams.

Table 1 - Studies on Alignment between Testing and Development

	Taipale and Smolander [29]	***Uusitalo et al. [33]***	***Bjarnason et al. [4]***	***Zhang et al. [35]***	***Barney et al. [3]***
Number of Studied of Companies/Projects	7 (26 Units)	6 (one project each)	6	1 (Large Global company)	2 (3 Projects)
Collocated	Yes	Yes	2 /6 Yes	Not clear	No
Distributed	No	No	4/6 Yes	Not Clear	Yes
Agile	No	No	2/6 Yes	Not Clear	2/3 Yes
Level of business orientation	✔		✔		
Level of testing involvement	✔	✔	✔		
Testing schedule stability	✔				
Level of communication and interaction	✔		✔	✔	✔
Level of planning of testing	✔	✔			
Level of use of components	✔				
Level of complexity	✔				
Coordination and Cooperation			✔		✔
Requirements Quality			✔	✔	✔
Validation and Verification Quality			✔		
Knowledge of Testers				✔	✔
Traceability of Requirements and Tests			✔		
Cultural Factors					✔

Crispin and Gregory [7] discuss the Agile Testing quadrants that are largely adopted in practice. Each quadrant in Fig. 1 reflects different reasons to test. Traditionally software testing focuses almost exclusively on the right hand side (Q3 and Q4), criticizing the product, but not playing a productive part in supporting the creation and guidance of the product (Q1 and Q2). Moreover, traditionally, software testing is involved late in the development process to detect failures but not to prevent them. In agile testing, the testers are not only involved in identifying, but also in preventing failures by continuous interaction with developers and customers. Automation is an important enabler for agile testing. Automation of the tests in Q1 is usually easiest to implement, and at the same time makes a big impact on the process effectiveness. Tests in Q3 are usually performed manually.

Continuous agile testing increases the need for improved communication and coordination between testers and developers, in addition to a new mind-set at the personal and organizational levels. In the context of GDAD projects, geographical and temporal distribution makes regular interaction challenging [1][2]. Yet, few papers have addressed the solutions and challenges in global software testing. Collins et al. [6] describe an industrial experience on the application of distributed testing in a software development environment following Scrum. The authors report on the adaptation in a testing process to make feasible the distributed tests in an agile environment, the solutions implemented to reduce the impact of communication issues and the difficulties observed in the allocation of tasks in the software project. Collins et al. identified four main challenges and key lessons learned for minimizing the impact of the geographical distance between the testing teams: communication and coordination are essential factors for the success of distributed testing; the project information should be available with details to all members; automation reduces the needs of physical presence in the testing process; supporting tools are always important, but testing team organization is more.

Shah et al. [25] conducted a study of three vendor-side testing teams. Their findings show how the participant test engineers perceive global software testing (GST) and deadline pressures, the challenges that they encounter, and the strategies that they use for coping with the challenges. The authors highlight the need for (1) appreciating test engineers' efforts, (2) investigating the team structure's influence on pressure and the GST practice, (3) understanding culture's influence on other aspects of GST, and (4) identifying and addressing quality-dilemma situations.

Tervonen et al. [32] studied the outsourcing of software testing in the Oulu area, the ways in which it is used, and determined the observable benefits and obstacles. The advantages that have most commonly been realized by the companies for the outsourcing of testing were: time zone effectiveness, an improved logging of communication, and closer proximity to the market.

We have previously described an experience report in which we describe how continuous testing based on continuous and frequent feedback ensures knowledge sharing and safeguarding the quality of the system under development [21]. We found the following enablers for a successful virtual agile team that testers were allocated in China and the developers in Norway: coordination by mutual adjustment, dedicated testers and low turnover, shifting working hours, and self-management and autonomy. Non-technical factors, such as socio-technical and organizational factors, have a significant influence on the way software testing is performed in an agile virtual team.

III. Research Methodology

A. Data Collection

In the study presented in this paper, we aim to investigate how testers and developers communicate in GDAD. To answer our research questions we performed two case studies [23][34] in the industry. We followed the teams for about 1,5 years as part of a four year project in which we do action research [9] [18] with the teams. The unit of analysis is the project, one company each.

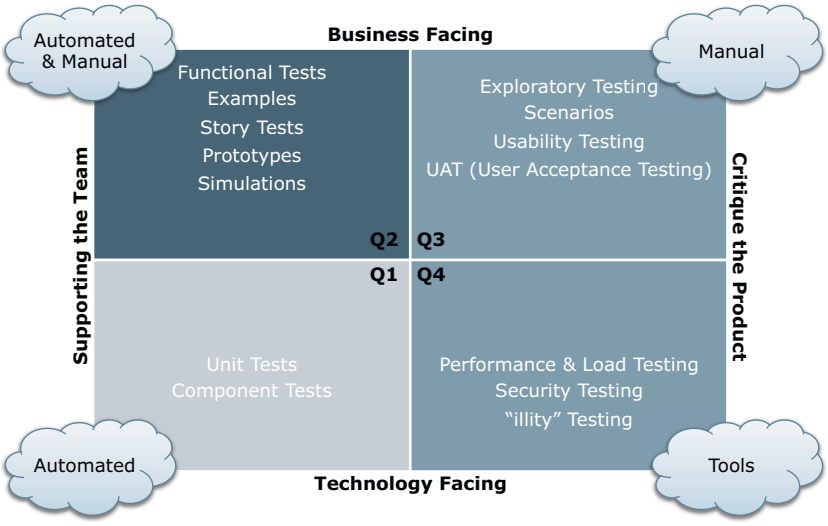

Fig. 1. Agile test quadrants [7]

The case studies reported in this paper are based on two distributed teams at two software companies, both headquartered in Norway, and with software developers and testers distributed in several countries, including Norway, Poland, Sweden, and one of them with a testing center in China (Fig. 2). Moreover, we drew up a non-disclosure agreement for companies. This step was important to establish a formal link between researchers and companies, and ensure data confidentiality. We performed interviews with different roles in each company asking about the communication between developers and testers as shown in Tables 2 and 3.

B. Data Analysis

We used thematic analysis to analyze the data from the semi-structured interviews we performed with team members, a technique for identifying, analyzing, and reporting themes found in qualitative data that are relevant to describe a phenomenon [5][8]. To support the data analysis, we used a tool for creating mind maps of the transcribed interviews. We then classified the data collected according to themes and, finally, all researchers interpreted and discussed the themes. The data from the interviews were analyzed for each company and enabled the within-case analysis. In addition we complement the results with the observations from the different meetings with the teams and managers. Then, we aggregated the results to compare both companies in a cross- case analysis.

C. Company A

Company A is a multinational provider of software for a safer, smarter and greener future in the energy, process and maritime industries. The team as investigated consists of 15 members (12 in Norway) including 4 testers (3 in China, one in Norway) and one test manager (in Norway), as shown in Fig. 2. Some members are working part-time in the team. The team follows a Scrum-based process, where a release is 2-4 months long. The goals of this process are to be a common platform for communication and understanding of the software testing processes; clarify responsibilities, milestones/tollgates and expected outputs; form the basis for project governance; and, be a high level checklist for projects.

The team follows a continuous testing software process. In the Planning phase, the testers develop a high-level test strategy for the product, and a test plan for the releases, and decide on how to follow up the project (test environment, schedule/milestones). In the Development and Testing phase the main objectives are to design and refine test cases for the features (user stories / enhancements) within the scope and to execute and report the test according to the test plan. This phase is typically accomplished in several sprints. In the Stabilizing Phase, the main objectives are to perform a complete system functional test, system integration test, and relevant non-functional tests in a test environment similar to the production environment. Typically, several runs of system and non-functional test are accomplished during the stabilizing phase. Each run of the system test is done on a new release candidate. When the acceptance criteria are matched according to the test plan, the project is released and deployed. After the release, some of the team members start a new release while some do support and others do bug fixing.

Table 2 – Data Collection in the two companies

	Company A	*Company B*
Focused Interview on communication	Test Leader Scrum master Two testers	Two developers Two testers Test Leader
Retrospective Observations	✔	✔
Planning Meetings Observations		✔
Process – Documentation Analysis	✔	✔
Discussions on testing processes	✔	✔

Table 3 – Semi-Structured Interview

Interview Guide
1) Tell us about your testing experience in general and here.
2) Which types of testing do you perform? (e.g. unit, integration, smoking, functionality, system, acceptance, performance, regression, exploratory, manual)
3) How is testing work shared between testers and developers ? a. Are testers part of the team? b. Does the team have a common project goal? c. Is there a defined time scheduled for testing?
4) How are responsibilities and the scope of work defined between developers and testers? a. Do you have an overview of what other developers (and testers) are doing? b. How are tasks handed over to testers? How do a tester know what to test? c. Are there well-defined hand-off points throughout the development process?
5) When and how do testers and developers interact? a. How do testers and developers give each other feedback? b. How do team testers and developers share relevant project information with each other? c. Do testers participate in meetings and discussions? d. Are testers able to discuss solutions? e. What are shared “artifacts” between testers and developers? f. Are there clearly defined forms of interaction between developers and testers (e.g. Jira, specific artifacts, defined time) ?
6) When you have a problem or a question during testing who helps you? a. When you have a doubt that relates to implementation, do you talk to the developer directly? How? b. How easy is it to test a feature that you have never tested before?
7) What “works” well in the testing setting you have?
8) What hinder work (bottlenecks) in testing?
9) Is there anything else you would like to add that you think is interesting in this context?

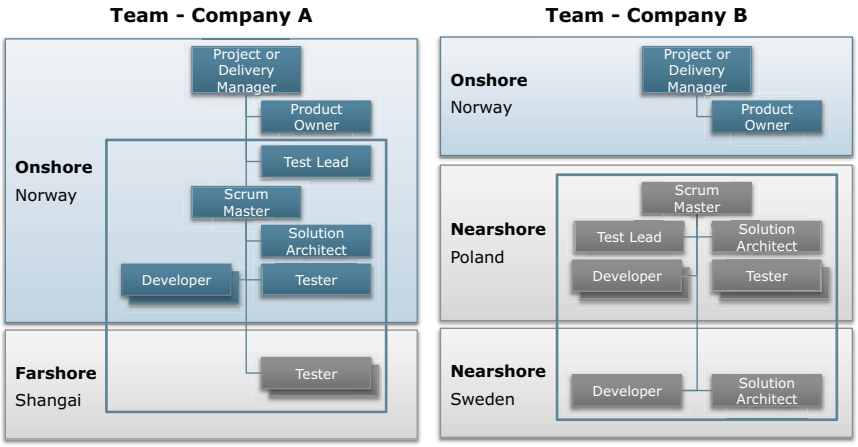

Fig. 2. Configuration of the Teams at Company A and Company B

The tester in Norway is a newly hired person fully dedicated to the team. At the time of the study, he was still trying to learn the system under development. In China, two testers with four and one year of experience on the product, work 100% on the team and one tester with six years of experience on the product is working 50%.

The testers in China are domain experts, but do not have background in software engineering. In Norway, the test manager is the one responsible for planning, communicating and organizing the test work between the team members in Norway and the testers in China. Working with China takes 40-50% of her time. The test manager is also working with the support department, and she acts as buffer between testers and developers. She is also engaged in the user acceptance testing with the customer.

Most of the testing in China is performed manually by following test scripts developed and maintained by the Chinese testers. When new functionality is to be implemented in the software, the team uses some time to ensure that everyone understands what is being developed, so that the test scripts can be updated. While test automation is seen as important in agile testing, it is not widely implemented in the testing activities. So far, the automatic user interface tests have a very short "life". The user interface is changed frequent and then the tests are broken. According to the testers, it takes too long to update the automatic tests. In addition, the ability to automate tests depends on the knowledge of the tester. The tester responsible for implementing automatic testing was on maternity leave at the time of investigation, which is one reason for why the level of automation was low.

Furthermore the Chinese testers are also responsible for integration testing and performance testing. The system integrates with several external systems (e.g. SAP), and when functionality is added or changed there is a need to run integration tests. Performance is important; however, performance testing is not needed for every release. When important modules, components or the database are changed, performance testing is scheduled. The team-lead, test manager and one architect take the decision regarding performance testing. The testers in Shanghai use designated tools when they are doing performance testing.

D. Company B

Company B is leading supplier of intelligent transportation systems to the public transport sector, including fare collection, travel information, infotainment, fleet management and traffic management. The investigated team consists of 9 members (8 in Poland one in Sweden) including 2 testers (in Poland), Product owners (POs) (in Norway) and customers mostly in Scandinavia. The team follows a Scrum-based process, where a release is about 6 months long. The testers are part of the team, allocated 100% for testing, and are also members of a team of testers in the organization, with a test manager in Poland. The test manager responsibility is to align the processes among teams in the organization. Also to identify good practices that can be spread in the organization. The test manager does not interfere in the daily work of the testers; he is also engaged in the user acceptance testing with the customer.

The testers have background in computer science and software engineering; therefore they can do development if needed. Domain knowledge was acquired during the work in the company. One of the two testers has eight years experience as a tester, and for two years has been working in this company. The second tester has four years experience as a tester and one year in the company.

The team follows a continuous testing software process. In the Sprint Planning, testers and developers estimates the duration of the testing for each user story planned to be implemented in the sprint. During the sprint, when the user story is implemented, the developer performs unit tests and integration tests. In some new areas the developers write automated unit tests, but legacy code is not yet covered by any automated script. When a developer is done with a task he or she changes the status of the task to "ready for testing". The tester checks everyday the ready for testing user stories and test them. The testers are part of the daily standup meeting, where project progress and challenges are discussed, and issues related to implementation are solved. Testers use the information shared during the standups to start planning the test cases and also to start thinking about which testing environments needs to be created for when specific user stories are implemented. Developers usually don't do regression tests or user acceptance tests. Testers perform regression tests based on risk areas he defines

by viewing the code changes and the user stories. Close to the release, the testers run the user acceptance tests (UATs) defined by the customers. Deployment and deployment tests are to be conducted by the Operations team.

While test automation is seen as important in agile testing, it is not widely implemented in the testing activities. Some developers have started to focus on automated unit tests. And testers have also shown interest in getting more involved in implementation tasks. Furthermore the testers are also responsible for integration testing and regression testing.

IV. Results

During the data analysis, we identified four main themes coming from the interviews with developers, scrum master, testers and test managers. The codes from the sources of information were organized into four main themes: handover through issue tracker system, formal meetings, written communication and coordination by mutual adjustment.

A. Handover through Issue Tracker System

Handover in this context means, the passing of control authority of a user story from developers to testers, most of the times after a code review. After the handover the user story goes to the state of Ready for Testing. Both teams investigated relied on JIRA (issue tracking system) when handing over the user story from developers to testers. In both companies, the testers affirmed that the communication through JIRA is not sufficient, as they still have to ask many questions. In company B, this is not much of a problem as the developers and testers have direct access to each other and are able to talk directly to each other when there is a need for clarifications. In Company A, the testers often do not have direct access to the developers, for two main reasons, the temporal distance and the role of the test leader as the broker between the testers in China and developers in Norway. In Company A one tester said explained the need for communication with the developer: *"Sometimes the developers have very limited time to code, then the description of the user stories not always get detailed information, we need to communicate with the developer directly because sometimes they just leave the main direction of the new feature, without details. Sometimes we can get some requirements documents from Sharepoint or from somewhere. Some times the requirements are not easy to understand because business people write them. So the main source for us to understand the issue is from JIRA"*

When we asked the testers in Company A on how often the descriptions are easy to understand, they answered that about 30% of the cases. In other cases, the tester from company A explained why handover through the issue tracker is not enough for doing their job: *" It takes a long time to understand the new feature. We are not involved in the requirement specification and we miss lots of information. It may be helpful that when a big new feature is introduced; that the developers could give a small demo to understand how it works. Sometimes it is not ready for the testing, because other features needs also to be implemented before it is possible to test but we don't get this feedback."*

The test leader in Company A is aware of the challenge related to handover in JIRA. She has found out that if the descriptions in JIRA are long and detailed, testers will ask less questions and misunderstandings will be found late. This is one reason for why some developers write short descriptions to ensure that the testers ask questions. The test leader in Norway always encourages the testers in China to talk to her, scrum master and developers in Norway. The test leader in company A explained*: "Every day Testers check new functionality created in JIRA and plan what test that need to be run. However some developers write a lot and some developers write very little. It seems to be more efficient to have short descriptions, then feedback and then give them answers than using a lot of time documenting. So them asking is important. We repeat it in every meeting (video). It is better to have a culture where people ask questions, than a culture where people use a lot of time trying to write good documentation. You should not try to standardize the way of writing in JIRA; you should continuously focus on reminding people to ask questions*.

However, asking questions and getting fast feedback was challenging because of the time difference and developers not being as accessible as needed. Specially close to a release. Even though the Chinese testers were encouraged to ask questions, the testers had the impression that the developers are always busy and don't want to be disturbed.

While having the testers in another time zone had the potential of testing while the developers were sleeping, the time-zone differences also introduced some challenges. When a tester was missing important information about the new feature to test and can´t get immediate feedback from a developer because he or she is busy, or has ended the working day, there is a delay in the process. Then a tester might end up wasting time trying to find documentation for that feature (that might be incomplete or missing), or waiting to the next day before continuing on testing the new feature. The testers also voiced that one possible way to help mitigate this problem is to join the planning meeting before release. But they have problems with this approach because of the temporal distance between their center and the developers' team in Norway.

In Company B, the developers and testers are mostly co-located with a developer in Sweden, and rely on informal communication to solve the challenges with handover in the issue tracker system. When talking about the need for information to perform the job, using JIRA descriptions and informal communication, one tester in company B said: *"I can say that it is enough for me because here we are in open space and we talk a lot about details together with the developers and if there is something unclear we discuss and if it is still not clear we send an email to the PO, and it gets clarified."*

In Company B, we asked one developer if when developing a new feature if he had the awareness to change the descriptions on JIRA or changing the handover approach for a specific user story. The developer answered: *It is not a formal approach. But yes. I even differentiate if it is Tester 1 or Tester 2 that is doing the testing. Then I write different testing procedure or description of what the implementation is about. Like Tester 1 knows the system so well that he will immediately see that he needs more information about the description. I think I know what they know more or less. We have very rudimentary documentation on the issues and it can be improved*."

B. Formal Meetings

Formal meeting as planning meetings, daily meetings, retrospectives, weekly meetings with the test leader are one of the venues where testers and developers have an open forum to communicate with each other. These meetings are important for the testers to participate as they use these to be updated about what is happening in the team, listen and participate of the discussion of the solutions, *have inputs for the priorization of Bugs* and give some feedback to the developers. Both companies make use of these meetings regularly, but testers in Company A are not able to participate in most of them, because of the time zone differences. One tester in Company A said: *"Our leaders prioritizes the bugs, and sometimes, if they do not agree in the prioritization, then they have a joint discussion. One of our leaders has a very good overview of what happens in support and in production; therefore she got a good overview of the business. The other one knows the technical system very good and the business requirement."*

In Company A, sometimes the testers are left out of some decisions, for example about the decision of running or not some tests for each release, the test leader said: *"By each release there is a discussion if there is a need to do performance testing. Sometimes important modules, components (that affect performance) or the DB is changed, and then performance testing is scheduled."* The test leaders are the ones taking the decision in Norway. When there are no important changes, performance testing is not done. Shanghai is doing this testing and they use a tool for it."

In Company B the testers participate in the daily standup and also participate in the sprint planning, reviews and demos, they also use this formal meeting to communicate to the developers some feedback on different issues, for example feedback on some user friendliness issue: "*Actually I have a pretty good contact with the developers and this standup meeting clarify a lot. And I get all information I need in the morning. Then I am always informed on changes that occur."*

C. Written Communication

Written communication is noticeable more used in Company A than in Company B. The reason is because the testers and developers are not in the same location. One tester in Company B commented: *"When the tasks are not simple I have to ask afterwards, I know if they are busy or if they have meetings and I can decide when to ask questions. Sometimes if I see that someone is busy, I just send an email, or use communicator. I know that it is not good to get interrupted all the time.* One tester in Company A said: *We usually use E-mail with architect, Lync with Test Leader. We communicate more with the Test Leader; she is coordinator. So, we talk more with her. And we send mail when we have a long list of questions. The email is usually answered in the next day. Talking on Lync is easier for us than writing".*

On the test reports, containing the results of tests, which were executed during the testing phase, in Company B one developer said: *"Mainly we use informal communication and it is good enough. Sometimes I just ask to wait for discussing the issue later. When I have to ask to wait, then we create a JIRA issue and I take care of it later. But sometimes after that I can have two hours of talking about bugs.* Another developer in Company B said: *I only get reports on what is related to me. And actually sometimes I answer with. Hey… This is not a bug… Sometimes it is a configuration problem. It is a matter of sharing the results. I don't get a report of the structured report on the bugs found on the test. Like, after a week, we could get some report on, like this week, we ran xxx number of tests and XXX passed or XXX failed. Currently we get more bad news than good news from testers."*

The Swedish developer in Company B voiced that one way to the testers to have a better understanding of the user stories can be to have unit tests always as a deliverable from the development phase, because it would be much easier for the tester to understand what the code is supposed to do. Another developer said*: "We have a JIRA field that we are supposed to fill with the testing procedure. This should be done by the developer, but sometimes because of the pressure of getting other work done so we don't fill this testing procedure, and then the tester have to fill in this field."*

D. Coordination by mutual adjustment in the virtual team

Agile teams are encouraged mutual adjustment over documentation, which enables coordination by informal communication. In both companies testing is performed continuously during development and stabilizing. In both teams, in the development phase a developer picks tasks in JIRA. The task is implemented and the developers run unit tests. When the task is completed, JIRA is updated. Testers can then start doing their job. Some developers write a very detailed JIRA description while others write only a very short text, which makes it challenging for the testers to understand how to run the test.

In company A, when testers find bugs or when they identify questions they need to ask their remote colleagues in Norway, they first discuss among themselves to understand if it is a real bug and if they can solve the issue. Even though the testers and developers have been on the product for a long time, the testers sometimes find bugs that the Norwegian developers do not perceive as a bug. Then the Norwegian team-lead (who knows the technology and the technical system), and the test manager (who knows what happens in support and production), look at the test-report and discuss with the Chinese. If the Chinese testers and the developers agree that it is not a bug, then the Chinese update the test scripts accordingly. Having such discussions and aiming at achieving a shared understanding enable shared mental models in the team and strengthens the autonomy of the Chinese testers.

The varying levels of details in the specifications are not seen as a problem by the team management in Company A. But it is seen as a problem by the testers, once that to coordinate work, there is always a need to communicate with the test leader. The testers in China asked for better documentation from the developers. One reason for that could be that they do not understand how they can talk more to the developers because they seem to be so busy and there is time difference. They are encouraged to talk to them, but get the feeling it is not really appreciated, at least by some. Then is a problem that they get two different signals

V. Discussion

A. How do developers and testers communicate in distributed continuous agile testing?

As shown in the results section, in the teams we investigated, testers and developers have mainly contact through the user stories, testing of the user stories is reason for them to communicate. Therefore, one of the main topics discussed when we talk about interaction between testers and developers in a distributed agile testing context is the handover between testers and developers. But our results show communication through JIRA/user stories showed to be not enough in both teams and they had to find turnarounds to avoid misunderstandings. Informal communication also plays a very important role on the communication and teams needs to find ways to keep this channel open. The cases shown in this paper shows how direct communication facilitates the daily work of testers and consequently developers. There is though little research on handover and knowledge transfer across roles in a software team [28].

Another main area of discussion was the participation on the meetings. Testers appreciate to participate in the meetings and get information about plans and daily changes and discussions in the project. As also pointed by Melnik et al. [19], knowledge sharing based on recurring practices such as agile stand-up meetings and socialization of team members enable agility and oppose communication errors on the team level. This is also confirmed in our study; because the testers are not able to be part of the main meetings of the development team, they miss a lot of information that would make their job faster and more effective.

In the case where the testers and developers are distributed, there is a higher need to also use other communication media such as email and Lync. Emails makes the communication more asynchronous and the testers many times have to wait for one or more days to get the answer to their questions. It was also pointed out by the testers in Company A (distributed) that they would prefer to not have to write an email in most of the cases, so they try to find the answer sometimes in some other documents and they end up using more time than it would be needed if they could just ask a developer. Lateral communication is very also important for enabling coordination through mutual adjustment between testers and developers and we could see that the usage of a broker between testers and developers created distances between them. On creating norms for the use of JIRA for the handover between testers and developers, it should be taken in consideration both testers and developers needs for information and how these norms will affect the daily job of testers and developers.

Relative to more traditional settings, communication processes that occur in agile testing are expected to be rapid, customized, temporary, greater in volume, more informal, and more relationship-based [10]. Hummel et. al. [15] describes six social agile practices and their impact on communication, and found that developers strongly preferred direct over indirect communication, these results are also confirmed by our study. They also found that a compromise between formal documentation and informal communication was necessary because both extremes were neither feasible nor desirable in real-life projects.

B. How is communication impacted by types of team configuration between testers and developers in distributed continuous agile testing?

Team distribution refers to the extent to which team members of the software development projects are separated from each other (distributed) or close to each other (collocated) in terms of the physical location [36]. Therefore, team distribution influences how, when and how frequent team members communicate [15][20][30]. The amount of verbal communication is lower in distributed projects compared to collocated teams, which may result in miscommunication, misunderstandings and confusion among team members. In Company A (distributed developers and testers), the team leader has to constantly remind the testers to ask questions to get clarifications. In Company B (collocated developers and testers), testers and developers just organically talk and get clarifications.

Temporal distance showed to be a major impediment for closer collaboration in Company A (distributed). This result also confirms the studies in communication in other contexts, also evidenced by Hummel, co-location enables quick and efficient direct communication due to close proximity; and, enables to clarify questions face-to-face, besides, the daily meetings, enables high frequency and high amount of direct communication in and after the meetings for co-located teams [15]. Improving collaboration infrastructure among distributed teams doesn't really solve the problem. It is important that team members are able to contact their remote colleagues at any time, without need to write an email or wait until the next meeting with the team lead. The collaboration among remote members is highly important for an agile team to carry out work together in a two-week iteration. In hectic periods there is a need to extend and shift working hours to be more flexible which is a cost for some of the team members. In addition, being in a different time zone makes it difficult for the testers to be part of the daily activities of the teams. Sometimes the testers miss knowledge on how the system is supposed to work and on what is expected when it comes to quality.

Company A (distributed) experienced some time zone benefits as well. For them it is beneficial for the team that testing happens while developers are sleeping, because it enables continuous testing and integrates testing activities as closely as possible with coding. Firstly, errors can be fixed quickly while the context is fresh in the developers' minds and before these errors lead to knock-on effects. Secondly, the underlying root causes that led to the problems may be more easily identified and eliminated. However, they often lost a day because of the need to clarity issues related to the feature under test. Another benefit of having remote tester was that it might be beneficial to the team that testers do not get too close to the developers which make them report what they perceive as bugs, not what the developers want them to report.

On the participation of the regular meetings, Company A (distributed) had much more problems in keeping the testers informed regularly about daily changes in the project. Several empirical studies have shown that engaging testers earlier and throughout the software development process is beneficial to a project team's performance [13]. To continue improving the testing and development activities, the team from Company A

plans to invest more in the initial phase of each release. The goal is to make sure that everyone in the virtual team has a shared mental model of what will be delivered for the next release. The testers need to be more involved in the planning phase to get a better understanding of the goal of the iteration. Organizing early demos for the testers might be one option. Furthermore, the degree of formality of the communication was bigger in distributed than in collocated teams. This result is also aligned with previous results from studies on communication in agile systems development [15].

Regarding the quadrants of testing, the testers that are far from the developers also get further from the business environment and it is harder for them to be able to perform tasks related to the "business facing", like criticizing the product in a customer perspective, in talks with the testers they mentioned how they get further from the business and business view, and it ends up that their tests are much more limited to "correction" than to "critiquing" the product. Besides, the tests that are performed by each role (testers and developers) are much more of a black-box when they are so far apart. Hindering more the possibilities for collaboration between roles. The lack of transparency of the activities in both sides also limits the possibilities for the testers to discuss how they can support the team on "building" the product, especially when the testers are not even able to attend the meetings in the team.

C. Implications for research and practice

Results from this research can aid team leaders to see their team's collaboration practices (i.e. communication via handover, chat systems, written or informal communication), trends (i.e. coordination by mutual adjustment) and flaws (i.e. missing or indirect communication, lack of participation in meetings) between testers and developers. The team can keep a constant check on reduced or increased communication among members through repeated checks. As a result, the leaders can take necessary action to orient the teams in a better way to incorporate more communication strategies among roles in the team for better performance.

To help bridge this gap, communication strategies can be used. Tarone [31] defines Communication Strategies (CS) as a mutual attempt of two interlocutors to agree on a meaning in situations where requisite meaning structures do not seem to be shared. Therefore, it can be viewed as attempts to bridge the gap between interlocutors in real communication situations. Four facets of communication strategies explained by Mohr and Nevins [23]: frequency, direction, modality and content. In the contexts we evaluated we see that teams can benefit on being aware of these concepts to get the communication to be more effective. However, it is challenging for developers to balance the completeness, conciseness and clarity of the content of the information that they need to share with the testers. Writing well enough specifications seems to be particular challenging because such specifications will require that the developers spend a lot of time doing it, and some developers are lacking skills in writing in another language. We found that both teams seem to find oral communication to be the best way to convey the information. On the consideration, teams needs to be able to know more about the competences and skills (who know what) so developers can prepare every handover with the specific tester in mind and try to understand what that specific tester would need of information. . It is important for an organization to decrease the knowledge and experience gap between its software developers and testers and create a working environment in which developers and testers trust each other and maintain a good relationship and have an open channel for informal communication. To build trust between developers and testers, an organization may also engage developers and testers in social activities outside work, besides thoroughly participation of both roles on the formal meetings of the teamwork.

D. Limitations

The research reported here has limitations. First, our study was limited to members of two companies and two projects. The two projects had different characteristics, helping to overcome some issues with generalizability of our results. Within the projects, we were careful to interview members playing different roles on the team, so we could obtain different perspectives of communication between developers and testers.

Another limitation of our study is that we relied mainly upon interviews and observations to derive our results. The lack of standardization that it implies inevitably raises concerns about reliability. To reduce some of these concerns, we discussed and improved the interview protocol iteratively before data collection. One researcher also visited the companies many times to better understand their context and observe the teams. The interviews were also conducted in different moments of the projects. For the analysis, two researchers coded all interviews individually and in a second step, they discussed each coding before including it in the software tool. One reliability issue that we did not overcome was the triangulation of the results with multiple methods of data collection.

VI. CONCLUSION

In this paper we have presented the results from a study on how developers and testers communicate in a distributed continuous agile testing environment. The testers and developers make use of different forms of communication, we describe four main themes: handover through issue tracker system, formal meetings, written communication and coordination by mutual adjustment. We have found that early participation of the testers is very important to the success of the handover between testers and developers and this is accomplished through participation in planning, standups and review meetings. The communication between developers and testers is not sufficiently effective through written communication and it has to be augmented by informal communication. One important observation is that the communication changes depending on the type of the tasks (new or old) and also on the experience of the testers and how much the developer knows about the experience of the tester.

Regarding the distribution or collocation between testers and developers, it is reasonable to conclude that distribution does affect the work of testing by creating more asynchronous communication and consequently less informal communication between developers and testers. The trade-off between "good documentation" vs. "asking questions" strategies is still an open question and needs to be better investigated to understand the real effects of it.

Temporal distance showed to be a major impediment for closer collaboration between testers and developers and participation of the testers on formal meetings of the project, such as planning, daily meetings and retrospectives. Some mitigation were also planned by the leaders after the results of this investigation, for example adding a video tapping of the daily meeting and planning meetings, so the Chinese testers could watch them afterwards.

The results of this study also indicate a number of directions for future research. There are a number of models of team effectiveness, originating from several disciplines of science [11]. In particular, teamwork has been a much-researched topic in management science and in psychology. Although the models use different terminology, there a number of similarities in the mechanisms included. Communication is central in all models; and, as with other areas of software development research, we also need more empirical studies and better theoretical grounding in studies of software teams and software team effectiveness. Our next goal is to apply a theoretical framework of team effectiveness in the understanding of how to team up testers and developers effectively. .

ACKNOWLEDGMENT

This work was supported by the Smiglo project, partly funded by the Research Council of Norway under the grant 235359/O30.

REFERENCES

[1] Y. I. Alzoubi, A. Q. Gill, A. Al-Ani, "Empirical studies of geographically distributed agile development communication challenges: A systematic review", Information & Management, August 2015, ISSN 0378-7206.

[2] N.-D. Anh, D.S. Cruzes, and R. Conradi, "The impact of global dispersion on coordination, team performance and software quality - A systematic literature review," Information and Software Technology, 2015, pp. 277-294.

[3] S. Barney, V. Mohankumar, P. Chatzipetrou, A. Aurum, C. Wohlin, and L. Angelis. "Software quality across borders: Three case studies on company internal alignment" *Inf. Softw. Technol.* 56, Jan. 2014, 20-38.

[4] E. Bjarnason, P. Runeson, M. Borg, M. Unterkalmsteiner, E. Engström, B. Regnell, G. Sabaliauskaite, A. Loconsole, T. Gorschek, R. Feldt: "Challenges and practices in aligning requirements with verification and validation: a case study of six companies." Empirical Software Engineering 19(6): 1809-1855 (2014)

[5] R. E. Boyatzis, "Transforming qualitative information: thematic analysis and code development". Sage Publications, 1998.

[6] E. Collins, G. Macedo, N. Maia, and A. Dias-Neto, "An Industrial Experience on the Application of Distributed Testing in an Agile Software Development Environment," IEEE Seventh International Conf. on Global Software Engineering (ICGSE), 2012,, pp. 190-194.

[7] L. Crispin and J. Gregory, "Agile Testing: A Practical Guide for Testers and Agile Teams", Addison-Wesley Professional, 2009.

[8] D. S. Cruzes, T. Dybå: "Recommended Steps for Thematic Synthesis in Software Engineering." ESEM 2011: 275-284

[9] R.M. Davison, M.G. Martinsons, N. Kock, "Principles of canonical action research", Information Systems Journal 14 (2004) 65–86.

[10] G. DeSanctis and P. Monge, "Communication Processes for Virtual Organizations." J. of Computer-Mediated Communication, 3: 0, 1998.

[11] T. Dingsøyr, T. Dybå: "Team effectiveness in software development: Human and cooperative aspects in team effectiveness models and priorities for future studies." CHASE 2012: 27-29

[12] B. Fitzgerald and K.-J. Stol , "Continuous software engineering: A roadmap and agenda." JSS. To be published in 2016.

[13] M. L. Gillenson, M. J. Racer, S. M. Richardson, and X. Zhang, "Engaging Testers Early and Throughout the Software Development Process: Six Models and a Simulation Study". Journal of Information Technology Management, 22(1), 8-27, 2011.

[14] G. K. Hanssen, D. Smite and N. B. Moe : "Signs of Agile Trends in Global Software Engineering Research: A Tertiary Study." Sixth IEEE International Conf. on Global Software Engineering (ICGSEW), 2011.

[15] M, Hummel, C. Rosenkranz, and R. Holten. "The Role of Communication in Agile Systems Development: An analysis of the state of the art", Business and Inf. Systems Engineering 5(5): 343–355, 2003.

[16] J. Lee, S. Kang, D. Lee. : "Survey on software testing practices", IET Softw., Vol. 6, Iss. 3, pp. 275–282, 2012.

[17] S. Marczak, D. Damian: "How interaction between roles shapes the communication structure in requirements-driven collaboration". RE 2011: 47-56

[18] L. Mathiassen: "Collaborative Practice Research" Scandinavian Journal of Information Systems: Vol. 14: Iss. 1, Article 5, 2002.

[19] G. Melnik, G. and F. Maurer.: "Direct Verbal Communication as a Catalyst of Agile Knowledge Sharing", Proceedings of the Agile Development Conference (pp. 21-31). Washington, DC, USA:, 2004.

[20] C. de O. Melo, D. S. Cruzes, F. Kon, R. Conradi: "Agile Team Perceptions of Productivity Factors". AGILE 2011: 57-66.

[21] N. B. Moe, D. Cruzes, T. Dybå, E. M. Mikkelsen: "Continuous Software Testing in a Globally Distributed Project". ICGSE 2015: 130-134

[22] N. B. Moe, D. Smite, G. K. Hanssen, and H. Barney, "From offshore outsourcing to insourcing and partnerships: four failed outsourcing attempts," Empirical Software Engineering, 2013, pp. 1-34.

[23] J. Mohr and J. R. Nevin, "Communication strategies in marketing channels: A theoretical perspective," The Journal of Marketing , vol. 54, no. 4, pp. 36–51, 1990.

[24] J. Gerring, "Case Study Research: Principles and Practices". Cambridge University Press, 2006.

[25] H. Shah, M.J. Harrold, and S. Sinha, "Global software testing under deadline pressure: Vendor-side experiences," Information and Software Technology, 2014, pp. 6-19.

[26] D. Šmite, N. B. Moe and P. J. Ågerfalk: "Agility Across Time and Space: Implementing Agile Methods in Global Software Projects". Berlin, Heidelberg, Springer-Verlag, 2010.

[27] D. Šmite, C. Wohlin, T. Gorschek, and R. Feldt, "Empirical evidence in global software engineering: a systematic review," Empirical Software Engineering, 2010, pp. 91-118.

[28] C. J. Stettina, E. Kroon,: "Is there an agile handover? an empirical study of documentation and project handover practices across agile software teams". In: 19th ICE & IEEE-ITMC International Conference, The Hague, Netherlands (2013).

[29] O. Taipale and K. Smolander: "Improving software testing by observing practice". ISESE 2006: 262-271.

[30] D. Talby, O. Hazzan, Y. Dubinsky, A. Keren: "Agile Software Testing in a Large-Scale Project". IEEE Software 23(4): 30-37 (2006).

[31] E. Tarone, "Some thoughts on the notion of communication strategy*," TESOL quarterly, vol. 15, no. 3, pp. 285–295, 1981

[32] I. Tervonen, A. Haapalahti, L. Harjumaa, J. Similä: "Outsourcing Software Testing: A Case Study in the Oulu Area". QSIC 2013: 65-74.

[33] E. J. Uusitalo, M. Komassi, M. Kauppinen, A. M. Davis: "Linking Requirements and Testing in Practice". In: Proceedings of 16th IEEE International Requirements Engineering Conference, pp. 295-302, 2008.

[34] J. Verner, J. Sampson, V. Tosic, N. Bakar, and B. Kitchenham, "Guidelines for industrially-based multiple case studies in software engineering," in Third International Conference on Research Challenges in Information Science, 2009. RCIS 2009., april 2009, pp. 313 –324.

[35] X. Zhang, T. F. Stafford , J. S. Dhaliwal, M. L. Gillenson, and G. Moeller: "Sources of Conflict between Developers and Testers in Software Development". Inf. & Management, 51(1), 13-26, 2014.

[36] P. J. Ågerfalk, B. Fitzgerald, S. A. Slaughter : "Flexible and distributed information systems development: state of the art and research challenges". Inf Syst Res 20(3):317– 328, 2009.