2015 Agile Conference

Stakeholder Perceptions of the Adoption of Continuous Integration – A Case Study

Eero Laukkanen
Department of Computer Science
Aalto University, Finland
Email: eero.laukkanen@aalto.fi

Maria Paasivaara
Department of Computer Science
Aalto University, Finland
Email: maria.paasivaara@aalto.fi

Teemu Arvonen
Ericsson R&D Center Finland
Jorvas, Finland
Email: teemu.arvonen@ericsson.com

***Abstract*—Continuous integration is an important support mechanism for fast delivery of new features. However, its adoption in industry has often been problematic, partly due to social challenges. However, there is little knowledge of the exact nature of the challenges, and how different stakeholders perceive the need for and adoption of continuous integration. In this paper, we describe how the introduction of continuous integration was perceived by different stakeholders in a R&D program at Ericsson. The case provided a rare opportunity to study the adoption of continuous integration in a large distributed organization. We interviewed 27 stakeholders and found differing perceptions of continuous integration: how suitable it is for the organization, how adoption should be organized, and whether it is possible to achieve sufficient quality through automated testing. These differences of perception were mainly consequences of the geographic distribution. Based on the case study, we propose three guidelines. First, understand that the product architecture has a significant effect on the adoption. However, do not let architectural problems keep you from implementing continuous integration. Second, give the team members sufficient time to overcome the initial learning phase in the adoption. Third, avoid centralizing competencies to individual sites, and invest in cross-site communication.**

I. Introduction

Increased competition requires companies to deliver changes to their products at a faster pace than before. More frequent releases have been perceived to shorten the feedback loop from the customers, decrease time-to-market and improve customer satisfaction [1]. However, the rapid release model puts more pressure on the quality assurance procedures: less time can be used for manual testing [2], instead the release process should be automated and optimized [3].

Continuous delivery (CD) is a software development practice in which the software is kept in a state that, in principle, it could be always released to its users [4]. While instructions on how to adopt CD have existed for some time [4], the industry is not practicing it widely yet [5] and those who have taken steps towards CD have found it challenging [1], [6], [7]. This raises a question whether the industry is simply just lagging in adopting CD, or whether unknown or unresolved challenges that effectively hinder its wide spread adoption exist.

In this paper, we describe how Ericsson adopted continuous integration (CI), the predecessor and a requirement for CD [4], in a new R&D product development program developing a XaaS [8] platform and a set of services. The aim of the development organization is to be able to deliver new functionality continuously. Furthermore, we studied how different stakeholders in the case organization perceived the adoption of CI. We reported the first steps of the agile transformation of same case organization in our previous papers [9], [10].

This paper is structured as follows. First, we introduce the concept of continuous integration and research related to it in Section II. Section III introduces our case organization and research methods. In Section IV we present our results. Finally, we discuss the results and the limitations of the study in Section V, and draw conclusions in Section VI.

II. Related Work

A. Continuous Integration

Continuous integration (CI) is a software development practice focusing on frequent integration of changes to the software during development [11]. CI aims to avoid the problems caused by a separate integration phase in the software process: unpredictability and large integration effort [11]. Practicing CI means that each developer integrates changes daily and that each integration is verified by an automated build and tests [11].

In practice, adopting CI fully has been troublesome. While some cases have achieved a smooth CI practice [12], others have not been fully satisfied with the implementation of CI [12], [13] or have found adopting CI challenging [6]. It has been reported that a functional CI practice requires sufficient levels of discipline [14], collaboration [15] and communication [15]. An empirical investigation of CI practices in industrial cases has revealed that the practices do not adhere to the definition of CI fully, but are somewhere between CI and a separate integration phase [5]. The exact causes for the lack of CI adoption are not yet fully known.

B. Previous Studies

Debbiche et al. [6] and Claps et al. [7] both studied the challenges companies face when adopting continuous activities, Debbiche et al. studied continuous integration and Claps et al. continuous deployment. Both found that the adoption is hindered by social and technical challenges. In both papers the adoption was studied at the company level, meaning that results from different products were handled as a whole. While company-level understanding of the adoption is important too, it would be interesting, especially in large-scale software development, to concentrate on a single product only, as the challenges seem to relate to product-specific constraints [6].

978-1-4673-7153-7/15 $31.00 © 2015 IEEE
DOI 10.1109/Agile.2015.15

Table I. GLOBAL DISTRIBUTION OF THE CASE ORGANIZATION.

Site	Country	Role	# of Teams Interviewed
Site A	Country A	Development	7
Site B	Country B	Development	3
Site C	Country B	Development	1
Site D	Country C	Operations & Support	0

This way, the challenges can be understood in the context they appear.

Ståhl and Bosch [5], [13], [16] studied the benefits and implementation of CI practices. They noticed that the benefits and the implementations of CI differ case by case. Thus, studying CI seems to require deep understanding of the case, which justifies the usage of case study research strategy.

Leppänen et al. [1] studied the state-of-the-art of continuous deployment in 15 Finnish software companies. They found that different companies have different goals considering continuous deployment based on the business constraints imposed to them. Furthermore, they found similar challenges as Debbiche et al. [6] and Claps et al. [7] that emerge when adopting continuous deployment.

Eck et al. [17] studied in their literature review how organizations assimilate CI practice. They found that organizations need to implement various changes when adopting CI. In addition, these changes are required in different stages of adoption. However, it is questionable whether the results apply to all organizations and to all the products inside one organization uniformly.

As a summary, CI adoption seems to be a complex task, and what it actually means depends on the case at hand. To better understand the challenges when adopting CI we have decided to study a large-scale software product organization in depth.

III. METHOD

In this section, we give a brief description of the case organization, introduce our research goal and questions and finally present our methods for data collection, analysis and validation of the results.

A. Case Organization

This study is based on a single case study [18] of an Ericsson software development organization developing a XaaS platform and a related set of services. The product provides a set of services to the business customers, who then in turn can provide various services to their clients. Architecturally, the product consists of components, which are physically on different machines in the production environment. The product is divided to two systems of components based on the functionality: the business system and the network system. Some of these components are developed by third-parties and some by Ericsson. The development of the components requires highly specialized expertise due to their complexity. The development organization of the product is currently distributed across three sites in two countries (Table I). In addition, an operations organization is located at a fourth site in a third country.

Ericsson has traditionally used a plan-driven software process. However, during recent years the company has started a global adoption of lean and agile software development. Regarding this product, the development organization decided in August 2011 to undertake such a transition, or “start the agile and lean journey”, as the organization puts it. While always challenging, this transition has been particularly challenging, as the organizational growth has been rapid during the transformation.

Table II. INTERVIEWEES AND THEIR ROLES.

Role	Site A	Site B	Site C	Total
Team members	12	4	3	19
Product owners	2	1	1	4
Others	1	2	1	4
Total	15	7	5	27

After the initial agile transformation steps, the case organization was required to pay more attention to agile testing. Originally, all the high-level test cases of the product were executed manually before each release. These manual tests were executed for each component individually, but the product as a whole was not tested at all. This manual testing effort was mainly driven by the Site B, where the case organization was originally located. Because of the manual testing effort and the lack of holistic end-to-end tests, the case organization on Site A started to put more focus on test automation and continuous integration in Fall 2013. This paper investigates how the adoption continued after that time.

B. Research Goal and Questions

Our main research goal was to investigate how different stakeholders perceived the adoption of continuous integration in this globally distributed organization. We purposefully selected this case, as we had the possibility to gain access to the case based upon participation in a joint research program, and we had previously studied the first steps of the lean and agile transformation in the same case (see [9], [10]).

We posed the following research questions:

- **RQ1: What actions were done to adopt continuous integration?** We asked this question to understand the current perceptions of the adoption.
- **RQ2: What challenges were faced when adopting continuous integration?** We asked this question to understand what kind of challenges had occurred during the adoption.
- **RQ3: How was the adoption of continuous integration perceived by different stakeholders?** We asked this question to understand how the perceptions differed.
- **RQ4: How would different stakeholders proceed with the adoption of continuous integration?** We asked this question to understand differing perceptions on the next steps of the adoption.

C. Data Collection

The data collection took place between December 2014 and March 2015. We interviewed 27 stakeholders with an interview guide approach [19]. The roles and sites of the interviewees can be found from Table II.

The roles of the interviewed persons included team members, product owners, coaches and managers. We aimed to interview a broad representation of the development teams in the organization, talking to informants in different roles, from different teams and with various backgrounds in order to gain as complete a view of the situation as possible. 1-3 members from each of the 11 development teams were included in the interviews. All interviewees were selected with the help of the case organization representatives.

The interviews typically lasted one hour, but the length ranged from half an hour to two hours. Especially for the first three interviews, we reserved more time, as we asked more background questions in order to understand the current situation of the adoption. These early interviewees were managers and key team members, who had a broader overview of the organization and had actively participated in the implementation of the CI system. The subsequent interviews were focused on the views of CI and were somewhat shorter. All interviews were conducted face to face in the organization's facilities.

First three interviewees were interviewed with an initial interview guide to elicit background information about the situation in hand. After these interviews, a more specific interview guide was constructed for the rest of the interviews. Since many interviewees had in-depth knowledge on specific areas, those areas were discussed more in depth with these interviewees. All sites participating in product development were included in the interviews. Only the operations site was decided to be left out of the scope of this study, as they were not actively involved with the CI.

D. Data Analysis

All interviews were voice recorded and later transcribed by a professional transcription company. The transcriptions and the interview notes were first read through by the first author to get immersed with the data. Second, relevant quotes from the transcriptions were collected according to the research questions. The quotes were of four types:

- **Past Events and Actions** to understand what happened during the adoption.
- **Perceived Challenges** to understand what had been difficult during the adoption to different stakeholders.
- **Perceptions of CI Adoption** to understand what was the interviewee's view of the CI adoption.
- **Opinions on How to Proceed** to understand how different stakeholders would like to proceed with the adoption.

These quotes were inspected and used to form the narrative results in this paper.

E. Validation

To improve the validity of study, we used several types of triangulation [18]: we collected data from several subjects and by several researchers. We interviewed a large number of subjects from different roles, with different backgrounds, from various sites, and with differing length of experience in the organization to get as broad view as possible. Also, the data was collected mostly by two researchers; 20 out of 27 interviews were conducted by two interviewers. Both researchers participated in data analysis and writing. The final version of this paper was reviewed by the third author, a representative of the case organization.

Table III. TIMELINE OF THE EVENTS DURING CONTINUOUS INTEGRATION ADOPTION.

Date	Event
June 2011	Platform acquired by Ericsson
October 2012	Introduction of agile practices
September 2013	Testing activities and test automation close to non-exist, two teams on Site A start to build the CI machinery and create legacy test cases
January 2014	Evaluation of test frameworks done by one team on Site A, one framework chosen for the product
March 2014	CI events organized at all sites to spread the mindset and give trainings for the chosen test framework
December 2014	*First interviews at Site A*
December 2014	50 % of legacy test cases for the business system created
February 2015	*Last interviews at Site A*
March 2015	*Interviews at Sites B and C*

IV. Results

In this section, we present the results of the study. The section is divided into four subsections, each answering one research question.

A. Timeline of CI Adoption (RQ1)

The case organization had originally already some initial CI systems for some teams. These CI systems included automated builds and continuous testing of the unit tests written by developers. However, what the organization was lacking was a unified CI system for the whole product, since these initial CI systems were mainly testing single components. In addition, there were no automated end-to-end tests, which would test the whole running product in a test environment. Formal release testing of the product was done manually for each component on Site B:

> *"This legacy, it's.. we have a huge technical debt, because imagine in the old time there's no automated cases. For many of the functionality, they left testing, tested manually by some of our testers, in Site B mainly."* — Site A employee

In September 2013, two teams were assigned to build the required CI infrastructure and create automated end-to-end test cases for the prevailing business system. These two teams also started to manage a CI community of practice (CI CoP), where decisions were discussed and made regarding CI practicalities.

During 2014, these two teams built the CI infrastructure and created numerous legacy test cases. In addition, the teams kept training sessions for the other teams on all the sites so that other teams building new features could create new test cases in parallel. However, despite the effort to train other teams, the creation of new test cases could not just be enforced, and some teams were not convinced of the benefits of automated test cases. Thus, the initial CI mindset was just inside the two teams who built the CI infrastructure:

> *"In these two teams we are, culturally speaking, much more advanced than many other teams already. I think there are teams*

that are not interested in the [test framework] at all."
— Site A employee

To spread this mindset further, the management of the product started to move key persons from these two CI teams to the teams developing the product. This action had led to desired consequences and other teams had started to create new automated test cases in conjunction with new features. However, even during the interviews, some parts of the product organization were not creating new automated test cases.

Since there was no general mindset for practicing CI yet, the responsibility to watch the builds in the CI system was on the two teams. To distribute this monitoring effort and also spread knowledge on the CI system, the organization started a "watchdog routine", a weekly rotating responsibility for teams to watch the continuous integration runs and investigate the causes for failing tests. This routine was just started when the interviews were done.

At the time of the interviews, the end-to-end business system CI contained two test runs: a "daily" test run which ran every two hours and a "nightly" test run which ran every night. The "daily" run contained only small amount of critical tests while the "nighty" test run contained a large amount of tests. There were still some tests which were not run continuously, because those tests created some amounts of data to the system that could not be removed. Therefore those tests were run more seldom manually.

Building and deploying different components was typically done automatically to the CI testing environment. Some components were exceptions, because they were updated more seldom and could not be deployed automatically. Therefore there was a specific time period during the week when these exceptional components could be manually deployed to the CI testing environment.

In addition to the end-to-end business system CI, some teams had their own CI systems. However, these systems were used by the individual teams only and they were not integrated to the centralized CI system. Furthermore, the development of these systems was not coordinated by the product organization, but the teams had created these systems when needed for single components.

The end-to-end CI system was still lacking the network system part of the product. In addition, the original release testing was still executed manually, although automatization of the release tests had started during the interviews due to release testing responsibility moving from the Site B to Site A where the CI mindset resided.

B. Challenges When Adopting Continuous Integration (RQ2)

In this section, we present seven challenges that the case organization had faced during the adoption (Table IV).

1) Lack of Time: Many interviewees told that learning new technologies for creating automated tests takes time that is not available for them.

The main reason for the lack of time was tight release schedule and pressure to create new features. The case organization had a periodic release schedule of eight weeks. According to the interviewees, the case organization prioritized the creation of new features higher than the adoption of CI:

Table IV. CHALLENGES AND SUGGESTIONS TO MITIGATE THEM.

Challenge	**Suggestions to mitigate**
Lack of Time	Better Communication, Division of Work, Support for Creating Automated Test Cases
Difficult Components	Change the Architecture
Unstable Tests	Change the Architecture
Slow Tests	Change the Architecture
Insufficient Testing Environments	Change the Architecture, Proper Testing Environments, Testing Strategy, More Frequent Smaller Releases
Agreement on Tools	Better Communication, Testing Strategy
Global Distribution of the Organization	Better Communication, Support for Creating Automated Test Cases

"I don't think we have the capacity for both in the development organization: to do this at the same time as we do everything else we have committed." — Site C employee

Also, we were told that when the release date came closer, corners were cut and testing was not done rigorously:

"Maybe that we sometimes get pushed to deliver fast, like we can skip the testing or skip some quality steps. It's more important to get the feature up to the customer than... I don't see that as a big problem, but we have some areas, where it kind of happens, that we get the features out without, automated test cases for example. Not enough coverage on the code. That might happen. I don't really see that as a big problem."
— Site C employee

The case organization had decided a single testing framework which should be used for creating the end-to-end tests. However, there was no initial competence for the framework, so team members creating the end-to-end tests needed to learn the framework and the style of the test cases from scratch. This, naturally, would have required more time than was available for most of the team members:

"I think it can, it's quite a lot of stuff that you need to get it working. Then to have your PC in an environment with the correct settings to make that work, it's.. sometimes hard to, come up with all that stuff. You need to have competence in a totally different area. I think it's, it will take just as much time basically as doing the feature." — Site B employee

In addition to learning, monitoring the build and fixing it would have taken more time. Because there was no time available for this task, build results were often ignored:

"The practice is good, but it takes time to figure out why the failures occurs. You need to be ready to stop and investigate why it fails, and you cannot go forward all the time. You might also investigate failures even when they are just some timing errors, not system errors, but errors related to the testing tool. It still takes time to open the log files and solve what was the cause for the failure." — Site A employee

"Our CI system can say that something is broken and then you investigate for half an hour and discover that it was an environment issue. Our test cases are not one hundred percent stable. Sometimes they fail, even when nothing is wrong. There is still some work to do." — Site A employee

Finally, one reason for the lack of time was many meetings to which people had to participate in. These meetings ranged from periodic Scrum meetings to CoP meetings. Some interviewees told that the meetings were inefficient, the discussion was not focused and decisions were not made:

> *"The problem is not that there are too many meetings... I think we call for meeting, too early many times... It's not efficient, because then you get a lot of questions, that are not relevant... I would remove some of those meetings that are involving a lot of people."* — Site B employee

2) Difficult Components: The product architecture contained third-party components, which were causing trouble in the CI system. For example, one of the systems could not be updated automatically and the configuration of the system could not be version controlled. This meant that the system needed to be updated to the CI system manually during a certain time weekly. The components also caused delays in automated tests, so the tests needed to either wait for long periods of time, or fail from time to time.

Due to these challenges, the case organization had started to replace one of these third-party components from scratch. Four teams were assigned for this single purpose alone and the new system was designed to be testable and deployable continuously. However, this new component was still developed in an isolated environment, not in the CI environment.

3) Unstable Tests: Some of the tests developed for the end-to-end CI system were unstable: they failed sometimes even when the actual code changes did not contain any errors. Interviewees noted that many tests had failed for unknown reason and nothing was done to fix that:

> *"No, [the tests] are not working. It's, maybe 30 or something like that, that fails every time and no-one cares because we know that they fail."* — Site B employee

This has lead to the situation that the tests could not be trusted and real bugs might be causing hidden problems:

> *"If we don't take the problem seriously, I mean, I may create an error in the code but, my bug will disappear among 30 others."* — Site B employee

According to the interviewees, there were multiple causes for the instability. First, some of the difficult third-party components caused delays and timing issues which randomly failed the tests. Second, testing the UI was causing problems, but none of the interviewees actually knew the root cause of that problem.

4) Slow Tests: Some interviewees noted that running the tests was quite slow, and considering that the test coverage needed to improve, there might come problems later on:

> *"It's many hours now, the nightly build. It's running several hours and... I'm not sure how many more test cases we can actually add to that before it starts taking too long."* — Site B employee

However, in general, it was surprising that the time needed for running the tests was not commented as a challenge more often.

5) Insufficient Test Environments: The test environments for the product were insufficient on two levels: there were not enough testing environments for everyone to use and the available environments were not clones of the production environment. This was a bigger issue for the teams working with the network system. Even the usage of the available testing environments was not clear to the interviewees:

> *"The upgrades of all these different things and do testing in different ways and connect them it's a mess in my point of view. And there's no clear common strategy on how to use this or what the different environments are for. So people use them as they like, which might be fine, better than not using them I guess, but I think there's a big room for improvement in this area."* — Site C employee

The team on Site C did not have a proper testing environment. This had lead to the situation that some testing needed to be done in the production environment. But since the production environment was controlled by Site D, the changes made to it were slow and hindered the testing process:

> *"There is a problem that, some of our, our test environments, like, I say staging for example, it doesn't have full connectivity at the moment. So, there are a number of workarounds that are in place, that some things are using, the production environment for doing certain, testing. Then, this becomes an even bigger problem because, we are, in a test environment wanting to do, small configuration changes, try things out. It's not working, let's try like this instead. Those changes, can result in that we have to wait ten days."* — Site C employee

6) Agreement on Tools: While there had been decisions to use a single test framework and a single CI system for the product, these decisions were not agreed with all the teams:

> *"I don't think we have a good agreement on the tool itself and how are we maintaining this tool and the libraries and the versions of the libraries. So I think the ways of working, the test framework and who are responsible for what and how do you discuss it and, I think it should be improved. Or we don't understand fully how they want us to work."* — Site B employee

Some teams had tried to use the given framework, but they had run into obstacles that prevented the usage:

> *"It's quite easy to write [test cases], but I experienced that the test framework was very buggy. So it was difficult to do the test cases."* — Site B employee

Thus, in reality many teams had chosen their own test frameworks and CI systems based on their own needs and competences:

> *"Function test with [the team's test framework] was much much simpler, and that the [product test framework] was.. it was very impressive for what it can do, but.. it was.. quite difficult for us to try and maintain so we've done, we've done a beauty contest between [the frameworks] and.. we thought that the [team's test framework], allowed us to build stuff much quicker, to be honest."* — Site C employee

7) Global Distribution of the Organization: The distribution of the organization caused problems when trying to communicate between the sites. The communication mainly took place through electronic communication tools and only part of the people had met each other face-to-face. It seemed that the lack of face-to-face communication and lack of good understanding of each other aggravated many of the previously mentioned problems.

The case organization used video conferencing, chat, and voice communication tools for the discussions between the sites. However, there were problems with the used tools:

> *"So, we were invited to, we use video conferences, we're invited, we often sit there listening, it's very hard to say what they [participants from Site A] say, they turn to each other, talk to each other within that little group, and eventually, we're, hey, we're here, could we say something. And that gets very little room."* — Site B employee

In many interviews, it was evident that the CI was practiced more on Site A, but the visibility to other sites was not good. Teams requiring help with the test framework and other CI related issues could get it immediately on Site A, but for other sites, getting help was harder:

> *"I don't know why it is hard. One reason can be that the teams in Country B ask for help rarely because, maybe email is not the best tool for this, but for the people who are here on Site A, it is easier for them to just walk here and ask, if they could be helped. At least it feels like that when you look with that person, from the same PC, it is easier to understand, because you can explain why to do it like this and not like this."*
> — Site A employee

In addition to that, the CI competence was located at one site, the operations organization was also separated to another site. This caused other problems regarding the deployment of the product:

> *"We have a DevOps organization, but we don't really have that type of cooperation where we can say OK, the scripts that I'm now using to deploy on my test environment, should be the same scripts that I can use to deploy my production environment, or at least, so similar that we don't introduce the risk that they won't work when we get to production. We don't have that kind of relationship really at the moment with the DevOps organization. They're asking for manual... What do they call them? It's basically an operational instruction telling you how to do the upgrade, and how to do the deployment."*
> — Site C employee

C. *Perceptions of Continuous Integration Adoption (RQ3)*

1) Positive Perceptions: Most of the interviewees on Site A were positive about the CI adoption. Some still complained that they lacked time to practice CI, but were positive about the direction they were heading. While the prevalent CI system had its problems, interviewees on Site A had already had positive experiences with it:

> *"In the end of last release, we had to make changes in a short time period. Because we had the automated cases, we could run them in an hour whereas manually testing them would have taken a day or two. So we can run the cases continuously and we don't need to go clicking the UI by hand. So I think the cases will pay for themselves."* — Site A employee

> *"We have found quite many faults from the system, and also if some modification changes something unintentionally, then we notice that quite well with the cases we have at the moment."*
> — Site A employee

2) Critical Perceptions: On other sites, there were critical perceptions of the adoption. Especially the interviewees on Site B were critical towards the adoption. However, there were some interviewees on Site B who were rather neutral towards the adoption, because they did not work with the end-to-end business system CI directly.

The critical perceptions were not towards the CI as a practice, but towards the way CI had been introduced in the product organization:

> *"I feel the CI thing here, has been very much driven by what people have perceived as being the correct way of doing CI. And, no one really cared to look upon reality and see can we do this, what are the limitations within the system, what kind of a legacy do we have. Are people competent about this, do we have an agile organization which is stable enough to do this. And, obviously there was a "no" to most of these questions. It really didn't quite matter, we should go CI anyhow, and the information from the CoP on what has been decided or not has been extremely limited."* — Site B employee

From the perspective of Site B, the adoption had been taken without considering the constraints of the product:

> *"[Site A] haven't cared at all about what the product is or the prerequisites for going agile or going CI, within the product."*
> — Site B employee

It was clear that the communication about the adoption had not worked between the sites as was supposed to:

> *"I think it could have been done in a much more efficient way, if it had been driven in a more open-minded manner and if it had been "communication" rather than an idea which was imposed."*
> — Site B employee

CI as a general practice was still seen as needed for the organization:

> *"We have to go, we have to do it, we have to go CI. Absolutely. I'm not arguing with that, quite the contrary. But we need to do it in a smarter, more open-minded way than we've done so far. More visible way."* — Site B employee

Simply enforcing it though was not working as intended, since the reasons why to adopt CI were not communicated clearly enough:

> *"When people haven't quite understood what it was for, they haven't understood the "Why" and the CoP hasn't been able to satisfactorily explain that either."* — Site B employee

Another critical perception was towards the enforcement of creation of automated test cases:

> *"I think automated test cases, is one big problem. I think it's becoming a religion that the management believes that, automated test cases will solve everything."*
> — Site B employee

The critique was mainly towards automatically testing the user interfaces, where there were multiple factors to take into account. Some interviewees claimed that the automated cases are never sufficient and errors can be found even when the tests pass:

> *"I knew that from when I was a tester that we could have something that gets passed with automated test cases and then you can find, a dozen issues in ten minutes just by looking at it."* — Site B employee

It was acknowledged on the Site A too that the user interface requires some manual testing. Most of the people agreed that not everything can be automatically tested:

> *"It's really hard to automate test cases for a presentation layer. Because it's the human eye that has to determine if it looks good, if it feels okay. If the text is, just alignment is quite a big problem. That's never picked up by the automated test cases."*
> — Site B employee

The requirement to do automated test cases was then fulfilled by creating test cases just because they had to do those:

> *"Basically we're just doing automated test cases because we have to. They don't say anything. Basically you just, go in and do something."* — Site B employee

While creating more comprehensive automated test cases was seen possible, lack of time to do that was seen as a challenge:

> *"It's a thousand different combinations you can navigate through a GUI, but, you can't make test cases for it. So basically you're making a best case scenario test case. Straight through, and then you say "it's successful". The problem is that it takes really quite a lot of time to do the test case."* — Site B employee

Finally, there was a critical perception that there was no general vision or strategy on testing the product:

> *"I'm lacking a clear vision or strategy for this on how it should work for us. We've been asking for a test strategy kind of long time and never seen anything. So what, by that I mean what are the different, what's the purpose of each different type of testing environment we have? How are we going to use it? How is it going to be connected to everything? How is it upgraded, everything done? And all that has to be understood and agreed between all the teams and organizations."*
> — Site C employee

This critical perception was given by some interviewees on all sites and thus was not specific to one site only.

3) Neutral Perceptions: There were some teams that were working with components not connected to the end-to-end business system CI. Thus, their perceptions of that system and its usage were fairly neutral due to low visibility:

> *"I don't really see it. I just see meeting notes, I see focus area discussion in our employee meetings that we should focus on it, but I don't really see the results. So, I don't know. It feels like there is a lot of focus, people want it and then we try to build it but I don't see the results. So, I actually don't know."*
> — Site C employee

Another reason for low visibility was the site distribution:

> *"I think our CI environment and stuff like that, I think we, I think it's quite.. it's very much in the background, it's.. we're not being informed when things break. We don't have our own continuous integration environment, it's sitting on a server down in, somewhere down in Site A or stuff like that."*
> — Site C employee

These teams still had their own CI systems for their components only and they were mostly satisfied working with those systems.

D. How Should the Adoption of Continuous Integration Proceed? (RQ4)

In this section, we present seven suggestions, made by our interviewees, on how the case organization should proceed to overcome the challenges and continue the CI adoption (see Table IV).

1) Change the Architecture: One major problem with the adoption of CI was the architecture of the product. The product consisted of third-party products, custom-made components and Ericsson products. The communication between these systems was not seen as clear enough and it could be enhanced so that changes in one system would not cause problems within other systems so easily:

> *"We should clean the vague signaling between the systems, such as databases. We should standardize the signaling between the systems."* — Site A employee

Other than that, it was seen that the third-party products made it really difficult to practice CI. Many interviewees suggested that these third-party products should be replaced:

> *"[We need to] change the architecture in such a way that you are not so, third-party product dependable anymore."*
> — Site B employee

One such third-party product was being replaced partially at the time of interviews to avoid the problems it was causing in the CI system.

Furthermore, there were even more ambitious goals regarding the architecture. Some interviewees would have liked to organize the architecture so that the whole product could be virtualized:

> *"We have a lot of issues now, not being able to test properly because we are, we have dependencies between the test environments not being in the same shape as the production environment or not having enough resources but, having.. more.. virtualized or cloud-enabled architecture that would not be a problem anymore because you could basically have it, capacity on demand or virtualization on demand if you need another, environment you can.. for a certain period of time you just, order it."*
> — Site B employee

2) Proper Testing Environments: While the case organization had put initial effort to build an end-to-end CI testing environment for the business system, the network system was not included in the environment. Furthermore, there was no production-like environment for testing purposes for the network system and the teams developing the network system needed to do the final testing in the production system.

During the time of interviews the CI end-to-end testing environment was reported to be unstable. Some tests failed frequently for unknown reasons. The interviewees told that this instability needs to be taken away before the system can be trusted.

3) Better Communication: Some interviewees hoped that the communication considering CI, especially between the sites, was more open:

> *"Looking at CI for its own sake, they [people at Site A] need to open up and be much more perceptive of the overall system and the requirements the system have on ways of working, the prerequisites that needs to be fulfilled for things to work.*

> *And, they also need to have some kind of a basic course in videoconferencing."* — Site B employee

Since the team members lacked time for meetings, it was suggested that email was used more for communication:

> *"I feel that they should use more.. mail distribution list and so on. So that you don't have to sit in meetings for everything, because we don't really have time. I think that's basically, what it boils down to is, I don't want to have to sit, for one day each week in a number of different CoPs, to get small pieces of information that I could've, gotten by just reading an e-mail."* — Site C employee

This would allow members to participate in the discussion when they could and not only on specific times. However, this suggestion was given by a single team member and it does not exclude other ways of better communication, such as face-to-face meetings. Interviewees also suggested that some of the meetings could be made more efficient with an agenda and a chairman leading the discussion.

4) Testing Strategy: The interviewees would have liked to have clearer vision and strategy how to test their product:

> *"When it comes to the verification I think we should really, really think through which nodes, which parts of the system we want to make automated test cases for."* — Site B employee

This would also include the usage of the test environments:

> *"I would just hope that we get this clear understanding and common view on the different environments and how to use them or how to book the system or how to, when we need to do upgrades and so on. Because just at the moment it's just a bit chaotic I would say. It might be good that it's chaotic but I think we are, it's not efficient at the moment anyways."* — Site C employee

5) Division of Labor: There were different opinions on how should the creation of automated test cases be done. Some would have liked that specified experts were creating the test cases, because with specialized knowledge, it would require less effort:

> *"We should probably also have dedicated resources, that are experts with the test framework."* — Site B employee

> *"If you have a couple of resources dedicated for just doing, creating automated test cases that would be, then it's a totally different, story. Then those persons could probably do it quite easily and fast."* — Site B employee

However, some thought that it would be better not to have specialized teams centered around CI but that the whole product organization should be responsible for running it:

> *"In my opinion, ideally the [CI system] should be maintained by the whole community. Rather than a physical team."* — Site A employee

One suggestion was to give concrete goals for teams to improve CI adoption instead of leaving it as voluntary:

> *"Let's take the POs and system architects to specify objectives to certain time periods, it can be a month, a year, or just a sprint. We could be given CI goals and tasks for each sprint. That way it would not be just a "nice to have" discussion, but there would be real requirements for it."* — Site A employee

> *"As a team member, I feel that if we had in each sprint some clear goal, for example that we had to get the functional test coverage a bit higher and in next sprint we would have to make a new deployment script. That would create us the pressure."* — Site A employee

6) Support for Creating Automated Test Cases: Additional support would be needed to further enhance learning to create automated test cases, especially at Sites B and C:

> *"The competence needs to be spread more. There has been some trainings, which are good, but if you don't start the work immediately then you forget it. Our team got lucky because we got the competence and the CI team does daily work with the system so they have good competence. If the teams cannot be mixed, then it would be good to have a support person from whom help could be asked and who could give tips to specific problems."* — Site A employee

> *"Well, for us it is quite easy because we are here and we have so many experts that we get enough help for creating the cases, that is not a problem. Maybe for Country B it is more difficult because it is a totally different thing when someone comes to sit next to you than trying to communicate over voice communication software or by phone."* — Site A employee

7) More Frequent Smaller Releases: Interviewees wanted to get rid of the current release model to allow more flexible releases:

> *"The releases at the moment, they tend to be kind of big, monolithic releases that you release the whole, many components changing at the same time. To be honest, I think that maybe isn't a good way to do it. Maybe we should be splitting it up into... OK, we have a feature here, it's affecting this component. Then you can release that. And you should be able to do that. Often, if you're working on a particular feature, you should be able to release it often."* — Site C employee

This would require that operations competence was brought also to the teams:

> *"If the team had access to the production environment and ways to actually do things there ourselves. Now we hand over all the deployment activities to operations and they do it for team units. But so the team can't actually do things themselves in the production environment."* — Site C employee

V. Discussion

In this section we discuss the answers to our research questions, as well as the limitations of this study.

A. What actions were done to adopt continuous integration?

The case organization had done three major actions for adopting CI: 1) assigned specialized teams for building the infrastructure and creating automated legacy test cases, 2) provided trainings for the infrastructure and test framework, and 3) spread the CI competence from specialized teams to the other teams. The organization had also started replacing a difficult component which caused trouble when adopting CI.

The adoption path of the case seems to have worked to some extent. The case organization created a working end-to-end CI infrastructure, however, only for the business system part of the product. Also, the automated test coverage of the legacy code base was brought much higher. However, the quality of the CI system was still low: there were unstable test cases and the speed of the system was not sufficient for fast feedback. Moreover, automated test cases were not created for new features by all the other teams, as the ”CI mindset” was just inside the two originating teams.

The quality of the system could have been better if some architectural changes to the system had been done before the CI adoption. The same suggestion is given by Gruver et al. [20] who describe how CI was adopted for HP LaserJet firmware. The mindset could have been broader if building the infrastructure was done in a more open fashion, a suggestion that was given by some critical interviewees. Finally, the team members could have been given more time to overcome the initial learning phase, as suggested by Eck et al. [17].

B. What challenges were faced when adopting continuous integration?

The organization had faced and was still facing numerous challenges. Previous studies have brought up that there are multiple challenges when adopting CI [6], [7], both technical and social. Our perception of the challenges was that the most critical challenges were related to the lack of time, the architecture of the product, and the global distribution of the organization. Most of the interviewees knew what their challenges were related to the test infrastructure and instability of the tests. They just lacked resources to do something about those issues. Because of the nature of the architecture, the adoption of CI required more resources than was perhaps suspected by the management.

Other major challenge hindering the CI adoption was the global distribution of the organization. There was a clear separation of opinions and perceptions between different sites. Most of the interviewees lacked visibility to the other sites and had found the communication tools uncomfortable to use. The distribution challenge was quite invisible for the interviewees. The problem only came visible when the other sites were required to create automated test cases for new features.

C. How was the adoption of continuous integration perceived by different stakeholders?

The adoption was perceived most positively by the people on the site driving the adoption. Some perceptions were rather neutral, because they were not included in the current stage of CI adoption. Most critical perceptions were given by interviewees on different sites than the driving site.

It was somewhat surprising to us that most of the interviewees were fairly familiar with the concept of CI and were generally positive towards it. The critical perceptions were mostly against the way of adoption, not against the adoption itself. Thus, the challenge of the adoption is not to convert people to believe in CI, but instead to find a common view of the adoption path towards CI. As in our case, this proved to be a difficult task because of the global distribution of the organization.

D. How would different stakeholders proceed with the adoption of continuous integration?

The interviewees gave seven suggestions on how to proceed with the adoption: change the architecture, proper testing environments, better communication, testing strategy, division of labor, support for creating automated test cases and more frequent smaller releases. To some extent, these suggestions were not mutually exclusive and possibly all of them could be implemented. Contrasting this result to the differing perceptions seems interesting; despite the differing perceptions, the given next steps were fairly consistent with each other.

The only things needed to drive these next steps further would be a common vision and time allocated for them. This will be a task for the leaders of the product organization to take.

E. Limitations

The results of this study were from a single product organization. A more general understanding of the phenomenon could be achieved by replicating the study in other product organizations and in other companies. Inside the single product organization, we were only able to visit the three development sites, but not the operations and support site in a third country. It would have been interesting to investigate their perceptions of the CI adoption too.

VI. Conclusion

For a globally distributed and large-scale software development organization, the adoption of continuous integration remains a challenging task. In our case study, we found that a difficult architecture, lack of time, and the distribution of the organization can be the most influential factors hindering the adoption.

Global distribution of teams causes problems, if some competencies are separated to a single site. In the studied case, the CI competencies and operations organization were isolated from other sites and this was seen to cause problems. The prerequisites for continuous delivery were not fully present in the studied case, as is summed up by Jez Humble:

> *”Nothing kills continuous delivery of high quality software faster than having a development team in one country, a testing team in another, and the operations team in a third.”*
>
> — Jez Humble [21]

This case study contributes to the previous research in three ways. First, we presented a description of how the CI adoption happened in the case organization. To our knowledge, similar scientific studies of CI adoption have not been previously published. Second, we presented challenges that the case organization faced during the adoption. Unlike previous studies, we could determine which of the challenges were the most influential factors hindering the adoption. Third, we investigated individual responses and their differences, which allowed us to discover the previously unreported challenges of distributed organization.

Based on this case study, three guidelines can be given to practitioners. First, understand that the product architecture has a significant effect on the adoption. However, do not let architectural problems keep you from implementing continuous integration. You may adjust the architecture by changing technologies or components if needed. Second, give enough time for your team members to overcome the initial learning phase. You have to lower the priority of new features during the adoption. Third, invest in the communication between different sites and avoid centralizing different competencies to certain sites. Understand the value of face-to-face communication and local support personnel when adopting CI.

Acknowledgment

The authors would like to thank Ericsson and in particular the interviewees for participating in the study. The study was funded by TEKES as part of the Need for Speed (N4S) research program of DIGILE (Finnish Strategic Centre for Science, Technology and Innovation in the field of ICT and digital business).

References

[1] M. Leppänen, S. Mäkinen, M. Pagels, V.-P. Eloranta, J. Itkonen, M. V. Mäntylä, and T. Männistö, "The Highways and Country Roads to Continuous Deployment," *Software, IEEE*, vol. 32, no. 2, pp. 64–72, 2015.

[2] M. V. Mäntylä, B. Adams, F. Khomh, E. Engström, and K. Petersen, "On rapid releases and software testing: a case study and a semi-systematic literature review," *Empirical Software Engineering*, Oct. 2014.

[3] N. Kerzazi and F. Khomh, "Factors Impacting Rapid Releases: An Industrial Case Study," in *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ser. ESEM '14. New York, NY, USA: ACM, 2014, pp. 61:1–61:8.

[4] J. Humble and D. Farley, *Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation*, 1st ed. Addison-Wesley Professional, 2010.

[5] D. Ståhl and J. Bosch, "Automated Software Integration Flows in Industry: A Multiple-case Study," in *Companion Proceedings of the 36th International Conference on Software Engineering*, New York, NY, USA, 2014, pp. 54–63.

[6] A. Debbiche, M. Dienér, and R. Berntsson Svensson, "Challenges When Adopting Continuous Integration: A Case Study," in *Product-Focused Software Process Improvement*, ser. Lecture Notes in Computer Science, vol. 8892. Springer International Publishing, 2014, pp. 17–32.

[7] G. G. Claps, R. B. Svensson, and A. Aurum, "On the journey to continuous deployment: Technical and social challenges along the way," *Information and Software Technology*, vol. 57, no. 0, pp. 21 – 31, 2015.

[8] P. Banerjee, C. Bash, R. Friedrich, P. Goldsack, B. A. Huberman, J. Manley, C. Patel, P. Ranganathan, and A. Veitch, "Everything as a Service: Powering the New Information Economy," *Computer*, vol. 44, no. 3, pp. 36–43, 2011.

[9] M. Paasivaara, B. Behm, C. Lassenius, and M. Hallikainen, "Towards Rapid Releases in Large-Scale XaaS Development at Ericsson: A Case Study," in *2014 IEEE 9th International Conference on Global Software Engineering (ICGSE)*, Aug. 2014, pp. 16–25.

[10] M. Paasivaara, O. Väättänen, M. Hallikainen, and C. Lassenius, "Supporting a Large-Scale Lean and Agile Transformation by Defining Common Values," in *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation*, ser. Lecture Notes in Business Information Processing, T. Dingsøyr, N. B. Moe, R. Tonelli, S. Counsell, C. Gencel, and K. Petersen, Eds. Springer International Publishing, May 2014, no. 199, pp. 73–82.

[11] M. Fowler, "Continuous Integration," May 2006. [Online]. Available: http://martinfowler.com/articles/continuousIntegration.html

[12] G. Brooks, "Team Pace – Keeping Build Times Down," in *Proceedings of the Agile 2008*, ser. AGILE '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 294–297.

[13] D. Ståhl and J. Bosch, "Experienced benefits of continuous integration in industry software product development: A case study," in *IASTED Multiconferences - Proceedings of the IASTED International Conference on Software Engineering, SE 2013*, 2013, pp. 736–743.

[14] C. Basarke, C. Berger, and B. Rumpe, "Software & systems engineering process and tools for the development of autonomous driving intelligence," *Journal of Aerospace Computing, Information and Communication*, vol. 4, no. 12, pp. 1158–1174, 2007.

[15] J. Downs, J. Hosking, and B. Plimmer, "Status Communication in Agile Software Teams: A Case Study," in *Proceedings of the 2010 Fifth International Conference on Software Engineering Advances*, ser. ICSEA '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 82–87.

[16] D. Ståhl and J. Bosch, "Modeling Continuous Integration Practice Differences in Industry Software Development," *Journal of Systems and Software*, vol. 87, pp. 48–59, Jan. 2014.

[17] A. Eck, F. Uebernickel, and W. Brenner, "Fit for Continuous Integration: How Organizations Assimilate an Agile Practice," in *Twentieth Americas Conference on Information Systems*, Savannah, Georgia, USA, 2014.

[18] R. K. Yin, *Case study research: Design and methods*, 2nd ed. Sage publications, 1994, vol. 5.

[19] M. Q. Patton, *Qualitative Research & Evaluation Methods*, 3rd ed. SAGE Publications, Jan. 2002, published: Hardcover.

[20] G. Gruver, M. Young, and P. Fulghum, *A Practical Approach to Large-Scale Agile Development: How HP Transformed LaserJet FutureSmart Firmware*, 1st ed. Addison-Wesley Professional, 2012.

[21] J. Humble, "Organize software delivery around outcomes, not roles: continuous delivery and cross-functional teams," Dec. 2011. [Online]. Available: http://continuousdelivery.com/2011/12/organize-software-delivery-around-outcomes-not-roles/