Tampere University

Katariina Collander

IMPROVING THE CAPACITY ESTIMATION OF A SOFTWARE DEVELOPMENT TEAM USING DATA

Master of Science Thesis
Faculty of Information Technology and Communication Sciences
Examiners: Outi Sievi-Korte
June 2022

ABSTRACT

Katariina Collander: Improving the capacity estimation of a software development team using data
Master of Science Thesis
Tampere University
Information Technology
June 2022

Team capacity is a measurement used to determine how much work a software development team can commit to working on in the future. Teams that use Scaled Agile Framework (SAFe) estimate capacity as a part of the product increment (PI) planning. This study identifies which factors affect the team capacity and how capacity estimations could be improved. Improved accuracy of capacity estimation can result in better predictability in the project.

The research was conducted as a case study. First, factors of team capacity were identified from existing literature. Data analysis was conducted on part of the factors to evaluate the impact of the factors in a case company with 12 teams working on a software project. Finally, recommendations on improving team capacity estimations were suggested.

Team capacity was seen to be affected by the team's performance and by the team's plans. Based on the research, team performance factors are team availability and size, corrective maintenance, project complexity, production environment, amount of work in progress, and technical debt. The soft factors are team climate, team age and individual competencies. The factors related to the plans' effect on team capacity are the quality of plans and effort estimations done by the team.

The relationships between team capacity and team availability, corrective maintenance, and team attributes were studied in the data analysis. The relationship between team availability and team capacity was only visible for two out of 11 teams, while the overall correlation for all teams combined was 0,53. While team capacity estimation in SAFe largely relies on the assumption that team capacity depends on team availability, in the case company, the data of 9 teams did not show this relationship. The central role of team availability in the estimations should be questioned. So far, this has not been done in the field research.

Team size was a statistically significant factor for 4 out of 11 teams, making the factor the most apparent one of the analysed factors. For the whole project, team size correlates with team capacity with a negative correlation of -0,29. Corrective maintenance and other team attribute factors were not found to correlate significantly with team capacity.

As seen from the results, team capacity is affected by many factors. It cannot be estimated by relying only on the team availability. Team capacity estimations could be improved by improving the estimation process and the related data. Teams should be encouraged to analyse their past more and be trained on what factors affect team capacity. The teams should be provided with data about their past work and capacity estimates from more than one PI. The risk of inaccurate estimations could be lowered by defining capacity estimate explicitly and using a range for the estimate instead of one value. The quality of the historical data used in the capacity estimations could be improved by encouraging teams to document changes in their work and record the realised efforts put into tasks in addition to the estimated efforts.

Keywords: SAFe, PI planning, team capacity, team velocity, software development estimation

TIIVISTELMÄ

Katariina Collander: Kehittäjätiimin kapasiteetin arvioinnin parantaminen datan avulla
Diplomityö
Tampereen yliopisto
Tietotekniikka
Kesäkuu 2022

Tiimin kapasiteettia käytetään tiimin tulevan työjakson sisällön ja työmäärän päättämiseen. Scaled Agile Frameworkin (SAFe) mukaisesti työskentelevät tiimit arvioivat kapasiteettinsa osana product increment (PI) -suunnittelua. Tässä työssä tunnistettiin mitkä tekijät vaikuttavat tiimin kapasiteettiin ja kuinka kapasiteetin arviointia voi kehittää. Kapasiteetin arvioinnin tarkkuuden kehittäminen voi parantaa ohjelmistoprojektin ennustettavuutta.

Tämä työ toteutettiin tapatustutkimuksena. Tiimin kapasiteetin tekijöitä tunnistettiin alan kirjallisuudesta. Seuraavaksi osaa tekijöistä ja niiden vaikutuksia tutkittavassa 12 tiimin projektissa analysoitiin. Tutkimuksen perusteella muodostettiin suosituksia kapasiteetin arvioinnin parantamiseksi.

Tiimin kapasiteettiin vaikuttaa tiimin tehokkuus sekä tiimin suunnitelmissa pysyminen. Kirjallisuuden mukaan tiimin tehokkuuden tekniset tekijät ovat tiimin saatavilla oleva työaika ja koko sekä vikojen korjaaminen,projektin kompleksisuus, tuotantoympäristö, työn alla olevien tehtävien lukumäärä ja tekninen velka (engl. technical debt). Muita tekijöitä ovat tiimin ilmapiiri, tiimin ikä ja tiimin jäsenten taidot. Suunnitelmat vaikuttavat kapasiteettiin suunnitelmien laadun ja tiimin tekemien työmääräarvioiden kautta.

Data-analyysissa tutkittiin tiimin saatavilla olevan työajan, vikojen korjaamisen ja tiimin ominaisuuksien suhdetta tiimin kapasiteettiin tutkittavassa projektissa. Tiimin saatavilla olevan työajan ja tiimin kapasiteetin korrelaatio oli tilastollisesti merkittävä kahdella 11 tiimistä. Koko projektin tasolla korrelaatiokerroin oli 0,53. SAFen kapasiteettiarviointi pohjautuu pääasiassa oletukseen, että kapasiteetti määräytyy saatavilla olevan työmäärän mukaan. Kuitenkin tutkittavassa projektissa 9 tiimin aineistoista ei löytynyt korrelaatiota kapasiteetin ja työmäärän välillä. Tiimin työajan rooli tärkeimpänä tekijänä kapasittettiarvioinnissa tulisi kyseenalaistaa.

Tiimin koko todettiin tilastollisesti merkittäväksi neljälle 11 tiimistä, tehden siitä näkyvimmän analysoiduista tekijöistä. Koko projektin tasolla tiimin koko korreloi kapasiteetin kanssa negatiivisesti kertoimella -0,29. Muiden tiimin ominaisuuksien ja vikojen korjaamisen ei löydetty korreloivan tilastollisesti merkittävästi kapasiteetin kanssa.

Tulosten perusteella todettiin, että tiimin kapasiteettiin vaikuttaa usea tekijä eikä kapasiteetin arviointia voi pohjata ainoastaan saatavilla olevan työajan määrään. Kapasiteetin arviointia voi parantaa kehittämällä arviointiprosessia ja siihen liittyvää aineistoa. Tiimejä tulisi kannustaa analysoimaan kapasiteettiaan enemmän sekä heille tulisi tarjota valmennusta kapasiteettiin vaikuttavista tekijöistä. Tiimeille tulisi myös tarjota tarvittava aineisto tiimin tehdyistä töistä ja kapasiteettiarvioista pitkällä aikavälillä, jotta tiimit voisivat kehittää arvioitaan. Virheellisten arviointien tuomaa riskiä voisi vähentää määrittelemällä mitä kapasiteettiarviolla tarkoitetaan sekä antamalla arvio vaihteluvälinä yhden luvun sijaan. Arvioinnissa käyttävän aineiston parantamiseksi, tiimejä tulisi kannustaa dokumentoimaan muutokset sekä tallettamaan toteutuneet työmäärät.

Avainsanat: SAFe, PI-suunnittelu, tiimin kapasiteetti, ohjelmistokehityksen arviointi

PREFACE

I'm extremely grateful to Outi Sievi-Korte, my supervisor and advisor who I could always count on for great advice and valuable feedback. Special thanks to the Case Company and all the colleagues who have made my days at the office (or mostly home) fun and shared their ideas regarding this thesis. Warm thanks also go to Tampere University, a friend with whom I have walked hand in hand for these years and now has to be left behind.

Thank you to all the friends at home I can't wait to return to. Thank you to the amazing friends that made this exchange semester so much more than thesis writing. Thank you to my family. Finally, thank you Eetu, you're my sun (Fin. *Aurinko*).

Zadar, Croatia, 15th June 2022

Katariina Collander

CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

PI	Program Increment
SAFe	Scaled Agile framework

1. INTRODUCTION

The software development field is looking for ways to adapt agile principles and methods on a larger scale. Scaled Agile Framework (SAFe) is currently the most common agile framework for larger enterprises, and there is an increasing number of enterprises adopting SAFe [1]. As the popularity of SAFe is growing, the importance of researching the assumptions it relies on is increasing. In SAFe, the amount of work that a team commits to depends on the capacity estimation of the team. Team capacity is the amount of work a team can complete within a period, and SAFe teams estimate it themselves. The process of capacity estimation is only shortly instructed in SAFe literature, and it relies significantly on the availability of a team.[2]. Teams are expected to estimate their future capacity without clear instructions on what to consider while constructing an estimation. The assumption that availability strongly correlates with team performance has not been questioned in the literature.

Unlike the SAFe instructions suggest, estimating team capacities in a changing environment can be challenging. In a quickly growing project, new teams are formed, the team structure can change, and the requirements of the team's development work can be unclear. Team capacity is a sum of two large factors: team performance and the team's capability to adhere to its product increment (PI) plans. Accurate team capacity estimations support good planning, which reduces risk and uncertainty. Capacity estimations can also support decision-making and provide information on team performance. Inaccurate capacity estimations result in teams under or over committing to development work.

This study was executed as a case study in a company that has adapted SAFe. The main goal of this research is to find ways to improve the accuracy of capacity estimation in the case company with a software project that 12 teams work on.

This thesis identifies the major factors affecting team capacity and proposes the case company ways to improve the accuracy of the teams' capacity estimations. The proposed recommendations are general and useful for software development teams aiming to improve their capacity estimations. The recommendations are proposed based on the identified factors of team capacity identified from existing literature and the data analysis performed on a project in the case company.

The research questions this study aims to answer are:

1. What are the major factors that affect team capacity?
2. How can team capacity estimations be improved?

The first research question is approached by identifying team capacity factors from existing literature. A part of the identified factors and their impacts on team capacity are studied more in depth in the context of the case company with data analysis methods. As a part of identifying the major factors of team capacity, the significance and the possible impact of factors are also explored. These factors can also be used to analyse fluctuations in team capacity. Changes in the team capacity are often visible, but the reason why there have been changes can be challenging to detect.

The second research question on ways to improve team capacity estimations is answered based on identified factors and the data analysis findings. The question is answered by proposing recommendations on improving the case company's estimation process and the related data.

This study comprises seven chapters with theoretical background, research methodology, results, discussion and conclusions. The theoretical background is divided into three. Software project management, chapter 2, introduces the important terms of this study which are agile, Scaled Agile Framework (SAFe), product increment (PI) planning and team capacity. Team capacity is then approached from the perspective of team performance in chapter 3 and from team plans in chapter4. Chapter 5 defines the research methodology of the data analysis. The results are presented and analysed in chapter 6. In chapter 7 recommendations are suggested, and the quality of the analysis is assessed. Finally, the findings are summarised in chapter 8.

2. SOFTWARE PROJECT MANAGEMENT

This chapter defines the important terms of project management used in this thesis. First, the waterfall and agile software development models are explained and the Scaled Agile Framework (SAFe) is introduced. Then, team capacity and its role in product increment (PI) planning are explained and tied to the processes in use at the case company.

2.1 Overview of software project management

The two main software development models are waterfall and agile models [3]. The waterfall approach to software development is a linear process where each step is started after completing the previous one. Royce first described it in 1970. The steps of the model are requirements definition, design, implementation, testing, deployment and maintenance. Communication with the customer is done at the beginning of the project when a detailed requirements document is formed. The next possibility for feedback is at the end of the project. [4] Defining the requirements well in the early phases of the project is a great challenge, and often impossible [3].

Software project management has become progressively more agile since the publishing of the Agile Manifesto in 2001. According to Dima et al., the agile model has been the main software development model after 2013 [3]. Agile highlights the importance of individuals and communication, working results, cooperation with customers and reacting to change. The key is to deliver what brings value to the customer iteratively. [5]. Agile methods include Scrum, DevOps, Kanban, Extreme programming and others. Scrum is a light framework and an iterative approach. It sets guidelines on breaking the work into smaller tasks that can be completed in the time frame of an iteration and defines a set of needed roles and meetings. Kanban is a way to visualise the teams' work in three categories: done, in progress and to be done. [6] Agile software development has been considered a more efficient way to produce better quality software than the waterfall model [7].

Agile development is based on iterative lean thinking. Development is done in cycles or iterations, where ideas are soon turned into testable products that can then be validated and iteratively improved. Agile development highlights the importance of good and continuous communication between the customer and the development team. Due to the iterative process and continuous feedback, the agile teams do not plan far ahead and are

ready to change if needed. This is based on the idea that it is better to improve something as soon as possible rather than later when it comes with a higher cost. [7]

A need for scaling the Agile way of working arose when larger enterprises wanted to become Agile. Agile development practices were initially designed for small teams, so new solutions to scale agile were needed. Laanti summarises scaled agility as a way to solve process problems, aside from just software development, using the agile mindset and its tools. [8] One widely used framework that brings agility to the whole organisation is Scaled Agile Framework (SAFe). SAFe was introduced to the industry in 2011 by Leffingwell to scale the practices of agile development for larger enterprises [2].

2.2 SAFe

The case company uses SAFe to implement agile practices. SAFe is a freely available framework that enables controlling and scaling software development to a larger scale. The framework combines some, but not all, agile methods and enables the usage of the methods on all organisational levels of a company. SAFe can be seen as a collection of agile and lean practices. The framework combines some central practices of agile and lean product development with system thinking and team management. [2] SAFe has become popular since 2013 and is now used in several companies. In 2018 SAFe was the dominant scaled agile framework. [9] Laanti expressed that the adopters of SAFe have reported improvements in their productivity and quality. [8] However, Putta et al. recognised that there is only a little scientific research regarding SAFe in practice [9].

Like Agile, SAFe emphasises value-driven development. Other SAFe values are respecting people and culture, flow, innovation and continuous improvement. [2] SAFe provides ways to work at three levels: the portfolio level, which is responsible for investment decisions; the program level, responsible for the execution of planned initials; and the team level. These levels respond to the needs of large enterprises struggling to efficiently organise portfolio and program levels by following the agile principles. [8] The teams can choose their working methods from the methods that are compliant with agile principles. Teams often end up using Scrum, Kanban or a combination of them. [2].

Essential practices of SAFe concerning this study are the cadence of work and especially the product increment (PI) planning. Development work in SAFe is done in program increments (PIs) which consist of iterations of two weeks. The purpose of grouping iterations into a longer period is to enable teams to plan and synchronise with the rest of the organisation. Each PI begins with PI planning, during which the objectives of the coming PI are set, and ends with a retrospective event called Inspect and Adapt. During the PI, there are multiple coordination and demo meetings to keep the program in sync. [2]

Putta et al. researched the benefits and challenges of adopting SAFe from scientific

literature and case studies presented by the SAFe organisation. The found benefits were increased transparency, alignment, predictability, and productivity, as well as decreased time to market. [9] These results align well with the core values of SAFe. However, Putta et al. noted that the business benefits were less visible in the peer-reviewed literature than in SAFe's case studies and that the methods of measuring the factors were not clearly defined [9]. According to Laanti et al., the most significant benefit of SAFe for companies in Finland is the transparency it provides. The other benefits are less visible in the responses to their questionnaire than in the results of Putta et al.. [10]

The challenges of adapting to SAFe are more visible in the peer-reviewed articles than in the SAFe case studies. The challenges Putta et al. found were staff resisting change, the difficulties of the first PI planning, controversies within the framework and the structure and filling its roles. Putta et al. also summarised criticism towards SAFe. SAFe has been called a step away from agile as it is a more plan-driven and detailed method than traditional agile is. [9] Laanti et al. reported that in Finland, most of the companies in 2019 were still at the beginning of their agile journey, combining practices from multiple frameworks instead of fully practising SAFe. [10]

2.3 Team capacity

Team capacity is the estimated number of story points a team can complete during the following PI. It is estimated by the team members and used as a starting point for committing to new work in the coming PI. Actualised team capacity, i.e. team velocity also brings visibility of the team performance to the management. [11] As team capacity only includes the completed story points, the capacity consists of only the development of new features. Other responsibilities of software developers, such as repair work, training, or attending Scrum ceremonies, are not included in capacity. [2]

Team capacity is used as a starting point in PI planning on what new feature work a team can commit to [2]. Improving capacity estimation increases planning efficiency as the teams are increasingly able to deliver what they committed to. Better planning improves resource usage, increases quality, and decreases the time it takes to deliver value to customers [12].

Improving capacity estimation means improving the accuracy of capacity estimation. Teams should aim to evaluate and improve their estimation accuracy while also avoiding high evaluation pressure [13]. Team capacity can be over or underestimated. Overestimation causes teams to fail to complete the work they committed to, which can have a substantial negative impact on the project. Underestimating team capacity results in the team underachieving resulting in wasted resources and potential. In the short run, overestimating capacity can have larger negative results than overestimating. In the field of software development, teams tend to be overoptimistic when it comes to software development

estimations. [14] Thus, as capacity estimation accuracy is increased, the team should keep in mind that underestimating capacity is often a safer choice than overestimating it.

Another term that refers to the story points completed by a team is team velocity. In this study, the difference between team capacity and team velocity is that team velocity refers to the story points a team completed in the past PIs, while team capacity is a prediction done when planning the contents of the coming PI. These terms are often used interchangeably in the industry. In some cases, team capacity refers to the total amount of available workforce. In this study, the number of available working hours of a team is called availability.

Team capacity, the completed story points in an iteration or a PI, is actually formed by two large factors:

- The quality of PI plans and the team's capability of adhering to those plans
- The amount of work that the team does, i.e. the performance of the team

A team cannot have a high capacity if it struggles with either one of the factors. Even if a team completes a large amount of work, i.e. shows a great team performance, if this work was not included in the plan and seen as completed story points, the resulting capacity is low. If a team has planned an iteration well, but the team struggles to complete the work, the capacity will be low. Capacity estimations are also affected by other software development estimations: effort estimation and repair effort estimation. In the next chapters 3 and 4 team performance and the effect of PI plans are explored in more detail. Team performance and effort estimation are topics that have been often researched, for example [14] [15] [16]. However, they are rarely explored from the whole perspective of team capacity. Plan adhering and team performance factors are divided into two chapters even though the factors can overlap and affect both. Another way to distinguish between plan adhering and team performance is that team performance is the true capacity of a team, and that combined with plan adhering is the perceived capacity.

Capacity estimation is rarely studied in depth. There is much research on effort estimation and improving team performance in the field, but the larger picture of planning that they affect is a research gap. Especially the more concise topic of estimating capacity is based on assumptions done by Cohn in 2005. The two resources that mentioned the possible processes of estimating velocity both simply summarised the findings of Cohn [17] [18]. In the reference guide of SAFe, the process of establishing the team's capacity is instructed in one paragraph [2]. The false assumption seems to be that the teams can easily make accurate estimations of their capacity.

Team capacity as a part of PI planning

Capacity estimation is done during the PI planning, where establishing the capacity is one of the first steps to complete. According to Leffingwell, capacity estimation should

be done based on team member availabilities and other standing commitments, such as refactoring and maintenance. The team then commits to work by comparing their team capacity to the estimated number of story points of the work. The capacity is used as a maximum for the committed story points. Once it is reached, the team should stop pulling items from the team's backlog. [2] The first PI plannings of teams adapting SAFe are considered very challenging [9].

During PI planning, the development team itself decides what they commit to completing during the next iterations. They make this decision based on their future availability and the effort estimations of the possible contents. However, teams often fail to meet their commitments even with this freedom. [19]

The goal of estimating the team capacity is to limit over- and under-committing by ensuring that the committed work fits the team's capacity. Cohn lists three ways of estimating team capacity: estimation based on historical data, executing a few iterations and basing capacity estimation on them and forecasting based on the team availability and story points estimations [19]. Cohn also suggests using McConnell's Cone of Uncertainty to define a team's capacity as a range. Cone of Uncertainty presents how in the early phases of a project, estimation is uncertain and difficult. The realised effort can be even double or half of what is estimated at the beginning of the project. As time goes on and the requirements become increasingly specific, the range of the estimate decreases. [20] McDaid et al. highlight that using historical data for estimation is only reliable if the project is stable. Changes in technology, domain, team, working environment and tools make using historical data less useful. [21]

In agile software development, it is solely the team's responsibility to plan and execute the work and manage itself [22]. Even with the freedom to commit only to work that the team finds reasonable, teams often fail to meet the commitments [19]. How often the team is genuinely free from pressure to over-commit can be questioned. However, Pomar et al. found that in the company they were researching, the developers felt no pressure from outside the team [23]. Improving capacity estimations can decrease the number of teams failing to meet their commitments.

The case company estimates team capacity by using a combination of estimations based on historical data and future team availability. Capacity estimations are done by the teams with the guidance of the scrum master during PI planning and accepted by the management. The historical data used is the team's velocity in the previous PI, and forecasting is done by adjusting the velocity with the team's availability in the next iterations. Having completed 10 story points with 15 available days of work in the past means the team can commit to a maximum of 20 story points when the future availability is 30 days. Often the teams only consider the most previous PI's velocity instead of basing their estimates on a more extended range of data. The teams take out shares for maintenance work and

other responsibilities from the capacity. Before setting the capacity estimate, the teams also consider if they have any future changes in the sizes of responsibilities outside of development work. Finally, the estimated capacity is treated as a maximum and committing over it has to be well reasoned.

3. PERFORMANCE OF A SOFTWARE DEVELOPMENT TEAM

In this chapter, the factors affecting a team's performance are explored. The team's performance stands for the amount of work the team completed, and the completion of planned user stories does not necessarily represent this work. Some factors are apparent in the performance in the short term, per PI or even per iteration, and some are only visible in the long run. This exploration of the factors is mainly focused on the technical and soft factors, leaving the relatively stable organisational and environmental factors out of the scope of the study.

The performance of software development teams and the factors affecting it are widely studied topics. Many researches and articles have been published especially on the topic of what affects and how to improve the performance of a software development team, for example [16] [24] [25] [26]. These mostly list soft factors that can improve the communication and motivation within the team. This study stresses the factors that can change per PI, i.e. the factors that teams should analyse and compare when planning for the coming 12 weeks. Long-term factors, such as team climate or diversity, can improve or decrease the team performance steadily, little by little, for each PI. The long-term factors are also explored here as the teams should regularly analyse them to adjust their capacity estimations.

Purna et al. divided the factors into four categories: technical, soft, organisational, and environmental. [25] The technical factors are related to the development processes, requirements and systems. The soft factors are related to communication, people and domain. The Agile Movement has shifted the team performance research from process-centric methodologies to people-centric, emphasising the soft factors [27]. This chapter discusses some of team performance's major technical and soft factors. Later, this listing is used to choose the relevant and available factors for the case company's data analysis.

3.1 Technical factors affecting team performance

Team performance can be affected by technical factors such as team availability and size, project size, production environment, corrective quality maintenance, work in progress

and technical debt.

Team availability

PI planning and SAFe capacity estimation are done based on availability. Team availability is the amount of time that team members can dedicate to their responsibilities inside the team. The assumption is that more availability also means more story points completed. [2] This assumption is widely accepted and used in most Agile companies. The teams state their future planned availabilities during PI planning for the coming iterations. The team's availability depends on the number of members, their amount of working hours, and other responsibilities. The actual availability is also affected by sickness and other unexpected days out of the office.

Not much research has been done to validate the assumption that availability is a major factor in team performance. According to Pomar et al., availability was not a major reason behind fluctuating velocity in one case company. In their case study, the correlation coefficient between velocity and availability was weak, 0.12. [23]

Team size

Team size is the number of people working in a team. Team size and changes to it through the team's history can be considered significant factors to team performance. The effect of a team size growing too large has been widely studied [22] [28] [29]. Team size also greatly affects team availability since the number of people affects the team's availability.

According to Schwaber, the optimal size of an agile team is about seven people. If the team grows larger than that, the chance of miscommunications also increases. [22] Mohamed also acknowledged that team size has a moderate impact on team performance. However, according to Mohamed, the optimal team size is five people, and after that, the communication in the team suffers. [29] Faraj and Sproull further explained that difficulties in coordinating expertise can cause the negative effect of increasing team size. As the team size increases, the team starts to struggle to use each member's strengths and areas of expertise in the most effective way. [30]

Not all studies have detected the effect of increasing team size. Ramasubbu et al. studied team size and team performance and did not find a statistically significant relationship between the factors [28].

Corrective quality maintenance

A significant portion of the responsibilities of a software developer is corrective maintenance, i.e. repair work, which addresses researching, fixing, and documenting found defects in the software. Quality maintenance is not included in a team's performance, but it affects the performance since the resources used to fix defects take away from the development work [2]. The development work done now can also affect the number of

defects in the future as teams with a lot of time pressure can produce more defect-dense software [12].

The estimation of repair work directly affects the team capacity estimation in the case company.The current capacity estimation method by the studied teams is that the teams cut a share of their capacity to represent the resources spent on repair work. During each PI planning, the team revisits that share and predicts how much corrective maintenance workload they expect in the following PI compared to the previous PI.

The amount of resources a team has to dedicate to corrective maintenance depends on the number of defects and the hours spent on fixing those defects. Defects can be found in different phases of development work, during unit testing, integration testing, system testing, or by the customer [31].

Repair work can be estimated with algorithmic or expert methods. Mockus et al. made a model to predict the repair effort based on changes in the project's version control system. They found a clear relationship between new features and defect fixes, suggesting that the number of defects can be predicted using only the number of new features. They also found that the timing of defect fixes can be estimated by calculating the average repair delay from the version control. [31] Goel et al. attempted to predict the time it takes to fix a defect. They found a direct correlation between the number of defects in the system and the fix-effort and that average cyclomatic complexity (a measure of the complexity of a decision structure) is a good indicator of the amount of fix-effort. [32]

The models by Mockus et al. and Goel et al. are examples of the many algorithmic methods developed to estimate corrective maintenance effort. Another method for estimating repair work is expert estimation, which is currently used in the case company. Neither of the methods is agreed to be more accurate than the other [33]. According to Mockus et al., expert estimation is a fitting choice when the project estimators have extensive expertise in the area and when the project has remained similar [31]. The repair work estimations could be improved by ensuring that the team analyses their corrective maintenance load, providing the teams with supportive algorithmic methods or by ensuring that the expertise of the developers with relevant domain backgrounds is used.

Corrective quality maintenance is one significant example of the responsibilities of developers that are not included in the team performance. Other responsibilities that take resources away from a team's performance include additional documentation, separate improvement items, training, supporting other teams and attending scrum events [2].

Project size and complexity

The complexity of a product or a sub-product can impact team performance. Schwaber stated that project complexity is the primary cause of failures in software projects [22]. According to Pomar et al., the complexity of a sub-product directly influences the team

performance, and complexity was the primary cause of velocity fluctuations in their case study. The complexity of a project can hurt team performance because of the increased complexity of requirements and the decreased predictability that comes with high complexity. Teams may not consider that during the planning, so project complexity can also affect the team's adherence to plans. [23] The effect of complexity in the quality of the plans and estimates could be minimised by acknowledging the complexity of a sub-project and making it a priority to consider it.

Schwaber divides complexity into three categories: requirements, technology and people complexity. Combining the requirements and technology complexity allows dividing projects or sub-projects based on the combination into simple, complicated, complex and anarchy projects. According to Schwaber, simple projects no longer exist, so the teams must navigate between complicated and complex projects and try to avoid anarchy. [22] Maroukian also adds to the topic that complexity of a project can decrease the quality of requirements gathering and even cause changing requirements, which in turn decreases the team performance [34].

Blackburn et Al. also found that project duration negatively correlates with team productivity. The longer a project takes, the less productive the team is. [15]

Production environment

Another factor that can impact team performance is the production methods, such as automated tools or software methodologies, that teams use. However, according to Sawyer and Guinan, this is not the case. In their case, there was no relationship between production methods and team performance. They conclude that social factors of a team may have a stronger correlation with team performance than production methods. [35] However, Asproni noted that automated processes help take the repetitive not-value-bringing work out of the hands of the developers. That allows the teams to focus on their prioritised tasks and improve performance. [27] The research of Sawyer and Guinan was conducted 20 years ago, making it highly possible that the impact of production environment on team performance has changed after their research.

Other technical factors

Team performance can be affected by a long list of factors. Two last examples of specific factors that have been researched are the amount of work in progress and technical debt. A high number of items in work in progress can decrease a team's performance. This is because many work-in-progress items can cause a lot of context switching for the team members. It can also result in the team partly completing many items instead of meeting the definition of done [19]. The negative effects can be avoided by developers avoiding working on multiple tasks simultaneously and by making the tasks as small as possible [23] [36].

Technical debt refers to the need for refactoring and the impact of design choices on a software product or sub-product [37]. Technical debt is difficult to quantify and often overshadowed by the clearly visible development of new features. Power suggested that teams that prioritise new development neglecting technical debt will have to go back to investing in improving their situation of technical debt. In other words, a team can temporarily increase their performance, but this can result in a difficultly noticeable decrease in performance in the long run. [38].

3.2 Soft factors affecting team performance

A team's soft, non-technical characteristics can also affect its capacity. These characteristics include, for example, team climate, the quality of communication, team diversity and individual competencies.

Team climate and communication

A healthy team that works well together can result in a better capacity, and a team that is growing together can increase their capacity [26]. Purna et al. also stated that especially mutual trust among the team and the effectiveness of their communication have a large effect on team performance. Other soft factors listed by Purna are team climate, diversity and innovation. [25]

According to Dingsoyr et al., especially the following five factors impact the team performance: team coordination, goal orientation, team cohesion, shared mental models, and team learning [16] Dingosyr et al. also called out for more studies on teamwork in Agile teams, stating that there is not enough evidence to state that self-management leads to better performance, yet. Another widely accepted factor of team performance is team motivation. Asproni called motivation the most important factor for performance [27].

Team communication can suffer because of many reasons. For example, Ramasubbu et al. recognised that team dispersion reduces team productivity significantly. They suggest that the reduction is due to information asymmetry between teams and non-clear authority. [28]

Team age

The age of the team can affect the team's performance. The factor has been studied from the perspective of team familiarity (the time a team has been working together) and team maturity (the level of a team's ability to work together) [26] [39]. The time a team has existed could be considered a rough approximation of the team's maturity and familiarity.

According to Al-Sabbagah et al., there is no significant correlation between team maturity and performance. However, their study stated a correlation between team maturity and the team's ability to adhere to plans. [26]

Team maturity not affecting team performance slightly contradicts Huckman et al.'s findings. According to their study, team familiarity, which is somewhat similar to team maturity, has been shown to have a large positive effect on team performance [39]. They also stated that the years a professional has worked in a company does not affect their performance, whereas the years worked in a specific role improve a member's performance [39].

Team member competencies

Team performance might be based on the individual performances of the team members. Purba et al. proposed a model that lists scores of individual performances that could be tracked to estimate the status of a project. However, they did not provide evidence on individual performance's effect on a team's performance. [40] Ong et al. focused their study on individual expertise. They express that an individual's expertise and their expertise-contribution fit greatly impact team performance. Expertise on a related topic does not increase team performance, showing that it is crucial to optimise the utilisation of expertise. [24] Also, Sawyer and Guinan expressed that individual differences in a team must account for the variations in team performance [35].

Ong. et al. studied the effect of team expertise on team performance and the compatibility of expertise and tasks. They concluded that expertise and especially a mismatch among tasks and expertise could significantly affect team performance. Expertise in a task only related to the current task was not found to affect the performance. [24]

While the literature mentioned above focuses on the team members' technical competencies, Asproni says there are two kinds of competencies. Individuals can have technical and personal competencies. Technical competency refers to the knowledge needed for the team's work and personal to the individual's people skills. According to Asproni, these competencies are equally essential and personal competencies can majorly affect a team's performance. [27]

The identified team performance factors based on existing literature were divided into technical and soft factors. The found technical factors are team availability and size, corrective maintenance, project size and complexity, production environment, work in progress, and technical debt. The soft factors are team climate, maturity and familiarity, and individual competencies.

4. EFFECT OF PI PLANS

The number of completed story points per team is affected by the plans made during PI planning. The impact that plans and adherence to plans have on team capacity is explored in this chapter. The quality of plans and effort estimations affect the realised capacity of a team and should be considered when estimating the team capacity. In this chapter, the impact of plans and effort estimations are explored. The relation between effort estimations and team capacity is highlighted, and ways to improve effort estimation are identified.

4.1 Quality of plans

Good quality plans are easier to adhere to by the teams, enabling good quality work. In the planning stage, the information that the teams are given on the task is vital, affecting the quality of effort estimations [14]. Planning can be of good quality when the team has a clear picture of what is required of them and the factors that the work depends on. Quality planning requires enough time, competence from the team and other teams, a stable environment and precise requirements. The failure to meet one of these can result in poor planning that can affect the team's final perceived or real capacity significantly. In addition to affecting the team capacity and adherence to plans, the quality of plans can cause, for example, missing milestones, extra load on the team, an increase in the number of faults, and a need for rework. [12]

Poor planning can result in an inaccurately small perceived capacity if the plans are later changed vastly [41]. This can be avoided if the teams always create user stories and make effort estimates for changing work, even in the middle of the PI. The real capacity of the team can also be affected by poor planning if it causes a slowing down of the team's work [12]. Team performance can be hurt by poor planning, for example, if a team does not notice a dependency during the planning or if a team has to redo work.

Badampudi researched the factors that affect the efficiency of planning overall [12]. They noted that the topic is not studied in the industry. The literature focuses mainly on technical parts of the planning, such as the requirements and effort estimation. According to Badampudi, 20 factors can affect the quality of the plans, out of which 11 were general or common in their study. The general factors were requirements volatility and quality,

poor effort estimates, the degree of team involvement in decision making and quality of communication. [12] According to Pomar et al., another factor that decreases the quality of plans is the complexity of a product which comes with complex requirements [23].

Planning with inadequate information or requirements can also lead to poor effort estimations. Since the team capacity and velocity are solely based on effort estimations, the impact of poor estimates can be significant. Estimates that are too low cause teams to have lower capacities than in reality, and exaggerated estimates lead to unrealistically high capacities. Effort estimation is explored in the next section.

4.2 Effort estimation

As explained above, team capacity and effort estimation are tightly tied together. Effort estimation is the process of predicting the amount of work, cost, and time required to complete a user story or another type of task [18]. In SAFe, teams estimate efforts of user stories [2]. Especially in the Agile context, where requirements are subject to change, estimating effort has become a more complex challenge as the contents of a task can change even during the PI [41]. Tanveer et al. refer to it as embracing dynamism [42]. The teams and the team's tasks are also often cross-functional, causing that the estimation process requires combining a large domain of skills [17].

All Agile teams do the process of effort estimation. In SAFe, feature and user story estimations are mostly done during PI planning or during other occasions where the tasks are pre-studied. The effort estimation process is the following: A team divides a feature into smaller user stories. Then these user stories are given story point estimations by the team. The team members use their expertise to estimate a user story's volume, complexity, knowledge and uncertainty. [17] This strategy is an expert estimation, which is the most common way to estimate software development effort [33]. Another strategy for effort estimation is algorithmic estimating [31].

In Agile development, the effort estimations are given in story points using a modified Fibonacci sequence with values 1, 2, 3, 4, 8, 13, 20, 40, and 100. The purpose of using modified Fibonacci is to consider the effect of uncertainty of estimating, especially with larger items of work. Using story points allows for estimating effort relatively. For example, a user story with 8 story points requires four times the effort of a user story with 2 story points. [19]

The effort estimation with story point in SAFe is often done with planning poker [43]. In 2015, 63 % of respondents in the survey conducted by Usman et al. said that they use planning poker [14]. Planning poker is an activity where the team members consider each of the estimated work items separately. Once the item is introduced, and questions answered, all team members choose a card with the story point value they estimate the

task as. The members reveal their estimations simultaneously, and the members reason their choices. The team discusses the estimates until they reach a commonly agreed estimation. Using planning poker instead of an open discussion ensures that all team members get a chance to make their estimates independently, and the estimates are not too largely based on the loudest team members. [44]

Effort estimation is a widely studied topic, for example [14] [17], [18] [42]. The research has faced some critique, mainly regarding lacking common terms, transparency or reaching a commonly agreed result. Jorgensen has criticised the field's research for using the term "effort estimation" without properly defining what it means. According to their research, the issue is especially relevant when estimations are based on expert opinions. Teams and management might not even acknowledge how they define effort estimations. Some possible interpretations of the estimation are "planned effort", "budgeted effort", "most likely effort", and "effort with a 50 per cent probability of not exceeding it". The definition of effort should affect the management's risk preparations. [13] The definition of effort estimation should be made clear by the experts performing the effort estimation.

Usman et al. reviewed the research on effort estimation and noticed multiple research gaps. Most studies that aim to improve the effort estimations do not show what the resulting estimation accuracy was. Usman et al. criticised that the shown estimation accuracies are overall poor, and the field is still missing a good method. They also expressed that the factors that impact the effort estimations are neither agreed on nor studied enough. [44]

How effort estimation and capacity estimation are related

Capacity estimation is dependent on effort estimation since historical velocity data is always given using the estimated effort in story points [19]. Capacity estimation for the following PI is done based on the completed story points of the previous PI. As capacity is based on the effort estimations of a team and the estimations are done relatively, the capacities of different teams are not comparable. Two teams estimating a capacity of 20 story points does not mean that the teams can take on the same amount of work.

The relation between capacity and the connected estimations makes capacity estimation complex in the long run. The teams should be able to perform effort estimation consistently and with good quality as fluctuations will impact their capacity estimations. However, expert estimations are often inconsistent [42]. Capacity estimations can become unusable if the team starts estimating their user stories on a remarkably different scale.

The figure 4.1 highlights the effects of fluctuating effort estimation. Overestimating effort compared to the effort estimations of the previous PIs can lead to the team underperforming. On the other hand, underestimating effort can lead to a team overcommitting and being unable to finish their committed work.

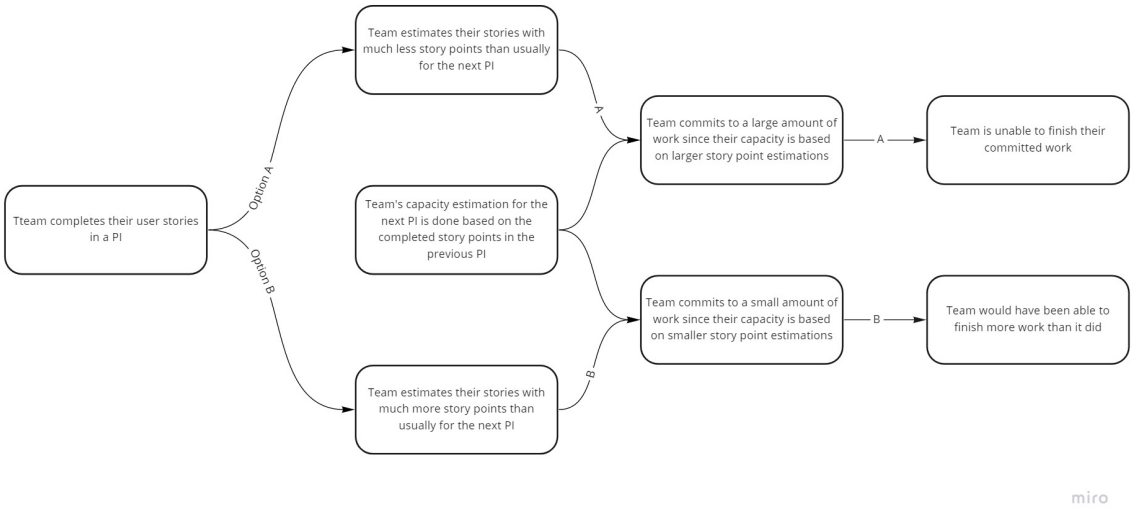

Figure 4.1. *The impact of unstable effort estimation on capacity estimation*

When teams estimate their work continuously similarly, their capacity estimates are not varying because of the effort estimations, and the teams are more likely to be able to finish their committed work. This is visualised in figure 4.2.

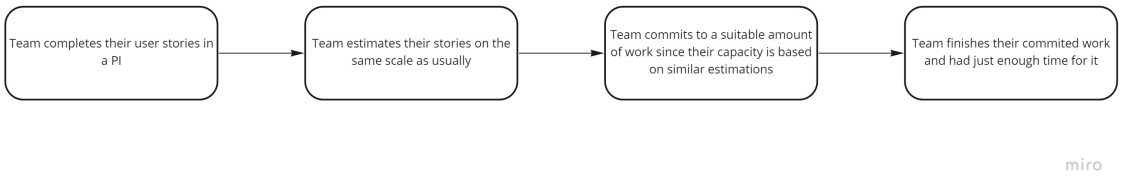

Figure 4.2. *The impact of stable effort estimation on capacity estimation*

In addition to fluctuating effort estimations, the team can make misinformed or poor effort estimations as the estimations are error-prone [42]. The teams may and will under- or overestimate tasks.

The effect of error-prone estimations is not explicitly considered in the case company. The case company does not record a task's realised effort, and capacity estimation is done based solely on the initial estimations. Thus, if a user story was remarkably underestimated, e.g. a task estimated as 3 story points takes two weeks, the completed story points do not correspond to the team's actual capacity. The capacity estimation will be inaccurately low due to the poor effort estimation.

On the other hand, the teams can often underestimate, too. The effect of underestimations may be evened out by overestimating in the long run. If a team uses historical data from multiple PIs, the effect of inaccurate effort estimations is smaller than when using just the most recent PI. That is why using more historical data can lead to better quality capacity estimations for the future PI. [18]

Effort estimation inaccuracy

Capacity estimations can be improved by improving the accuracy of effort estimations. According to Jorgensen, the most commonly used ways to evaluate the estimation accuracy are Mean Magnitude of Relative Error (MMRE) and PRED or similar methods [13]. However, this is often used to calculate the accuracies of developed estimation models. Commonly, like in the case company of Tanveer et al., software companies do not track the accuracy of their estimations [42].

Teams tend to underestimate rather than overestimate [44]. Only 22 % of the respondents of the research of Usman et al. answered that they believe their estimations are close to the truth. 52 % of the respondents believe that they under- or overestimate on average by 25 % or more. Inaccurate estimations can be due to many reasons. The reasons behind inaccurate effort estimations must be explored to answer how the effort estimations can be improved. The crucial role of initial information and requirements of a task is noticed in many studies. Ramessur et al. mentioned false initial information as the largest cause for inaccurate estimations [41]. Coelho et al. and Tanveer et al. defined inadequate or limited information as the problem [17] [42]. Also, Usman et al. labelled quality, volatility and complexity of requirements as the number one reason for estimation inaccuracies in practice [14].

Another factor that is often recognised is the expertise and experience of the teams [17] [44]. Inexperience can result in over-optimism in the estimations, as inexperienced developers often lack the skill to view other than best case scenarios or estimations that cannot be fulfilled by the less skilled members of the team [14]. Tanveer et al. also noted the overoptimism in effort estimations stating that it can be caused by individual human judgement bias or because of group effects [42]. The attempt to minimise the impact of group effect is often made by using planning poker. Ramassur et al. highlighted the soft factors in effort estimation. They expressed the importance of communication and team dynamics in effort estimation. [41]

Some less recognised factors of effort estimation quality in the literature are management-related issues and splitting the work into smaller units. Splitting a feature or a user story into smaller tasks can help the teams provide more accurate estimations. However, the practice also increases the likelihood of forgetting a piece of the work [17]. Usman et al. explored the impact of management-related issues in practice. Agile practitioners reported that a scrum master that does not guide the team or poor change control by the management could decrease the accuracy of effort estimates [14].

Teams may not know which factors to consider when performing effort estimations. In the study of Tanveer et al., developers were first asked to list the factors they consider when making an effort estimation. Later they were told to rank factors listed by the researchers. The developers ended up ranking factors important to effort estimations that they did not consider themselves [42]. Also, Usman et al. note that teams are missing a process to

seek guidance from [14]. While agile is about the teams and individuals instead of strictly defined processes [5], the teams could still benefit from guidance on what to remember during effort estimation. According to Tanveer et al., the currently used methods do not consider the impact of possible changes and the whole variety of factors that can impact the effort of a task [42].

Improving effort estimation

Improving the effort estimations can make project planning more efficient and decrease the number of work spilling into coming iterations. [41] Based on the factors that cause inaccuracies in effort estimation, improving the requirements and supporting inexperienced teams are the most critical factors to improve. Other improvement ideas have been presented by researchers as well.

According to Ahmed et al., the effort estimations done in their study were significantly improved after focusing on the impact of testing and the expectations from people outside of the team when estimating [18]. Usman et al. contradicted this result by expressing that most teams do consider the related testing activities when estimating effort [14]. Ahmed et al. suggested that teams should validate their estimations by revisiting them and checking if they have included all necessary steps for the user story [18].

Regarding the completeness and clarity of requirements, Coelho et al. recommended that teams should be provided clear definitions of done for the tasks that they are estimating [17]. This should be the goal in companies. Tanveer et al. listed ways to improve estimation accuracy and asked developers which they find the most important. They also found that improving the knowledge of the task for the developer is the most important factor. Other researched ways to improve estimation accuracy were increasing the level of experience within the team related to the estimated task, decreasing the task complexity, dependencies and impact, and improving the team's experience in the estimating process. [42]

As mentioned above, teams are not supported in the estimation process, and they may not know what factors to consider when making estimations [42]. The teams should be provided instructions or a framework to improve the accuracy of effort estimations. Tanveer et al. developed a framework for the estimation process based on the findings of their study [42].

The last found method of improving estimation accuracy is storing, calculating and evaluating the accuracies of the teams. In the case study of Tanveer et al., the company did not record the planned and actualised efforts of tasks. As a result, the accuracy of estimations was not followed. [42] If the accuracy was measured and stored, estimations could be tracked and evaluated. With information on the estimation accuracy, the management would also know how to treat and use the future estimations.

5. RESEARCH METHODOLOGY

This thesis was conducted as a case study. After identifying the major factors of team capacity, their significance was researched in the case company. The data analysis and the necessary steps for implementing it in the context of the case company are described in this chapter. First, the decisions of which factors are taken into the study are explained in the context of available data in the case company. Next, the data collection and preprocessing are described, continuing with the technologies and methods used to analyse the data.

5.1 Description of the case study

This thesis was conducted as a case study on a software development project. It consists of research on the background theory of team capacity and data analysis performed on the project. The goal of this research was to answer the following questions:

- What are the main factors that affect team capacity?
- How can team capacity estimations be improved?

The factors of team capacity were identified from existing literature. Some of the factors were then analysed on the case project with data provided by the case company. The factors to include in the data analysis were chosen based on the reliability, simplicity and relevance of the factor's data.

The data analysis aimed to evaluate the significance and impact of the factors and to detect ways to improve capacity estimations. Identifying the main factors affecting capacity estimations can help teams improve their estimations and evaluate the assumptions behind capacity estimation. Finally, improvements for the case company were recommended based on the literature, data analysis and the limitations of the available data.

5.2 Case company

The case company is a large international company. This study was conducted in collaboration with the company's staff at their office in Finland. The study was conducted on a project that is majorly software development related.

Eleven teams of sizes 4 to 10 members were included in the study. Team data was collected over the time of 2 years. The project is developed quickly, and the team structures and divisions are often changing. All teams are working on the same project and have different areas of responsibility when it comes to sub-projects, areas of expertise, and responsibilities outside of development work.

5.3 Choosing factors for the study

This section explores the major factors of team capacity and the data available. To analyse team capacity, data on the work completed by the teams also had to be chosen. The major factors were identified in chapters 3 and 4.

The decisions on which data to include in the data analysis were made based on the identified factors and the related available data sources in the case company. The factors were then evaluated based on the data's reliability, simplicity and relevance. Reliability is the objectiveness and correctness of the data, simplicity refers to the ease of gathering and using the data, and relevance is the importance of the analysed data. The factors should also have as simple of a relationship as possible with the team capacity to form an initial hypothesis on the relationship.

The data for the study was chosen with this evaluation, and the initial hypotheses were formed. In addition, minor recommendations on improving the company's data collection are given related to the factors that could significantly affect team capacity, but no related data exists.

Completed tasks: user stories

The amount of work a team has completed can be seen from the user stories they have closed. User story data is simple to gather as user stories are used and tracked attentively in agile software development. The case company manages user stories in a project management system called Atlassian Jira. Jira is a widely used tool for issue tracking, reporting, scrum, and backlog work in the software development industry [45]. User stories can have many fields associated with them, for example, descriptions, progress dates, work estimations, and links to other user stories, features or epics. All these can be exported from Jira as a table.

The quality of the user story data depends largely on the chosen parameters. The table 5.1 presents commonly used user story parameters with an analysis of their reliability, simplicity and relevance to the study.

Out of the variables presented in table 5.1 the variables with major impacts were chosen for the data analysis while leaving out the complex variables such as long free text fields. The chosen user story data fields are:

Table 5.1. *Commonly used user story parameters*

Field	Reliability	Simplicity	Relevance
Definition	Medium, written by the person defining the user story. Anyone can later modify it. The field can be outdated or contain human errors.	Low, a free-form of text.	Low, a poorly written definition can result in poor estimations and surprises.
Work estimation in story points	Medium, set during the PI planning by the scrum master. The quality of this data depends on the quality of the estimation.	High, work estimations are integers and simple to use in the analysis.	High, story points are used to calculate the team velocity in the past and to estimate the team capacity for the near future.
Status	High, changed manually from a set of possible statuses.	High, simple set of possible values.	Medium, user stories that had unexpected statuses can show that there were issues in completing the user story.
Resolution	High, set manually from a set of possible resolutions.	High, simple set of possible values	High, user stories that were not completed should not be included in calculating the velocity.
Planned implementation	Medium, set during the initial planning and checked by many people but can be changed later.	High, the name of an iteration or a PI	Medium, could be used to calculate planning accuracy.
Realised implementation	High, set automatically after closing a story. Scrum master closes a story.	High, the name of an iteration or a PI	High, shows when work was officially completed.
Comments	Low, anyone can comment on the user stories with any reliability	Low, a free-form of text	Low, a high number of comments can signify that the user story is complex.

- Work estimation in story points
- Resolution date

The work estimations are not highly reliable, but they are the only possible sources of information on the team's velocity and planned capacity. The definitions and comments

of user stories can hold valuable information, but they are too complex for the scope of this work. The complexity of the other data fields might be higher than expected, and there is no guarantee of the stability of the data if there have been significant changes in the way of working with Jira.

It should be noted that the closing of user stories can take some time due to dependencies, forgetting or minor missing details. The user story closing dates from the project management tool might not represent how largely the development work of a story was done during an iteration.

Completed tasks: features and epics

Another way to view the work completed by teams is by looking at the features and epics they have completed. Features consist of multiple user stories. Epics are the highest level of work items that consist of features. Feature and epic data are available in a simple table format in Jira. They are kept up to date, often by people outside of the team in addition to the team itself. Some variables of epics and features that are meaningful concerning the capacity of a team are:

- Epic and feature effort estimates
- Hierarchies and structures of epics and features
- Business value of epics
- Labels

Epics and features are less related to team capacity than user stories since they are not estimated with story points like team capacity is. In the case company, the work estimates of epics and features are done by people outside the team and primarily used for long-term planning. The estimates can represent the complexity of the task, and they are comparable between different teams. However, as the estimates are not done by the people who are developing the feature, these estimates may not relate to the number of working hours dedicated to the feature as well as the user story estimates.

Instead of story points, epics have business values that the team and others set. This value signifies how much value is delivered to the user. [46] Analysing the business value delivered by the teams is another interesting topic but outside the scope of this work. While epics and features are more easily comparable among teams than user stories, user stories were chosen for the analysis since they are estimated in the same unit as team capacity.

Team availability

Team availability is the number of days the team members work on the team's responsibilities. It is used closely in the PI planning with the assumption that team availability correlates with team capacity. In the case company, the historical availability data can be

collected from the previous PI plannings. This, however, is the planned availability and not the actualised availability. The real availability with unplanned sick days and such was not available for this study; thus, the reliability of the available data is medium. The data is simple as the availability is represented as an integer for each iteration and team. The relevance of team availability is very high as it is the primary factor in the current capacity estimation process in the case company. Team availability was chosen for the data analysis due to its simplicity and high relevance, even though the reliability of the data is questionable.

Team attributes: team size, age, and climate

The identified team attributes that can affect team capacity were the team's size, familiarity, maturity, and climate. Data on each team's members and structures can be collected manually from historical files. Team data includes the number of members during each iteration and the number of iterations that a team has existed.

Team age can be used as an approximation of team familiarity and maturity, recognised factors of team capacity. The factors can be explored with available data with the team's age and the number of new members. New members are the members that have joined the team during that PI. The effect of team size on team capacity can be analysed with the data about the number of team members. No data related to the team climate was available thus, the factor was not taken into the analysis.

Team age, team size, and new team members were chosen for the data analysis. The data is stable and can be considered reliable. The data has been handled and is collected manually, which might result in some errors. The personal data about the team members, such as the team members' names, ages, and titles, are left out of this study.

Corrective maintenance

Corrective maintenance takes resources away from team capacity. Corrective software maintenance is managed and tracked in Jira at the case project. The data can be exported as a table with each row corresponding to a found fault in the system. The faults have multiple fields. Some commonly used and relevant fields are presented in table 5.2. The corrective responsibilities of a team can affect the time left for development work significantly, thus, the fault data should be taken into the data analysis. The data fields to include in the study are chosen based on evaluating the field's reliability, simplicity and relevance.

The fault fields chosen for the study are the issues' dates, severity and resolution. This data is enough to calculate the number of concurrent faults and the sum of faults in each iteration or PI. The severity and duration of the fault can be used to estimate the required workload of the fault. The status change dates and tags were not chosen for the study due to their lower relevance and simplicity.

Table 5.2. *Commonly used fault parameters*

Field	Reliability	Simplicity	Relevance
Severity	High, set by the fault manager.	High, simple set of possible values.	High, larger severity can correlate with a larger workload.
Tags	Medium, written by the fault reporter or fault manager.	Low, handwritten, so different formats of similar tags occur.	Medium, shows if the fault is related to a release or testing.
Opening and closing dates	High, dates are stored automatically based on the fault statuses.	High, in date format.	High, used to determine the affected iteration and PI.
Resolution	High, set by a developer after closing the issue.	High, simple set of possible values.	High, used to differ between solved and ignored issues.
Changing state dates	High, dates are stored automatically as the developer changes the issue's status.	Medium, in date format, but the complete log is complicated.	Medium, can show how much time each step of fixing the fault took.

Project complexity and technical debt

Data related to the project complexity and its technical debt can be gathered from the content and changes in the code base. These changes are visible in a version control or code review system. Version control offers data on the number of changes, sizes of changes, and the impacted components. Code review system can offer information on, for example, the required edits to a change before committing it, how many people are involved in the change and how long it takes to get feedback and an approval.

This data is objective and reliable, as it does not require human input. However, the data from the version control and code review systems are multifaceted and complex. That is why project complexity and technical debt were left out of the data analysis.

Production environment

The production environment was identified as a potential factor of team capacity. The environment includes automated tools, software methodologies, and other production methods. There was no data available regarding the changes in production methods.

The one possible simple data regarding the production methods is data about the health of the production pipelines and environments. Pipelines and production and testing environments used in development work are monitored and visualised. This data is automatically collected and reliable. However, this data has low relevance to team capacity as the impact of the production environment on team capacity is not agreed on, and the states of pipelines only represent a minor part of the production environment.

Work in progress

Many user stories in progress have been seen to harm team capacity. The number of user stories in progress can be derived from the "in progress" status history of user stories. However, Jira does not support extracting that information in a simple manner, so the data is unavailable. It would have also been challenging to present the number of items in progress as one value for an iteration. The relationship between work in progress and team capacity was not chosen for the analysis.

Individual competencies

The research literature sees individual competencies as a significant factor in team capacity. This factor was chosen to be left out of the data analysis as it would require handling personal data, which was not wanted. The data analysis is done on the teams as a whole instead of going to the level of the individual team members.

Quality of PI planning

The quality of PI plans was identified to affect the completed story points of a team. No data was available on the planning quality as the teams do not reflect on it after the PI. This data could be collected as a part of each PI planning.

A possible opportunity for collecting data on the quality of the plans would be a confidence vote after each PI. Similar voting is already a part of SAFe during the PI planning. Each team member votes on how confident they feel regarding the made plans on a scale of 1-5. [47] It could be valuable to ask this question after the PI, too, when the team knows how the plans worked. The quality of the plans could also be evaluated in more detail right after the PI planning. Teams could be asked more detailed questions as a part of the confidence vote, such as "How well informed was the team of the tasks?" and "How well did the team prepare for the PI planning?".

Impact of effort estimation

No data comparing the effort estimations and realised effort was available. It is not a case company practice to re-evaluate user stories' efforts after completing a task. The impact of inaccurate effort estimations could not be analysed in depth and was left out of the data analysis.

5.4 Data collection and pre-processing

To summarise, the data presented in table 5.3 was chosen for the data analysis. This data was collected for each team that has worked on the project. These factors were considered the most reliable, simple and relevant.

The data was collected from the sources shown in the table 5.3. The manually collected

Table 5.3. *Factors chosen for the data analysis*

Factor	Data source	Data description
User stories	Project management tool Jira	Story points, closing date, resolution
Team availability	PI planning history	Sum of team's working days
Corrective maintenance	Project management tool Jira	Severity, dates and resolution
Team structure	Collected manually	Team age, number of members and number of new members

data is more prone to errors in the original data and from the process of the collecting than the data exported from tools.

As a part of cleaning the data, rows with impossible values such as earlier closing than starting dates were deleted. Teams that had existed for less than three PIs were excluded from the study as there was not enough data. Timestamps were converted into corresponding iterations and PIs. The duration of each fault and the number of concurrently open faults were derived from the corrective maintenance data.

Part of the data represented an event, and part represented a period. All data was modified to represent an iteration or a PI. For example, each team's faults were combined into iteration and PI summaries. These summaries had information on the maximum number of concurrent faults, severities, the sum of the duration of the faults, and the number of created and resolved faults. User story summaries showed the sum of the story points implemented per period. The data regarding user stories, team availability and corrective maintenance was normalised to calculate Pearson correlations. Team structure data was not normalised as it was differently formatted and analysed with Spearman ranking correlation.

5.5 Technologies used

Pandas and Seaborn tools were used for the data analysis. Correlations were calculated, correlation heat maps were plotted with Pandas, and heat regression model fits were plotted with Seaborn.

Pandas data analysis tool for Python was used for manipulating and analysing the data. Pandas dataframe is widely used for data analysis tasks with tabular data. It handles data efficiently, allows quick manipulation of the data and offers an extensive set of features for data analysis. [48] The data was visualised using Seaborn, a visualisation library built on top of Matplotlib for Python. Seaborne is meant for statistical data exploration and can be used to create informative and visually pleasing graphs with high-level commands [49].

6. RESULTS AND ANALYSIS

In this chapter, the null and alternate hypotheses are formed, and the analysis results are presented and analysed. Based on the results, the null hypotheses are potentially rejected. After obtaining the results, the findings are summarised and analysed. To answer the research questions, the main factors affecting team capacity are presented and analysed regarding the findings from the research literature.

6.1 Setting the hypotheses

The data analysis goals were to test the stated hypotheses related to the factors presented above and to explore the sizes of the impacts of the factors in the capability. Analysis was done separately per team and for the whole project. The analysis was also performed on iteration and PI levels.

The research hypotheses for the chosen factors of team capacity are restated in table 6.1 below as null and alternate hypotheses:

Table 6.1. *Null and alternate hypotheses*

ID	Null hypothesis	Alternate hypothesis
1	Team availability has no effect on team capacity	Team availability has a positive effect on team capacity
2	Amount of corrective maintenance has no effect on team capacity	Amount of corrective maintenance has a negative effect on team capacity
3	Team size is not related to team capacity	Team size is related to team capacity
4	Team age is not related to team capacity	Team age is related to team capacity
5	Number of new members is not related to team capacity	Number of new members is related to team capacity

The first step of the analysis was calculating the number of story points completed per availability, i.e. the efficiency of the teams. Efficiency was calculated by dividing the story points completed in an iteration by the full-time equivalents of work the team was

available. This was used to measure the team's performance to decrease the impact of changes and differences in the team sizes and availabilities.

The possible rejection of the null hypotheses was based on the p-value obtained from the correlations and linear regression models. A p-value of less than 0,05 can support rejecting a null hypothesis. Since the calculations were done for the whole project and per each team, a null hypothesis could be rejected either for the whole project in total or for a certain number of teams. A larger number of teams indicates that the relationship is significant for many of the teams. The strength of the relationship was evaluated based on the correlation's strength and the effect size obtained from the linear regression model.

The relationship between team availability and capacity

Alternate hypothesis 1 proposes that team availability positively affects team capacity. PI planning largely relies on this assumption. The relationship should be studied to verify the relationship and to examine how significant the effect is. The relationship can be studied with Pearson correlation and linear regression using the team availability and completed story points as data.

The relationship between corrective maintenance and capacity

Alternate hypothesis 2 suggests a causal relationship between corrective maintenance and team efficiency, story points completed per week. Before the relationship could be studied, the corrective maintenance workload was summarised for each iteration and PI in meaningful ways. Few summaries were formed: the number of faults created, the number of faults resolved, the sum of fault durations and the maximum number of concurrently open faults. Team efficiency is used as a measure of team capacity.

These summaries of corrective maintenance-related work were visualised, as seen in the figures 6.1 and 6.2. Tickets in the figures refer to the work items on Jira that are created based on found faults. The relationship between corrective maintenance and team capacity was studied with Pearson correlation and linear regression of each of the above summaries and team efficiency.

The relationship between team attributes and capacity

Alternate hypotheses 3, 4, and 5 propose relationships between three different team attributes and team capacity. The hypotheses do not propose the direction of the relationship as the data is not normal, and linear regression cannot be used. These possible relationships can be studied by investigating the Spearman correlations between the team attributes and team capacity. Team efficiency, i.e. story points completed per week, is used in this analysis as the team capacity.

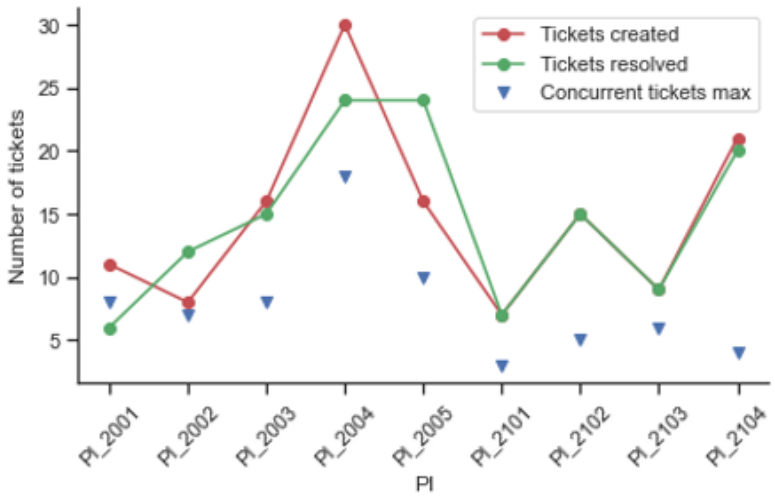

Figure 6.1. *Example of a visualisation: Corrective maintenance summary of a team*

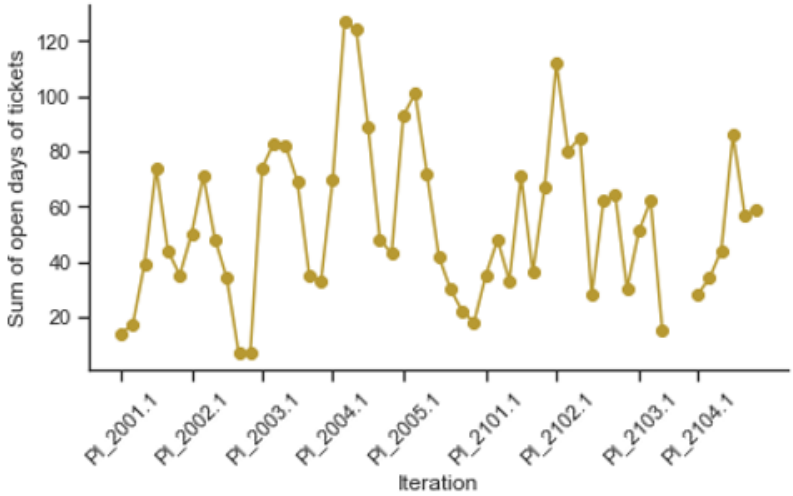

Figure 6.2. *Example of a visualisation: The sum of fault durations of a team*

6.2 Results

The results are presented in subsections based on the relationship under analysis. Each relationship, its correlations, impact sizes, and p-values are studied independently.

The relationship between team availability and team capacity

The null hypothesis number 1 states that team availability does not affect team capacity. The relationship was approached by calculating the Pearson correlations and p-value of team availability and capacity for each team and for the whole project. The impact of the relationship was studied with linear regression and the calculated slope, i.e. the effect of availability on work completed. The results gained are presented in this chapter. The table 6.2 presents the results on the iteration level and table 6.3 presents the PI level for each team.

Table 6.2. *Relationship between team availability and capacity on iteration level*

Value	Correlation coefficient	P-value	Effect
Team 1	0,167	0,278	0,606
Team 2	-0,148	0,511	-0,331
Team 3	0,224	0,218	1,125
Team 4	0,652	0,000	2,440
Team 5	0,316	0,068	1,149
Team 6	0,279	0,109	1,325
Team 7	0,289	0,245	0,701
Team 8	0,113	0,552	0,452
Team 9	0,174	0,451	0,560
Team 10	-0,018	0,922	-0,107
Team 11	0,149	0,497	0,461
Whole project	0,296	<0,001	0,927
Median	0,174	0,278	0,606
Mean	0,200	0,350	0,762

The results vary among the teams. The correlation for the whole project is 0,296, which can be considered a weak or moderate correlation. The p-value is smaller than 0,001, suggesting that the result is significant on the project level.

The weakest correlation among the teams is -0,018 showing no detected correlation. Only one team, Team 2, has a weak negative correlation. Team 4 is the only team with a strong correlation and a small p-value. For other teams, the p-values are large, possibly due to the weak or moderate correlations, small sample sizes or a large amount of variance in the team data.

The effect column of table 6.2 lists the slopes of the linear regression model of the relationship. For team 1, for example, the team completes on average 0,45 story points more for each additional working week based on the model. For the whole project, the effect is 0,93.

The results vary more among the teams on the PI level than on the iteration level. The strongest correlation of a team is 0,989, and the weakest is -0,091. Team 8 has a negative correlation of -0,250. Teams 4, 5, and 9 have small p-values, which is two more teams than on the iteration level.

At the project level, the correlation is 0,529, which is a strong correlation, and the p-value is less than 0,05. The results support rejecting the null hypothesis that there is no relationship between team availability and team capacity on the project level. Surprisingly,

Table 6.3. *Relationship between team availability and capacity on PI level*

Value	Correlation coefficient	P-value	Effect
Team 1	0,217	0,576	0,452
Team 2	-0,091	0,816	-0,133
Team 3	0,205	0,596	0,670
Team 4	0,824	0,006	2,772
Team 5	0,783	0,013	2,335
Team 6	0,349	0,357	1,083
Team 7	0,165	0,791	0,209
Team 8	-0,205	0,516	-0,821
Team 9	0,989	0,011	1,651
Team 10	0,102	0,794	0,384
Team 11	0,506	0,305	0,653
Whole project	0,529	<0,001	1,076
Median	0,217	0,516	0,653
Mean	0,345	0,435	0,842

this is not the case for most teams individually, as there are as many as 8 teams with large p-values, suggesting that there is not enough evidence to reject the null hypothesis for most teams. The overall effect of additional working weeks on team performance is a slight increase of 1,1 story points, and the maximum effect among the teams is 2,77, which is the effect of team 4.

For most of the studied relationships, the correlations on the iteration level are smaller than on the PI level. This shows that the relations are less visible on the iteration level, which can be due to more randomness or more complicated relationships than on the PI level. That is understandable as an iteration is a smaller unit than a PI. The iteration level is more detailed and more difficult to plan. The results can also indicate that the work items are too large to be completed within an iteration, which causes work to be finalised in the following iteration.

Two examples of teams' completed story points as a function of team availability in working weeks (FTE) are presented in figure 6.3. The figure presents the results on the PI level, which show a stronger correlation than the iteration level.

The relationship between corrective maintenance and team capacity

The relationship between corrective work and team capacity was explored with multiple summaries of the corrective effort. The null hypothesis under testing was that the amount of corrective maintenance does not affect team capacity. The summaries used were the

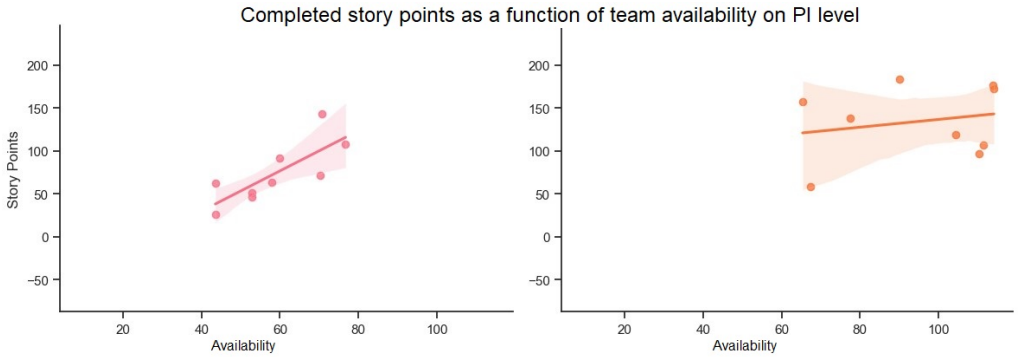

Figure 6.3. *Example of linear regression model visualisations of capacity and team availability on PI level*

sum of fault durations, number of faults created and resolved and maximum number of open faults. The summaries were also analysed by combining them with ticket severities, but this did not affect the results and is not reported separately here. The correlation coefficients, p-values and linear regression models were calculated for each of the summaries. Part of the results is presented for the whole project in tables 6.4 and 6.3.

Table 6.4. *Relationship between corrective maintenance and capacity on iteration level for the whole project*

Value	**Correlation coefficient**	**P-value**	**Effect**
Sum of fault durations	-0,119	0,704	-0,003
Number of faults created	-0,126	0,049	-0,058
Number of faults resolved	-0,081	0,200	-0,035
Maximum number of faults open	-0,062	0,289	-0,015

At the iteration level, the only summary with a p-value lower than 0,05 is the number of faults created and assigned to a team. The value has a weak negative correlation, -0,126, with an effect of -0,058 story points per created fault. Even though the p-value is small enough to possibly reject the null hypothesis regarding corrective maintenance on the iteration level, the correlation is weak and the impact negligible.

Table 6.5. *Relationship between corrective maintenance and capacity on PI level for the whole project*

Value	**Correlation coefficient**	**P-value**	**Effect**
Sum of fault durations	-0,041	0,712	0,000
Number of faults created	0,036	0,748	0,002
Number of faults resolved	0,093	0,405	0,004
Max number of faults open	-0,051	0,648	-0,004

At the PI level, no summary has a p-value that would be small enough to reject the null hypothesis. There is not enough evidence to say there is a relationship between corrective maintenance and team capacity. Even the largest found effect is minor, 0,004. Thus, the results suggest that on the PI level, the amount of work dedicated to corrective work does not have a relationship with the team capacity on the project level. The null hypothesis cannot be rejected based on this evidence for the whole project as no strong correlations were found between corrective work and team capacity on iterations or PIs.

The relationship was also studied for each team. This was particularly important for the corrective work as the teams have very different types and amounts of corrective maintenance work. The correlation coefficients of each of the summaries are summarised in figures 6.4 and 6.5. The figures show stronger correlations amongst the teams than on the project level.

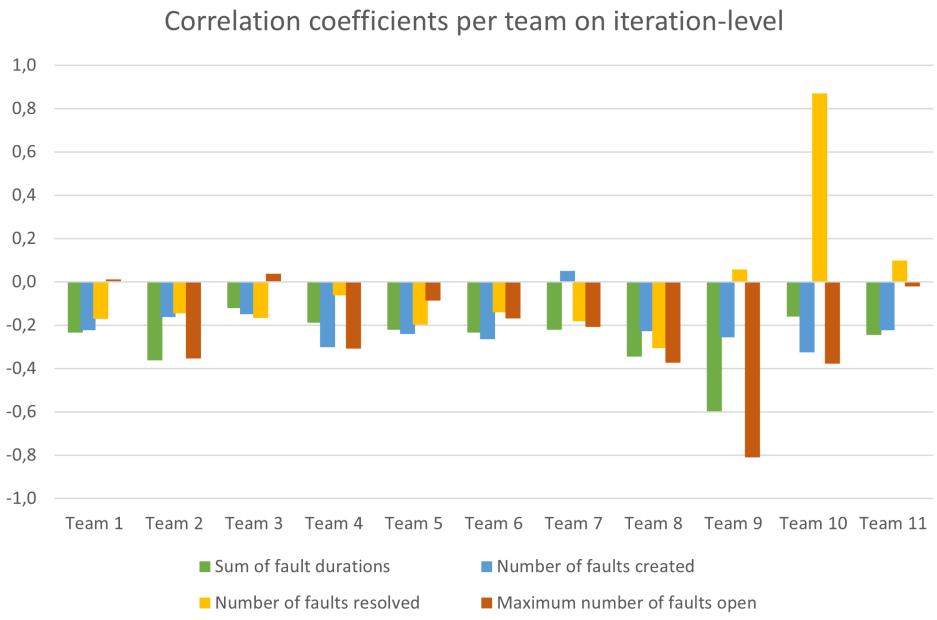

Figure 6.4. *Correlation coefficients of the corrective maintenance summaries on iteration level*

There are strong correlations present for some teams on the iteration level. Most of the correlations are negative as expected, and mostly the correlations are weak to moderate. Team 10 has a very strong positive correlation with the number of faults resolved, which seems counter-intuitive and can be due to the small amount of data. The median correlations of each of the summaries are weak; each is between -0,145 and -0,235. It cannot be concluded which of the summaries presents the corrective maintenance work the best for iterations.

The results for the PIs are very different compared to the iteration results. For example, team 11 has a positive correlation coefficient for the number of tickets created on the iteration level and a moderate negative correlation coefficient on the PI level. The results

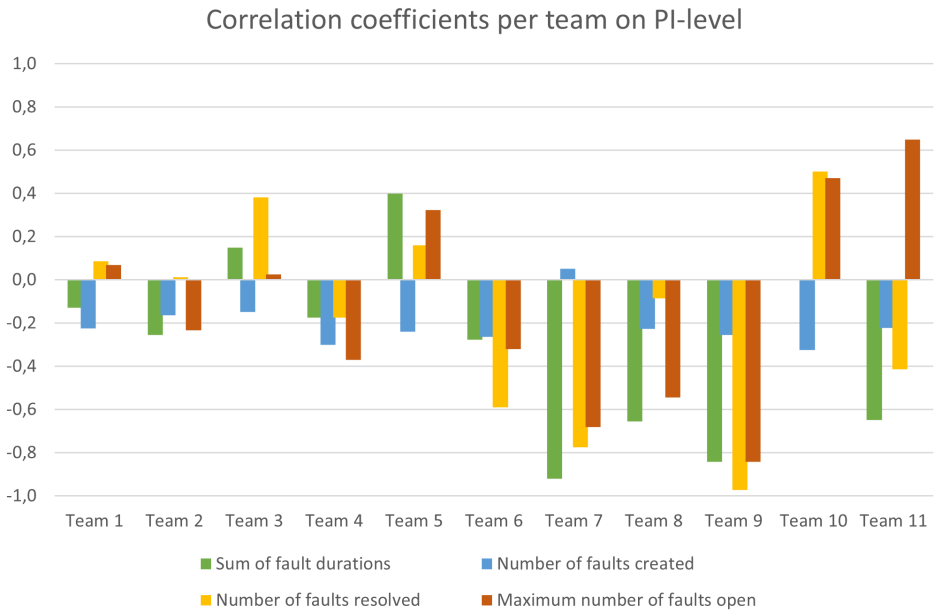

Figure 6.5. *Correlation coefficients of the corrective maintenance summaries on PI level*

seem to suggest that the effect of corrective maintenance on team capacity is either more complex than what can be explored here or very small. On average, the correlation coefficients are larger on the PI level than on the iteration level.

The p-values of the summaries of each team are presented in figures 6.6 and 6.7.

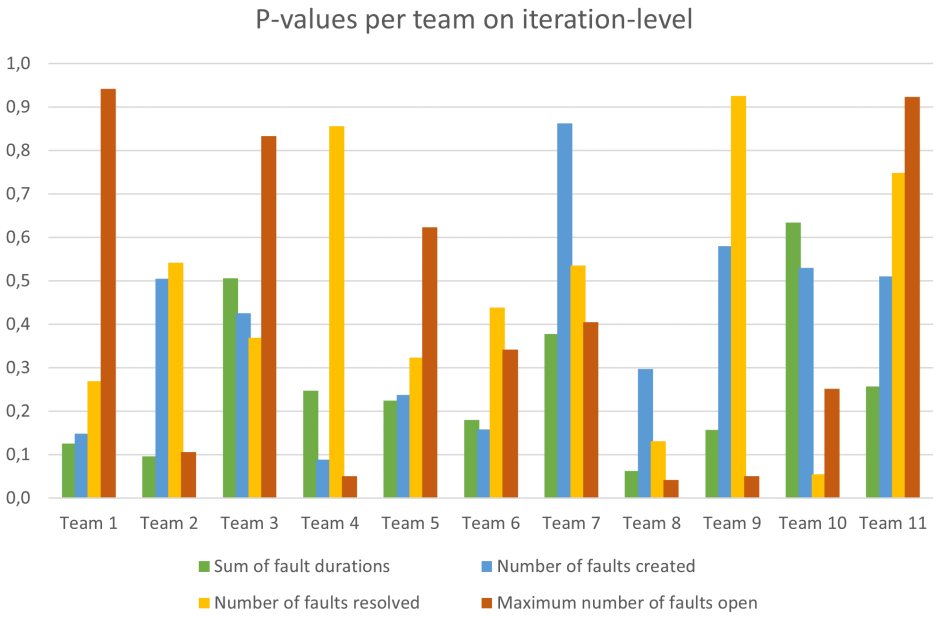

Figure 6.6. *P-values of the corrective maintenance summaries on iteration level*

Most of the p-values are very large in both of the figures. On the iteration level, only the p-value of maximum faults open for team 8 is less than 0,05. The maximum number of faults open and the sum of fault durations have only one team with a small p-value on the

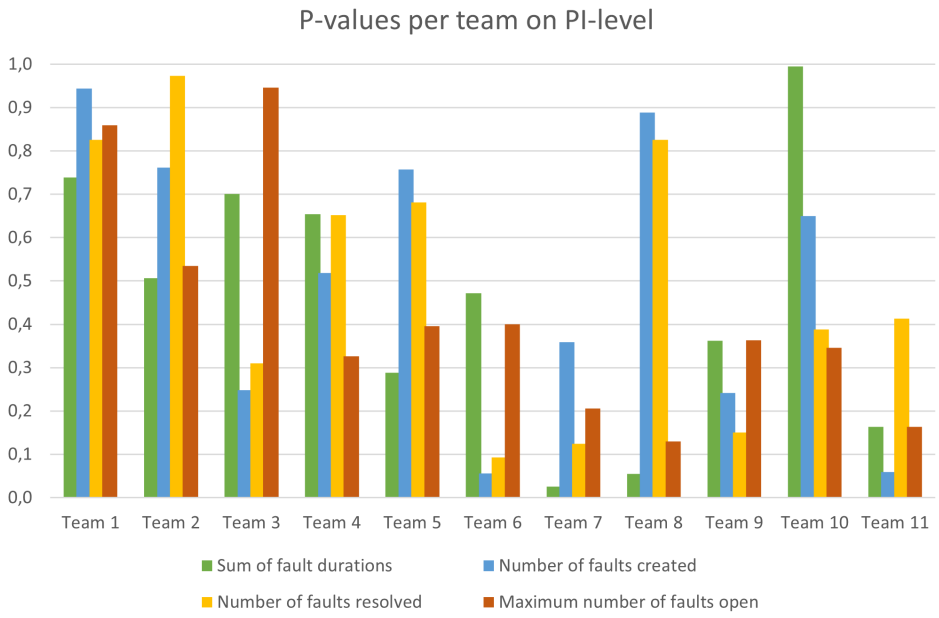

Figure 6.7. *P-values of the corrective maintenance summaries on PI level*

PI level, while the other summaries have zero teams with low p-values. In addition, the effects obtained from the calculated linear regression models were minor for each team and the project. There is not enough evidence to reject the null hypothesis suggesting that corrective maintenance harms the team capacity.

The relationship between team attributes and team capacity

The alternate hypotheses related to team attributes suggest a relationship between the team attributes on team capacity. The studied team attributes were team size, the number of new members in the team and the time that the team has existed. Team attribute data was not normally distributed. Thus Spearman ranking correlations were calculated instead of Pearson coefficients. For the same reason, linear regression models could not be created. The Spearman-correlation coefficients and p-values are presented on the project level in tables 6.6 and 6.7.

Table 6.6. *Relationship between team attributes and capacity on iteration level*

Value	**Correlation coefficient**	**P-value**
Number of members	-0,253	<0,001
Number of new members	-0,024	0,792
Team age	0,063	0,255

The relationship between the number of members in a team and team capacity for the whole project was found to be significant on the iteration and PI level. There is a weak to moderate negative correlation suggesting that as the team size increases, the team's

Table 6.7. *Relationship between team attributes and capacity on PI level*

Value	Correlation coefficient	P-value
Number of members	-0,286	0,008
Number of new members	-0,048	0,791
Team age	0,134	0,217

productivity per member decreases. The p-values of the related null hypothesis are less than 0,001 on the iteration level and 0,008 on the PI level. The results have provided enough evidence to reject the null hypothesis on the project level, as there seems to be a relationship between the number of members and team capacity.

The relationships were also studied for each team. Figures 6.8 and 6.9 visualise the calculated correlation coefficients. The strengths and even the signs of the correlations vary greatly among the teams, from strong positive correlations to strong negative correlations.

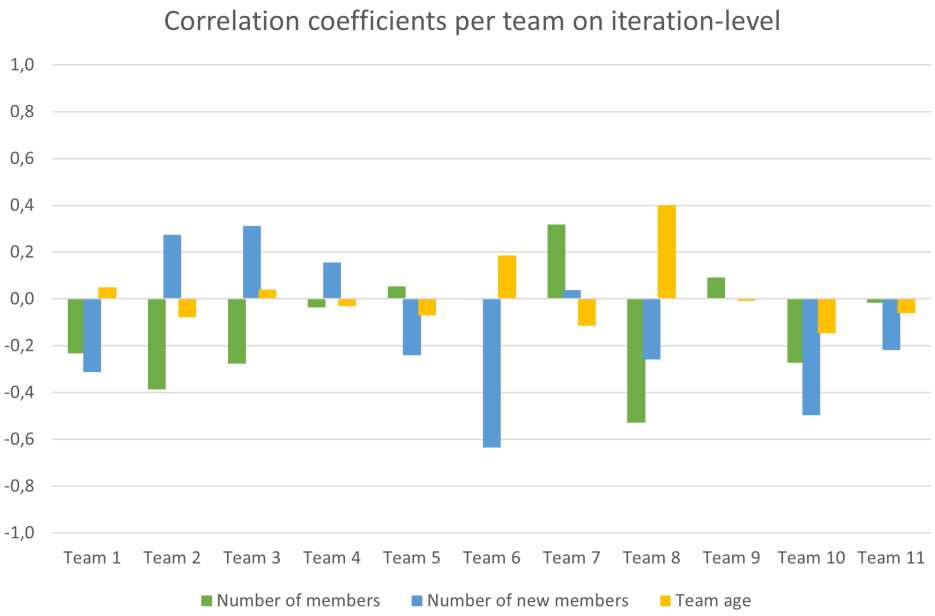

Figure 6.8. *Correlation coefficients of the team attributes on PI level*

The resulting p-values are shown in figures 6.10 and 6.11. On the iteration level among the correlations calculated for each of the summaries, only one team had a small p-value per summary. This was also the case for the effect of the new members and team age on the PI level. The relationship between the number of members and team capacity was significant for 4 teams. Thus, on the PI level, the null hypothesis can be rejected for 4/11 teams and the project as a whole. Out of these four teams, teams 1, 8, and 10 have a strong negative correlation between team capacity and the number of team members, whereas team 7 has a strong positive correlation. The exceptional result of team 7 can be

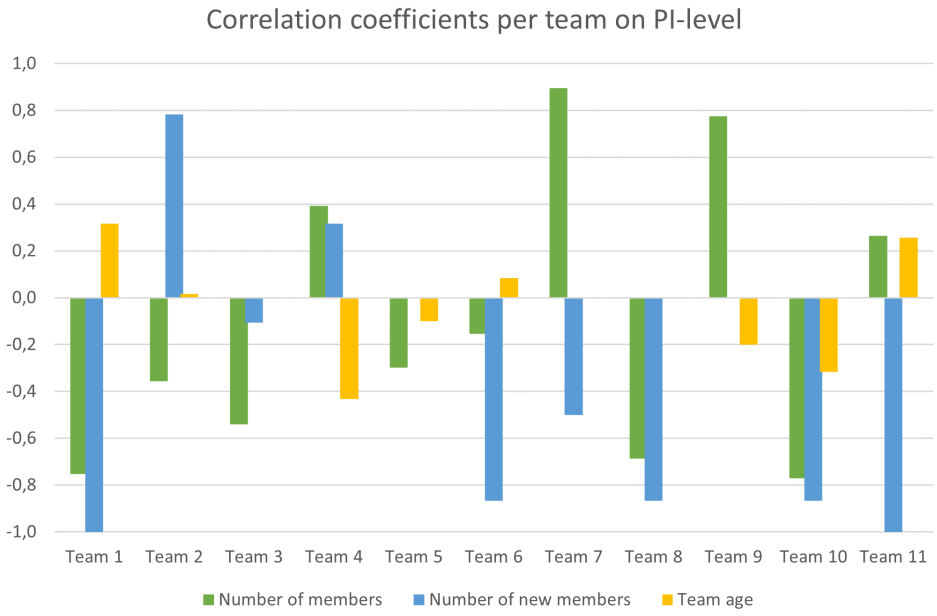

Figure 6.9. *Correlation coefficients of the team attributes on PI level*

explained by the fact that the team is new and started as a smaller team. The increase in the capacity with more members can result from the team getting larger at the same time as the team overcoming the struggles of starting the team.

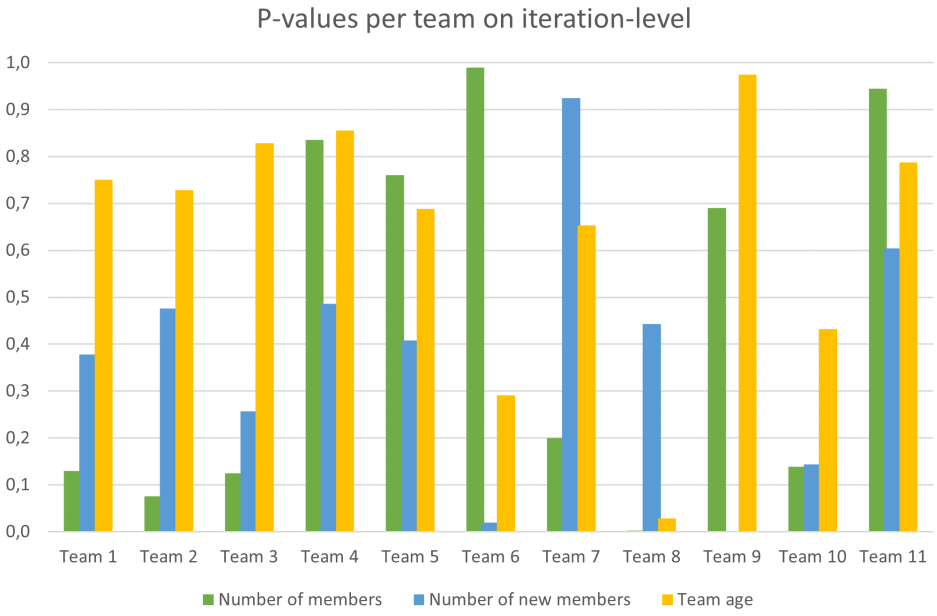

Figure 6.10. *Correlation coefficients of the team attributes on PI level*

Summary of the results

The significant results are summarised in table 6.8. For each relationship with the team capacity, the project-wide correlation is provided if its p-value is smaller than 0,05. Also, the number of teams for which a relationship was proven significant is shown.

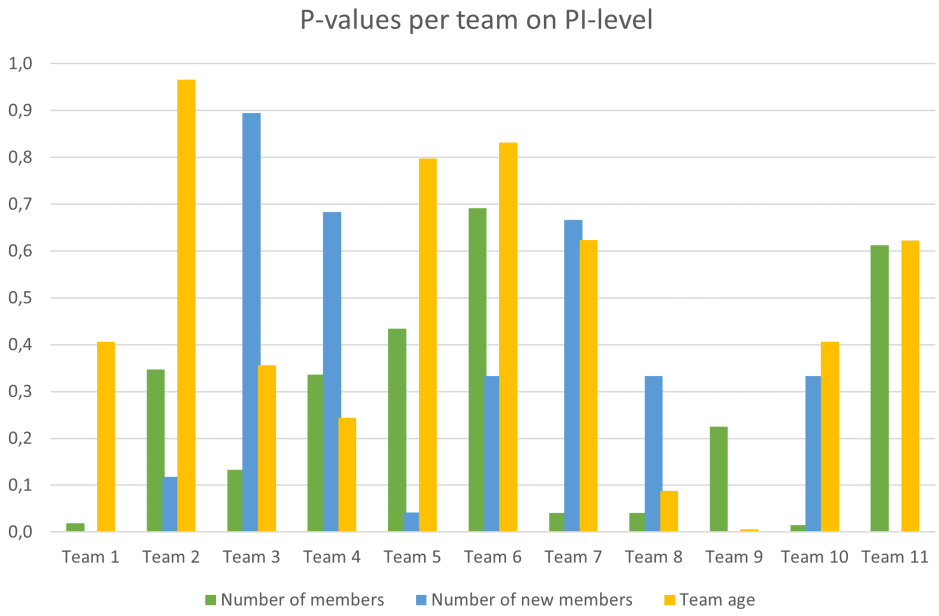

Figure 6.11. *Correlation coefficients of the team attributes on PI level*

Table 6.8. *Summary of the analysed relationships*

Relationship	**Iteration: Project level correlation**	**Iteration: No. of teams with significant relation-ships**	**PI: Project level correla-tion**	**PI: No. of teams with significant relationships**
Team availability	0,296	1	0,529	2
Sum of fault dura-tions	insignificant	0	insignificant	1
Number of faults created	-0,126	0	insignificant	0
Number of faults resolved	insignificant	0	insignificant	0
Maximum num-ber of faults open	insignificant	1	insignificant	0
Number of mem-bers	-0,253	1	-0,286	4
Number of new members	insignificant	1	insignificant	1
Team age	insignificant	1	insignificant	1

None of the relationships of team capacity was proven significant for the majority of the teams. The found correlations were mostly weak or moderate. Based on this study, team availability, team attributes, and corrective maintenance should not be used alone to estimate team capacity. The topic is much more complicated, and the relationships are unclear.

The factors shown to be significant on the project level are team availability, the number of faults created (only on the iteration level) and the number of new members. Out of these, the only strong correlation with team capacity was found with team availability. However, the p-values among the teams varied highly, and the relationship was significant for only 2 teams.

The number of members had a statistically significant moderate negative correlation for the whole project. The p-value of the relationship was below 0,05 for four teams, which was the highest number of teams for any of the relationships. In addition, the significant correlations calculated for teams on the PI level are strong, between -0,687 and -0,894.

The results were compared to the hypotheses. Null hypothesis 1 stated that team availability does not affect team capacity. However, team availability was found to be related to team capacity when looking at the project as a whole and for 2 teams on the PI level. Thus, the hypothesis can be rejected only on the project level and for 2 teams. The null hypothesis of no relationship between corrective maintenance and team capacity cannot be rejected. Only one of the maintenance summaries for the whole project had a p-value less than 0,05, and only one team had a significant correlation with another corrective maintenance summary. Hypotheses 3 to 5 were formed regarding the effect of team attributes. Out of these, only the null hypothesis 3, which states that team size is not related to team capacity, can be partly rejected. The alternate hypothesis 3 was supported on the project level and for 4 teams when looking at the data from PIs.

6.3 Analysis

The goals of this study were to explore which factors affect team capacity and find ways to improve team capacity estimations. The sizes of the impacts of the factors were also evaluated as a part of the data analysis. The main factors that affect team capacity are analysed in this section.

The results varied greatly between teams which was expected. Teams estimate their work and prepare their plans in different ways. Teams also have different areas of expertise, responsibility and experience, which can cause differences among teams. Mostly, the found correlations were higher on the PI level than on the iteration level.

SAFe capacity estimation relies mostly on the historical velocity data and the future team availability [2]. This basic idea was evaluated by analysing the relationship between team

availability and capacity in story points. Surprisingly, this relationship was only shown to be significant when looking at the whole project in total and for one separate team. Out of the 11 studied teams, 10 had a p-value of more than 0,05, and the null hypothesis of no relationship between team availability and capacity could not be rejected for the teams. Multiple teams showed very weak correlations between the factors, and for 2 teams, the correlations were negative. This result clearly shows that team capacity estimations cannot be done solely based on team availability. Pomar et al. acquired similar results in their case study [23]. Especially the teams with a low or negative correlation between the factors should analyse their capacity and factors affecting it in depth.

In SAFe, user stories are often estimated based on one story point being worth one day's work [2]. This was not found in the data analysis. The results indicate that the mean effect of increased availability is about 0,8 story points per added working week. This result is not necessarily a problem, it tells about how teams make their effort estimations. The results regarding team availability contradict how Leffingwell describes SAFe procedures [2].

The major factors identified as a part of the literature research are presented in the table 6.9. The major factors not chosen for the data analysis are also included in the table. The significance and impact of the factor are shown for the ones included in the analysis.

None of the studied relationships was visible for the majority of the teams, and most of the found relationships were weak or moderate. Team age and corrective maintenance were not found to affect team capacity. The insignificance of corrective maintenance on team capacity may be because the summaries used for corrective maintenance might not present the workload of the repair work. This risk was mitigated by using multiple different summaries of corrective maintenance. Team age was not explicitly mentioned as a team capacity factor in the literature. However, it was used to approximate team familiarity and maturity, which are considered major factors of team capacity [39] [26]. Team age may not be a valid representation of team familiarity, which could have resulted in the relationship found by Huckman et al. being invisible in the case company.

The impact of team size on team capacity was visible for four (36 %) studied teams. The finding was similar to the research of Schwaber et al, Ramasubbu et al. and Mohamed et al.. The number of members in a team was found to moderately hurt team capacity. Even though this relationship was significant for the highest number of teams among the studied relationships, it was still not visible for the majority of teams. Only a small number of teams had statistically significant results and enough evidence to support rejecting the null hypotheses for all of the studied relationships. This can be due to small correlations in the data, a small amount of team data or high variability in the data.

The second research question regards improving capacity estimations. Capacity estimation is complex and cannot be tackled with only team availability. This was shown by

Table 6.9. *Identified factors of team capacity*

Factor	Impact based on literature	Impact based on data
Team availability	Assumption of SAFe	Only significant on the project level, small impact
Team size	Significant after a threshold [22] [28] [29]	Negative moderate correlation
Corrective maintenance	Significant, repair work estimation is a widely studied topic [31] [32]	Not significant
Project complexity	Significant [22] [23]	Not analysed
Production environment	Not agreed on [27] [35]	Not analysed
Work in progress	Significant [19]	Not analysed
Technical debt	Significant [38]	Not analysed
Team climate	Significant, widely studied topic [16] [25]	Not analysed
Individual competencies	Significant [24] [27]	Not analysed
Team age	Similar factor is (team familiarity) is significant [39]	Team age not significant
Quality of plans	Little research [12]	Not analysed
Effort estimation	Significant, widely studied topic [14] [42]	Not analysed

the small number of significant relationships and especially the finding that there is insufficient evidence to demonstrate a relationship between team availability and capacity. The study showed that it is essential that teams analyse their capacity more in depth and the recommendations in chapter 7 are needed to improve capacity estimation. Software development teams should be provided more information and tools to estimate their capacities.

7. DISCUSSION

In this chapter, the proposed recommendations based on the results of the data analysis and the existing literature are presented. The recommendations aim to answer the research question of how to improve the capacity estimations of teams in the case company. The recommendations can be generalised to be helpful outside of the case company. The quality and limitations of the analysis are also presented in this chapter.

7.1 Recommendations

This study's results show that a team's capacity cannot be explained using only historical performance combined with future team availability. The process of capacity estimation should be improved. As Cohn stated, relying on historical velocity should only be done if the project, team and environment remain the same PI after PI, which is rarely the situation in the case company [19]. Recommendations were formed based on the literature and the results and limitations of the data analysis. The recommendations can be divided into two categories: recommendations that improve the process of estimating and recommendations that improve the historical data.

The recommendations to improve the capacity estimation are:

- Teams should be encouraged to analyse their past more and trained on what factors can affect team capacity.
- Capacity estimate should be defined, and a range of capacity used instead of one value.
- Teams should be provided with the data needed to analyse their capacity and capacity estimates.

The team capacity estimation process can be improved by educating the teams on the topic and providing them with the needed data. The teams should be encouraged to analyse their past more and be trained on the possible impacts of different factors and risks. Tanveer et al. and Usman et al. highlight the importance of providing the teams with a framework to support the estimation process [14] [42]. From this study, it is evident that there is a large number of possible factors and their effects vary between the teams. Still, teams can benefit from learning what factors to analyse and what they should consider

when estimating their capacity. The listing of factors of team capacity in this thesis can be used as a starting point for bringing the factors to teams' attention.

The two other recommendations for improving the process are more detailed than the first. The definition of capacity should be made clear. Jorgensen criticised that the definition of effort is poorly defined in the industry [13]. This is also relevant to the definition of capacity in the case company. Is the estimated capacity, for example, the maximum capacity or the most probable capacity? In any case, the usage of the estimation should depend on the definition, and the definition should be clarified. The risks of poor estimations could be mitigated using a range of capacity instead of one value suggested by Cohn [19]. The teams could estimate their capacity as a range of the worst, most likely, and best scenarios.

The teams should also be provided with the data needed to analyse their past. The teams should be given data from a longer time frame than the most recent PI so they could base their estimations on a more extensive set of data. The teams also need data on the accuracy of their past capacity estimates to improve them. To encourage and enable teams to analyse their capacity in detail, they should be provided with the needed data clearly and efficiently. To improve the usability of the data, the recommendations below are proposed.

The recommendations to improve team capacity data are:

- Teams should create user stories for their changing work.
- Teams should record the actual effort after PIs in addition to the estimates.

The two recommendations to improve data are needed to ensure that teams that use data to analyse their capacity can derive as much value from the analysis as possible. Improving the data also improves the simple capacity estimation by teams relying on only historical velocity. The need for these actions was recognised based on the limitations of the user story data in this study. If a team works on completing a task that is not presented as a user story, the team will show a lower velocity than it had in reality. Similarly, if the team works on a user story much less or much more than planned, the capacity is skewed by the difference between the actualised and estimated effort. This suggested practice should be used for exceptional work or by taking it into use slowly and consistently.

The teams should be encouraged to always create user stories for appearing or changing work. Estimating the related is also essential. This would allow the teams to see their real capacity and visualise the changes during the PI, and the management could get more information about the team's performance. This practice is already encouraged by the case company's management, but it could still be prioritised more.

Lastly, the teams should record actualised story points of their work to visualise their velocity better and adjust their capacity estimations on the actualised velocity. The data

could also be used to measure and track the team's estimation accuracy, as suggested by Tanveer et al. [42]. Using the actualised efforts to estimate capacity should be taken into use carefully as it could easily result in teams using it to give overly optimistic capacity estimations. When actual effort is recorded, teams can see how their estimations differ from reality and improve their estimations.

At the beginning of adapting the suggested new practices, the likelihood and effect of teams using these recommended practices to overestimate their capacity should be minimised. Suddenly creating user stories for all appearing work and including these in the team capacity might result in recording work that previously happened outside the plans and result in committing to too much work. Teams could also be tempted to increase the actual effort of some user stories if they are eager to increase their capacity estimation to commit to new work. The danger of overcommitting due to these suggested practices can be decreased by starting them as only a way to gain more information about capacity and efforts instead of basing decisions and capacity estimations on them. Recording the appearing and changing works can provide valuable insights into the team's work and effort estimations even without including them in capacity estimation calculations.

7.2 Evaluation of the quality of the analysis

The results of the data analysis vary greatly between the teams, and they are partly not in line with the field literature. Some of the results were even surprising, for example, few teams had a positive correlation between corrective maintenance and team capacity. This can be due to many reasons. The data used can be unreliable, the tools used to study too simple to detect a complex relationship, or the relations studied might be non-existent or complicated. Also, as two different types of correlation, Pearson and Spearman, were used, comparing the different correlations is less reliable than if only one type was used. This had to be done because of the non-normal data.

This study was done for one case company where the teams have existed for a maximum of two years. One limitation of the analysis was the limited kinds of data available. The available data limited the factors that could be studied in the data analysis. Some interesting factors, such as the quality of team communication and team dispersion, had to be left out of the data analysis because data on them was unavailable. Other data-related issues are the mentioned issues of user stories. The closure dates of user stories do not tell when most of the body of the work was done, the story points are only estimations, and a large amount of work might be undocumented. Also, the team availability used was planned availability instead of actual availability.

The importance of the results on the project level treating all of its teams as a whole is not evident. The result simplifies the project, provides information about it as a whole and performs analysis on much more extensive data than teams independently. However, as

mentioned before, teams do their estimations and work differently, which can impact the results. The corrective maintenance workload was also presented by summaries, and their meaningfulness was not separately evaluated.

Another limitation was the amount of data available. As data is only recorded for PIs and iterations, the amount of data was low, especially at the PI level. Some relatively new teams only had 4-6 entries which is undeniably a very low number. Also, the analysis could have provided more clear results if the structures of the teams were more stabilised.

Overall, the results show that capacity estimation cannot be based solely on historical velocity and team availability. They also show that team capacity is a complex topic. However, the reliability of the detailed results can be questioned due to the small amount of data.

8. CONCLUSIONS

The final chapter concludes the findings of this study. The identified factors of team capacity are revisited, and the results of the data analysis are summarised. Lastly, the recommendations are listed to answer the research question on improving team capacity estimation.

Team capacity is estimated as a part of the PI planning by the software development teams. It is used to determine how many story points the team can commit to working during the next PI, and it is measured in story points. The goals of this case study were to identify which factors affect the team capacity and how capacity estimations could be improved in the case company. The factors were first identified from existing literature, and part of them were analysed in depth with data analysis methods. The purpose of the data analysis was to analyse the significance and impact of the chosen factor in the context of the case company. Lastly, recommendations were proposed based on the literature findings and data analysis results.

Team capacity is affected by the performance of the team and by the team's plans. Team performance is the amount of work that a team can complete in a period. However, as team capacity is measured in the story points that the team completed, only the work that was planned, documented, and estimated counts towards team capacity. Thus, team capacity is affected by the way a team adheres to its plans in addition to the team's performance.

Factors of team performance can be divided into two categories: technical and soft factors. Lately, there has been a large amount of research on the soft factors. The technical factors that are undisputed in the existing literature are team availability, team size, corrective maintenance, project complexity, amount of work in progress, and technical debt. The significance of the production environment as a factor of team performance is under discussion. The soft factors of team capacity are team climate, team familiarity and individual competencies. The factors that affect the impact of the team's plans on team capacity are the quality of plans and effort estimations. Effort estimation is a widely studied topic, while the research on the overall quality of plans is lacking.

The relationships between team capacity and team availability, corrective maintenance, and team attributes were studied in the case company. These factors were chosen for the

analysis based on the reliability and simplicity of the related data from the case company, as well as the relevance of the factor. The factors were analysed on the iteration and PI levels. The results summarised here are on the PI level, as the relationships were more apparent for PIs than for iterations.

The relationship between team availability and team capacity was only visible for two teams, while the overall correlation for all teams combined was 0,53. While team capacity estimation in SAFe largely relies on the assumption that team capacity depends on team availability, in the case company, the data of 9 teams did not show the relationship. The major role of team availability in the estimations should be questioned. So far, this has not been done in the field research.

The relationship between corrective maintenance and team capacity was studied with three different summaries of the corrective maintenance effort. None of them had a statistically significant and correlating relationship with team capacity. Surprisingly, the amount of fault fixes that the teams do during a PI does not seem to affect the capacity of the team. Out of the analysed team attributes (team size, team age i.e. team familiarity, and new team members), only the team size has a statistically significant relationship with team capacity. For 4 teams, team size correlates significantly with team capacity making team size the most apparent factor of team capacity in the data.

The results of the data analysis showed that the team capacity cannot be estimated by relying only on the team's availability. Team capacity is affected by many factors, and the impacts of the factors are difficult to detect in the resulting capacity. Overall, capacity estimation is a complex task. The following recommendations for improving capacity estimations were proposed in this thesis:

- Teams should be encouraged to analyse their past more and trained on what factors can affect team capacity.
- Capacity estimate should be defined, and a range of capacity used instead of one value.
- Teams should be provided with the data needed to analyse their capacity and capacity estimates.
- Teams should create user stories for their changing work.
- Teams should record the actual effort after PIs in addition to the estimates.

Capacity estimation can be improved by supporting and guiding the teams in the process and by improving the data used in the estimation. Improved accuracy of capacity estimation can result in better predictability in the project.

REFERENCES

[1] *15th Sate of Agile Report: Agile adoption accelerates accross the enterprise*. Digital.ai, 2021. URL: https://digital.ai/resource-center/analyst-reports/state-of-agile-report (visited on 06/10/2022).

[2] Leffingwell, D. *SAFe 4.5 Reference Guide: Scaled Agile Framework for Lean Enterprises, Second edition*. 2018. URL: https://learning.oreilly.com/library/view/safe-4-5-reference/9780134892917/pref01.xhtml (visited on 05/19/2022).

[3] Dima, A. M. and Maassen, M. A. From Waterfall to Agile software: Development models in the IT sector, 2006 to 2018. Impacts on company management. *Journal of International Studies* 11 (June 30, 2018), pp. 315–326. URL: http://www.jois.eu/?438,en_from-waterfall-to-agile-software-development-models-in-the-it-sector-2006-to-2018.-impacts-on-company-management (visited on 06/08/2022).

[4] Royce, D. W. W. Managing the development of large software systems. (1970), p. 11.

[5] Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D. *Manifesto for Agile Software Development*. 2001. URL: http://www.agilemanifesto.org/.

[6] Ashmore, S. and Runyan, K. *Introduction to Agile Methods*. Addison-Wesley Professional, 2014. URL: https://learning.oreilly.com/library/view/introduction-to-agile/9780133435245/ (visited on 06/09/2022).

[7] Gullo, D. J. *Real World Agility: Practical Guidance for Agile Practitioners*. 2016. URL: https://learning.oreilly.com/library/view/real-world-agility/9780134191768/ (visited on 06/09/2022).

[8] Laanti, M. Characteristics and Principles of Scaled Agile. Lecture Notes in Business Information Processing. Vol. 199. May 26, 2014.

[9] Putta, A., Paasivaara, M. and Lassenius, C. *Benefits and Challenges of Adopting the Scaled Agile Framework (SAFe): Preliminary Results from a Multivocal Literature Review*. Vol. 11271. Lecture Notes in Computer Science. Springer International Publishing, 2018. URL: http://link.springer.com/10.1007/978-3-030-03673-7 (visited on 06/08/2022).

[10] Laanti, M. and Kettunen, P. SAFe Adoptions in Finland: A Survey Research. *Agile Processes in Software Engineering and Extreme Programming – Workshops*. Lec-

ture Notes in Business Information Processing. Springer International Publishing, 2019, pp. 81–87.

[11] Agarwal, M. and Majumdar, P. R. *Tracking Scrum projects Tools, Metrics and Myths About Agile*. 2012.

[12] Badampudi, D. Factors Affecting Efficiency of Agile Planning: A Case Study. (2012), p. 98.

[13] Jørgensen, M. A Critique of How We Measure and Interpret the Accuracy of Software Development Effort Estimation. (2007), p. 6.

[14] Usman, M., Mendes, E. and Börstler, J. Effort estimation in agile software development: A survey on the state of the practice. Apr. 27, 2015, pp. 82–91.

[15] Blackburn, J. D., Scudder, G. D. and Van Wassenhove, L. N. Improving speed and productivity of software development: a global survey of software developers. *IEEE transactions on software engineering* 22 (1996), pp. 875–885. URL: https://ieeexplore.ieee.org/document/553636 (visited on 06/05/2022).

[16] Dingsøyr, T., Fægri, T. E., Dybå, T., Haugset, B. and Lindsjørn, Y. Team Performance in Software Development: Research Results versus Agile Principles. *IEEE Software* 33 (July 2016), pp. 106–110.

[17] Coelho, E. and Basu, A. Effort Estimation in Agile Software Development using Story Points. *International Journal of Applied Information Systems* 3 (Sept. 1, 2012).

[18] Ahmed, A. R., Tayyab, M., Bhatti, S. N., Alzahrani, A. J. and Babar, M. I. Impact of Story Point Estimation on Product using Metrics in Scrum Development Process. *International Journal of Advanced Computer Science and Applications* 8 (2017). URL: http://www.proquest.com/docview/2656452091/abstract/5A3336092A0B423CPQ/1 (visited on 05/19/2022).

[19] Cohn, M. *Agile Estimating and Planning*. 2006. URL: https://learning.oreilly.com/library/view/agile-estimating-and/9780137126347/ (visited on 05/23/2022).

[20] McConnell, S. Software Project Survival Guide. *Microsoft Press* (1998), p. 35.

[21] Mcdaid, K., Greer, D., Keenan, F., Prior, P., Coleman, G. and Taylor, P. Managing Uncertainty in Agile Release Planning. 18th International Conference on Software Engineering and Knowledge Engineering, SEKE 2006. Jan. 1, 2006, pp. 138–143.

[22] Schwaber, K. *Agile Project Management with Scrum*. 2004. URL: https://learning.oreilly.com/library/view/agile-project-management/9780735619937/ (visited on 06/05/2022).

[23] Albero Pomar, F., Calvo-Manzano, J. A., Caballero, E. and Arcilla-Cobián, M. Understanding sprint velocity fluctuations for improved project plans with Scrum: a case study. *Journal of Software: Evolution and Process* 26.9 (2014), pp. 776–783. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.1661 (visited on 04/06/2022).

[24] Ong, A., Tan, G. W. and Kankanhalli, A. Team Expertise and Performance in Information Systems Development Projects. (2005), p. 7.

[25] Purna, S. G., Farooq, A. and Patnaik, S. Soft factors affecting the performance of software development teams. *Team Performance Management: An International Journal* 17.3 (Jan. 1, 2011), pp. 187–205. URL: https://doi.org/10.1108/13527591111143718 (visited on 04/06/2022).

[26] Al-Sabbagh, K. W. and Gren, L. The connections between group maturity, software development velocity, and planning effectiveness. *Journal of Software: Evolution and Process* 30 (2018). URL: http://onlinelibrary.wiley.com/doi/abs/10.1002/smr.1896 (visited on 04/06/2022).

[27] Asproni, G. Motivation, Teamwork, and Agile Development. 4 (2004), p. 9.

[28] Ramasubbu, N. and Balan, R. K. Globally distributed software development project performance: an empirical analysis. *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering - ESEC-FSE '07*. the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium. ACM Press, 2007, p. 125. URL: http://portal.acm.org/citation.cfm?doid=1287624.1287643 (visited on 04/06/2022).

[29] Mohamed, S. I. Comprehensive Measurement Analysis for Software Productivity-IJSE. (2015). URL: https://www.academia.edu/32117009/Comprehensive_Measurement_Analysis_for_Software_Productivity_IJSE (visited on 05/19/2022).

[30] Faraj, S. and Sproull, L. Coordinating Expertise in Software Development Teams. *Management Science* 46 (Dec. 1, 2000), pp. 1554–1568.

[31] Mockus, A., Weiss, D. and Ping Zhang. Understanding and predicting effort in software projects. *25th International Conference on Software Engineering, 2003. Proceedings.* 25th International Conference on Software Engineering, 2003. Proceedings. IEEE, 2003, pp. 274–284. URL: http://ieeexplore.ieee.org/document/1201207/ (visited on 09/10/2021).

[32] Goel, B. and Singh, Y. An Empirical Analysis of Metrics to Predict the Software Defect Fix-Effort. *International Journal of Computers and Applications* 33 (Jan. 1, 2011), pp. 124–131. URL: https://www.tandfonline.com/doi/abs/10.2316/Journal.202.2011.2.202-2749 (visited on 06/07/2022).

[33] Jørgensen, M. A review of studies on expert estimation of software development effort. *Journal of Systems and Software* 70.1 (Feb. 1, 2004), pp. 37–60. URL: https://www.sciencedirect.com/science/article/pii/S0164121202001565 (visited on 06/07/2022).

[34] Maroukian, K. IT Project Environment Factors Affecting Requirements Analysis in Service Provisioning for the Greek Banking Sector. 3 (Sept. 30, 2010). URL: http:

//www.scirp.org/Journal/Paperabs.aspx?paperid=2676 (visited on 05/22/2022).

[35] Sawyer, S. and Guinan, P. J. Software development: Processes and performance. *IBM Systems Journal* 37 (1998), pp. 552–569. URL: https://www.academia.edu/305229/Software_Development_Processes_and_Performance (visited on 05/22/2022).

[36] Kniberg, H. *Scrum and XP from the Trenches - 2nd Edition*. 2007. URL: https://www.infoq.com/minibooks/scrum-xp-from-the-trenches-2/ (visited on 05/29/2022).

[37] Sterling, C. *Managing Software Debt: Building for Inevitable Change*. Addison-Wesley Professional, 2010.

[38] Power, K. Understanding the impact of technical debt on the capacity and velocity of teams and organizations: Viewing team and organization capacity as a portfolio of real options. *2013 4th International Workshop on Managing Technical Debt (MTD)*. 2013 4th International Workshop on Managing Technical Debt (MTD). May 2013, pp. 28–31.

[39] Huckman, R. S., Staats, B. R. and Upton, D. M. Team Familiarity, Role Experience, and Performance: Evidence from Indian Software Services. *Management Science* 55 (Jan. 2009), pp. 85–100. URL: https://pubsonline.informs.org/doi/10.1287/mnsc.1080.0921 (visited on 04/06/2022).

[40] Maria Purba, S. E., Simaremare, M. E. S., Hasibuan, R. D. and Tambun, M. D. R. Measuring the Individual Performance of A Software Development Team. *2021 8th International Conference on Computer and Communication Engineering (ICCCE)*. 2021 8th International Conference on Computer and Communication Engineering (ICCCE). June 2021, pp. 78–81.

[41] Ramessur, M. A. and Nagowah, S. D. Factors Affecting Sprint Effort Estimation. *Advanced Computing and Intelligent Engineering*. Springer, 2020, pp. 507–518.

[42] Tanveer, B., Guzmán, L. and Engel, U. M. Effort estimation in agile software development: Case study and improvement framework. *Journal of Software: Evolution and Process* 29 (2017), e1862. URL: http://onlinelibrary.wiley.com/doi/abs/10.1002/smr.1862 (visited on 05/21/2022).

[43] Haugen, N. C. An Empirical Study of Using Planning Poker for User Story Estimation. *Proceedings of the Conference on AGILE 2006*. IEEE Computer Society, 2006, pp. 23–34. URL: https://doi.org/10.1109/AGILE.2006.16.

[44] Usman, M., Mendes, E., Weidt, F. and Britto, R. Effort estimation in agile software development: a systematic literature review. *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*. PROMISE '14: The 10th International Conference on Predictive Models in Software Engineering. ACM, Sept. 17, 2014, pp. 82–91. URL: https://dl.acm.org/doi/10.1145/2639490.2639503 (visited on 06/06/2022).

[45] *Who uses Jira?* URL: https://www.atlassian.com/software/jira/guides/use-cases/who-uses-jira.

[46] *PI objectives*. URL: https://www.scaledagileframework.com/pi-objectives/.

[47] *PI planning*. URL: https://www.scaledagileframework.com/pi-planning/.

[48] *Pandas documentation*. URL: https://pandas.pydata.org/docs/index.html.

[49] *Seaborn*. URL: https://seaborn.pydata.org/index.html.