



Handelshögskolan

Karlstad Business School

Laura-Liisa Lillemäe

Key Performance Indicators for Measuring Efficiency and Effectiveness in an Agile Software Development Team

Development of a KPI Dashboard for Measuring
Individuals' Performance Through an
Entrepreneurial Study within Extenda Retail

Information Systems

Bachelor's thesis

Semester: HT-2024
Supervisor: Odd Fredriksson
Examinator: John Sören Pettersson

Abstract

Measuring individual performance in agile environments presents unique challenges, yet it holds the potential to drive growth. This study aims to identify and apply Key performance indicators (KPIs) to offer a clearer picture of individual contributions and areas for improvement within agile software development.

The main purpose of this bachelor's thesis in Information Systems is to identify and describe key performance indicators for measuring efficiency and effectiveness in an agile software development team, from individual team members' perspective.

The sub-purpose is to develop a KPI dashboard that measures individuals' performance based on data from a project management tool.

Using a mixed-method approach that integrates an entrepreneurial approach with an action research approach, iteratively a KPI dashboard is developed. SPIN interviews were conducted to gather insights into the degree of performance measurement challenges and degree of needs of different stakeholders, while the action research approach guided the prototyping process through iterative cycles of diagnosing, planning, action taking, and evaluation.

The first conclusion of this study is that individual Velocity, Cycle time, and Change failure rate are, to a higher degree, effective KPIs for measuring the degree of efficiency and degree of effectiveness in agile teams. Cycle Time, in particular, helps identify workflow bottlenecks that are often missed when relying solely on Velocity. The second conclusion is that a balanced approach, combining both individual and team-level KPIs, is preferred by participants. The third conclusion is that the SPACE framework, while more useful in theory, requires adjustments when applied in practice due to challenges with the degree of data availability and quality, particularly for objective KPIs. The fourth conclusion highlights the importance of a higher degree of data quality for more reliable performance insights.

Keywords: Performance measurement, Key performance indicators, Efficiency, Effectiveness, Agile team, Scrum team, Individual performance, SPACE framework, dashboard

Preface

This thesis would not have been possible without the incredible support I've received from those around me. I would like to extend my deepest thanks to my supervisor, Odd Fredriksson, whose guidance and expertise helped me navigate this journey. Your encouragement has been invaluable.

A special thank you goes to team C, whose insights shaped the direction of this research. Your willingness to engage and share your knowledge has been a true privilege.

Lastly, I am grateful to my family for their unwavering support and patience throughout this process. Your belief in me has kept me motivated, and I can't thank you enough.

Thank you to everyone who contributed to this work in one way or another. I couldn't have done it without you!

Laura -Liisa Lillemäe

Table of Contents

1	Introduction.....	1
1.1	<i>Problem background</i>	1
1.2	<i>Purpose.....</i>	2
1.3	<i>Target audience.....</i>	2
1.4	<i>The case firm</i>	2
2	Theory and Earlier Studies	3
2.1	<i>Performance measurement in agile software development teams.....</i>	3
2.2	<i>Efficiency KPIs for performance</i>	3
2.3	<i>Effectiveness KPIs for performance</i>	5
2.4	<i>The SPACE framework</i>	8
2.5	<i>Measuring individual performance.....</i>	10
2.6	<i>Selecting and adopting KPIs</i>	10
2.7	<i>Benefits and drawbacks of dashboard implementation.....</i>	11
2.8	<i>Design thinking.....</i>	12
3	Method.....	13
3.1	<i>Approach and chosen methods</i>	13
3.2	<i>Selection of interview participants</i>	13
3.3	<i>The case study method.....</i>	15
3.4	<i>Entrepreneurial approach.....</i>	15
3.5	<i>Action research approach.....</i>	15
3.5.1	<i>Diagnosing.....</i>	16
3.5.2	<i>Action planning</i>	16
3.5.2.1	<i>Identifying KPIs</i>	17
3.5.2.2	<i>Dimensions</i>	17
3.5.3	<i>Action taking</i>	17
3.5.3.1	<i>Tools and technology.....</i>	17
3.5.3.2	<i>Dashboard designing and prototyping</i>	18
3.5.3.3	<i>Challenges.....</i>	19
3.5.4	<i>Evaluating.....</i>	19
3.6	<i>Theory and earlier studies</i>	19
3.7	<i>Degree of validity and reliability of the study.....</i>	20
3.8	<i>Degree of empirical saturation.....</i>	20
3.9	<i>Ethical considerations and GDPR.....</i>	20
4	Results.....	22
4.1	<i>Results from SPIN interviews</i>	22
4.2	<i>Prototype</i>	31

4.3	MVP 1	40
4.4	MVP 2	44
4.5	MVP 3	47
5	Analysis	49
5.0	Key performance indicators.....	49
5.0	SPACE framework	50
5.0	The degree of need for performance measurement.....	51
5.0	Measuring on individual level versus team level	52
5.0	The impact of data quality on KPI reliability and actionability.....	53
6	Conclusions.....	54
6.0	Deeper contextual analysis enhances the meaning of KPIs.....	54
6.0	Measuring individual performance offers both perceived opportunities and risks	54
6.0	The SPACE framework works in theory, but needs adaptations in practice.....	55
6.0	The degree of data quality is paramount for KPIs to be more actionable.....	55
6.0	Contributions of the study and future studies	55
7	Recommendations.....	57
	List of references.....	58
	Written sources.....	58
	Oral sources	61
	Appendices	63
	Appendix 1: Probing dialogue 1.....	63
	Appendix 2: Probing dialogue 2.....	64
	Appendix 3: Brainstorming with Team Lead.....	66
	Appendix 4: SPIN-Interview guide for Engineering Manager/Team Lead	67
	Appendix 5: SPIN-Interview guide for Developer	68
	Appendix 6: SPIN-Interview guide for Director of Human Resources (HR).....	69
	Appendix 7: Protocol of SPIN-Interview with Engineering Manager	70
	Appendix 8: Protocol of SPIN-Interview with Software developer.....	75
	Appendix 9: Protocol of SPIN-Interview with Software engineer.....	79
	Appendix 10: Protocol of SPIN Interview with Director of Human Resources (HR).....	81
	Appendix 11: Protocol of SPIN Interview with Team lead A	85
	Appendix 12: Example of a Custom Calculated Measure in EazyBI	89
	Appendix 13: Protocol of MVP 1 Test with Senior Frontend Developer.....	90
	Appendix 14: Protocol of MVP 1 Test with Team lead B.....	92
	Appendix 15: Protocol of MVP 1 Test with Engineering Manager.....	93
	Appendix 16: Protocol of MVP 1 Test with Software Engineer	95
	Appendix 17: Protocol of MVP 1 Test with Team lead A.....	96

<i>Appendix 18: Protocol of MVP 2 Test with Team lead B.....</i>	<i>99</i>
<i>Appendix 19: Protocol of MVP 2 Test with Software Engineer</i>	<i>101</i>
<i>Appendix 20: Protocol of MVP 2 Test with Senior Frontend Developer</i>	<i>102</i>
<i>Appendix 21: Protocol of MVP 2 Test with Engineering Manager.....</i>	<i>103</i>
<i>Appendix 22: Protocol of MVP 2 Test with Team lead A.....</i>	<i>104</i>
<i>Appendix 23: Protocol of MVP 3 Test with Senior Frontend Developer</i>	<i>105</i>
<i>Appendix 24: Protocol of MVP 3 Test with Software Engineer</i>	<i>106</i>
<i>Appendix 25: Protocol of MVP 3 Test with Team lead B.....</i>	<i>107</i>
<i>Appendix 26: Protocol of MVP 3 Test with Engineering Manager.....</i>	<i>108</i>
<i>Appendix 27: Protocol of MVP 3 Test with Team lead A.....</i>	<i>109</i>

1 Introduction

1.1 Problem background

Agile methodologies and iterative approaches to project delivery have been around for a long time but have increased sharply in popularity in recent years, even outside of software development, according to Whiteley et al. (2021:23). In 2001, agile software development was introduced, a collection of methodologies built on iterative and incremental concepts. The basic principles of agile methodologies are valuing the importance of individuals and interactions over processes and tools, functioning software over extensive documentation, customer collaboration over contract negotiations, and response to change over following a plan (Beck et al. 2001). Despite the widespread use of agile methods, there is a problem with the lack of clearly defined ways to measure performance in agile software development processes. KPIs such as Function Points and Lines of Code have been widely used but have received a lot of criticism (Shah et al. 2015:103). McKinsey (2023) states that there is still a problem with the belief that software development is too complex to be measured accurately, which has led to the area being left undermeasured. However, this is becoming unsustainable as the dependency on software across all industries grows, and leaders need better insights into how efficiently and effectively their development teams are performing. It's the highly collaborative and creative process that makes the standard effectiveness measurement a problem. Different levels- systems, teams, and individuals require unique KPIs to account for both collaborative work and individual contributions. McKinsey (2023) means that a KPI like deployment frequency, useful at a system or team level, doesn't translate well for individual assessment.

Another ongoing problem is the shortage of skilled developers versus the high demand for new software, which, according to CIO (2023), is a source of interest among organizations in measuring developer effectiveness. CIO (2023) quotes Keith Mann, senior director and analyst at Gartner, saying, "Gartner's surveys and data from client inquiries confirm that developer effectiveness remains a top priority for software engineering leaders."

Although new KPIs are not always needed, they are necessary in areas where agile principles affect project management (Shah et al. 2015:103). As noted by (Shah et al. 2015:103), time, quality, quantity, and customer satisfaction are important factors. Despite these insights, there is a lack of consensus on appropriate measures for performance in agile contexts.

KPIs are useful indications for evaluating and optimizing processes, projects, products, and people that are part of software development teams (Budacu & Pocatilu 2018:71, Machuca-Villegas et al. 2021:59, Natarajan & Pichai 2024:2). These are then used to make decisions, run projects, and improve software development as well as project management processes. In addition to identifying areas for improvement by performance measurement, support can be provided as needed to strengthen both the individuals' and the team's overall performance, according to Davis (2015). In this way, software development teams take responsibility for tracking themselves through KPIs that are easy to obtain and communicate.

The studied case firm is Extenda Retail AB, which is active in software development for the retail and logistics industry. The case firm has expressed a desire to have a tool to visualize performance in terms of efficiency and effectiveness, based on data from the project management tool Jira (See Appendix 1 & Appendix 2). The identified problem is the current lack of performance measurement in the software development teams at the case firm, thus the need for a tool that measures the performance of the software development teams. At present, it is perceived as difficult to identify deviations and problems in the development process. The tool's desired effect is to provide a better insights into how teams and individuals are performing over time (See Appendix 1). There should also be an opportunity to detect when someone may be struggling with their workload or having technical problems. This study will focus solely on the individual level.

1.2 Purpose

The main purpose of this bachelor's thesis in Information Systems is to identify and describe key performance indicators for measuring efficiency and effectiveness in an agile software development team, from individual team members' perspective.

The sub-purpose is to develop a KPI dashboard that measures individuals' performance based on data from a project management tool.

The two purposes are related to each other. Identifying and describing the KPIs help to establish what should be measured and ensures that the focus is on meaningful aspects of performance. The sub-purpose operationalizes the identified KPIs by creating a dashboard, which relies on the KPIs identified in the first purpose. These purposes bring together theory and practice, with the aim to enhance knowledge about measuring individual's performance in agile software development.

1.3 Target audience

The primary target audience is the case firm Extenda Retail. The secondary target audience is members of agile teams or other agile software development practitioners who may benefit from the content of identifying KPIs for agile team performance or developing a KPI dashboard. Additionally, a target audience is academic researchers focused on performance measurement.

1.4 The case firm

Extenda Retail is a leading provider of retail and warehouse management software solutions, with over 40 years of experience (Extenda retail 2025a). Extenda retail (2025b) offers a comprehensive suite of products designed to streamline retail operations and enhance customer experiences, including advanced Point of Sale (POS) systems, warehouse management tools, and loss prevention software. The software development teams at Extenda retail work according to Agile and Scrum (see Appendix 7).

2 Theory and Earlier Studies

The keywords in this study form the framework for the theoretical perspectives and research studies chosen to create a solid background and understanding of the problem area. The keywords have been based on chapter one.

2.1 Performance measurement in agile software development teams

Defining follow-up KPIs before beginning project management is the first step to ensure success, according to Almeida and Carneiro (2023:2). The KPIs serve several key purposes: improving efficiency, monitoring progress, planning future sprints, and measuring customer satisfaction (Pham and Neumann 2024:7). However, the importance of KPIs vary depending on the phase of Scrum (Almeida & Carneiro 2023:9), which is an agile methodology that uses an incremental and iterative process to identify priority tasks in each phase and effectively manage time (Almeida & Carneiro 2023:2). KPIs related to the daily execution phases of Scrum (daily scrum) are not as important as those related to planning (e.g., sprint planning meeting and sprint retrospective). However, Almeida and Carneiro (2023:2) emphasize that besides planning, monitoring the performance of processes and teams are necessary for projects developed in Scrum. By using KPIs, teams can be guided to a higher degree of effectiveness and necessary improvements can be made in the software development process during execution of a project.

In software development, KPIs are often focused on output-oriented KPIs, according to Budacu and Pocatilu (2018:70). Ramírez-Mora and Oktaba (2017:52) identified in their study that the most commonly used KPIs for efficiency measurement is the amount of software developed and the time it took to develop the software. Only a minority of the studies measured quality, which was measured by defects and the time it took to fix them. However, Ramírez-Mora and Oktaba (2017:52) point to the significance of delivering software free from rework and products that meet stakeholder expectations as key factors in perceived efficiency.

Developer efficiency plays a key role in the success of software development organizations (Oliveira et al. 2020:1). Organizations usually neglect the usage of KPIs, even though it can assist a leader, which leads to the leader sticking to subjective perceptions only. Oliveira et al. (2020:2) mean that project managers often rely primarily on team leaders' insights when making decisions about development teams. Therefore, the perception of developers' efficiency is only based on observation, which can be biased.

2.2 Efficiency KPIs for performance

When it comes to agile teams, efficiency often relates to resource consumption, timeliness, and meeting budget and schedule goals (Ramírez-Mora et al. 2019:3).

Lines of Code (LOC) is a KPI that has been used frequently in software development teams, where the amount of lines of code written by a developer is measured (Shah et al. 2015:105, Forsgren et al. 2021:18). LOC has been measured differently within different organizations, such as by individual, team, and hour (Shah et al. 2015:104). However, there are limitations to using LOC as an effectiveness measure since it does not account for the complexity or quality of the code, nor the time and effort involved in planning and thinking (Shah et al. 2015:105, Forsgren et al. 2021:18). Riihimäki (2024:12) means that it can also encourage developers to write less concise and lower-quality code.

A similar KPI is *Halstead effort by time*, which goes beyond simple code volume and considers the cognitive effort involved in writing code (Oliveira et al. 2020:2). The calculation for this KPI is based on the number of operators and operands used in the code and provides a more nuanced view of the complexity of the work produced. While it is more complex to calculate than Lines of Code, the Halstead effort by time KPI gives a deeper understanding.

Code Owned by Time is a KPI that represents the number of source code files a developer owns, meaning files that the developer has contributed the most to (Oliveira et al. 2020:6). This KPI is used to understand developer effectiveness but also to identify the key developers in a team who are essential to a software project's development. It is however important to note that some team leads from the study conducted by Oliveira et al. (2020:19) found that this KPI does not always accurately reflect developer effectiveness.

Oliveira et al. (2020:3) also write about commit-based KPIs like *Commits Performed by Time*, measuring the number of commit operations a developer performs over a period of time. The higher number of commits indicates higher effectiveness amongst developers. However, this KPI alone as an indicator of effectiveness can be inaccurate since commit policies can vary across projects. For example, one developer might make a single commit after producing a large amount of code, while another might make multiple commits for the same amount of work. This KPI is practical in projects where developers follow strict and consistent commit policies.

The second commit-based KPI is *Committed Source Lines of Code by Time*, which measures the number of lines of code developed in each commit by a developer over a given timeframe, assuming that more lines of code changed in a commit indicate more work done (Oliveira et al. 2020:3).

The third one is *Committed Characters by Time*, similar to the previous one, this KPI aims to provide a measurement of effectiveness by considering the number of characters changed in each commit (Oliveira et al. 2020:3). This is calculated using Levenshtein's edit distance, which measures the difference between two sequences of characters. The rationale is that sometimes developers make small changes, such as adding comments, that affect many lines but only involve changing a few characters. Committed Characters by Time is designed to reduce false positives that might arise from simply counting lines changed.

The most influential KPI for measuring effectiveness is *Velocity* (Budacu & Pocatilu 2018:71, Pham & Neumann 2024:6). Natarajan and Pichai (2024:3) also advocate, among other things, Velocity as an appropriate KPI. Velocity is measured by the amount of work a team can complete by sprint, and usually, story points are the foundation for this calculation as they are added up in a completed sprint (Pham & Neumann 2024:6). Although, Velocity can also be measured over any period of time and does not have to be by sprint (Menezes et al. 2024:9). The Velocity measure then helps the teams predict what their capacity is and what realistic goals to set in the upcoming sprints based on the average amount they typically complete.

Story points are assigned based on how complex, uncertain, and risky a task is instead of estimating a task by time (Pham & Neumann 2024:6). The main advantages of Story points are flexibility and adaptability to changes and uncertainties in software development. However, there is a risk that story points are subjective, which can hinder the accurate comparison of Velocity.

Menezes et al. (2024:9) also present a KPI called Velocity deviation, that measures Velocity stability across multiple sprints by calculating the ratio between the standard deviation of Velocity, which reflects how much the Velocity fluctuates, and the average Velocity based on previous sprints. A lower value of this ratio indicates greater stability, meaning the team is delivering at a more consistent pace. A higher value suggests more variation in Velocity, which could indicate unpredictability in the performance.

While Velocity is a popular KPI, the KPI should not be used in isolation, and should therefore be combined with other KPIs (Almeida & Carneiro 2023:10). The recommended KPIs to combine Velocity with are for example *Cycle time* and *Lead time*. This multi-faceted approach to measuring Velocity provides a comprehensive understanding of team efficiency and helps identify potential issues like overburdened or underutilized resources. Similarly, Microsoft noted that Velocity alone cannot explain the reasons behind variations in the development process, proving its insufficiency as a standalone KPI (Teiber 2021:57).

Pham and Neumann (2024:5) also found that Cycle time is a common KPI. It measures how long work items stay in specific workflow states, tracked in days, to understand the flow of work through the process (Power 2014:5). This KPI is beneficial for teams having problems with a large amount of work in progress when there is a lack of visibility into progress, or the team autonomy is ineffective. Measuring and visualizing the Cycle time KPI helps track duration in workflow status, identify delays, and improve processes. The result of this would be improved team collaboration, reduced work in progress, and better focus on completing work efficiently.

Additionally, by measuring Cycle time it is possible to predict future work items that are similar to each other, which benefits future planning and coordination dependencies across teams (Power 2014:6).

In contrast, Lead time measures the overall duration it takes for individual work items to progress through the entire workflow, from the initial request, idea, or report to the point where the item is delivered to the end users (Power 2014:6). This KPI provides an end-to-end perspective. It is measured in days but often extends over several months due to the nature of release processes.

Power (2014:6) emphasizes that Lead Time offers a holistic view of the value delivery process and covers gaps that are not addressed by narrower KPIs like Cycle Time. It enables organizations to identify delays across the entire workflow, including areas outside development teams, such as planning, approvals, or deployment phases. By tracking Lead Time, teams can visualize and address bottlenecks that impact the overall flow of value, ensuring alignment with customer satisfaction. The limitation of the Lead Time KPI is that the underlying causes and effects of variability within software teams and organizations are not illustrated.

The KPI First Pass Yield (FPY) helps understand the proportion of units that successfully pass through a process on the first attempt without needing any corrections or rework (Armbruster Santiago 2015:3). It is often used to measure the efficiency of a process by considering the yields of each process step without rework. The calculation involves dividing the number of units entering a process minus the defective units that require rework by the total number of units entering the process. Armbruster Santiago (2015:3) refers to a regulatory compliance contractor in the life sciences industry seeking to implement the FPY KPI to demonstrate the quality of their validation test case development process. By focusing on improving its FPY, the company aimed to distinguish itself from competitors by providing customers clear evidence that it can deliver defect-free test cases, ultimately saving clients time and resources. By Implementing the FPY KPI in combination with implementing a process improvement project using the DMAIC (Define, Measure, Analyze, Improve, Control) methodology, they were able to address issues in the process. These improvements led to a significant increase in their FPY, rising from 66% to 80,5%, which is a 22% improvement in their process efficiency (Armbruster Santiago 2015:9-10). This was a reduction of 6-15 hours per week spent on reworking defective test cases, freeing up valuable time and resources.

2.3 Effectiveness KPIs for performance

Team effectiveness goes beyond efficiency, focusing on the team's ability to achieve project objectives, meet quality standards, and satisfy customer needs.

Defect density is one widely used KPI for measuring effectiveness, and it measures the number of defects within a specific unit of software size, for example, Lines of code, story points, or product backlog items (Huss et al. 2023:314). The calculation simply divides the total number of defects (identified before and after delivery) by the chosen size KPI.

Defect leakage is another similar KPI that measures the proportion of bugs that seep through the testing phase and are identified only after the software is delivered. This KPI is a good indicator of the effectiveness of testing processes. The calculation of post-delivery defects/pre-delivery defects, meaning a higher ratio, indicates a larger proportion of defects escaping the testing phase and being detected later (Huss et al. 2023:318).

Wilkes et al. (2023:63) define a KPI called *Change failure rate* (CFR) as “the percentage of deployments to an organization's software or IT systems that fail or result in unintended consequences”. This definition emphasizes that CFR is not just about outright deployment failures but also situations where deployments have unexpected negative impacts. Wilkes et al. (2023:63) further clarify that CFR can also be viewed as “the percentage of deployments which contain a bug”. Riihimäki (2024:17), on the other hand, explains CFR as a KPI for measuring how often deployed changes fail. To ensure consistent and meaningful measurement of CFR, Riihimäki (2024:17) acknowledges the importance of clarifying the definition of what is meant by “failure”, as different software teams handle deployment failures in different ways. One disadvantage with the CFR KPI is the challenge to match the exact incidents to specific deployments, which makes it difficult to calculate (Rüegger et al. 2024:39). Otherwise, CFR was ranked as the second most important KPI to be automatically measured (Sallin et al. 2021:114). This highlights its significance in assessing the quality and reliability of changes deployed by teams.

Visser and Hulstijn (2023:20) introduce the User stories planned versus delivered KPI, measuring the completed number of story points by the planned number of story points, with a focus on measuring the throughput in software development, applicable to both Agile and DevOps methodologies. In other words, it gives an indication of how much a team delivers compared to what they committed to deliver, but not the quality of the work delivered.

In Table 1, all of the KPIs named in the subchapters 2.2 *Efficiency KPIs for individual performance* and 2.3 *Effectiveness KPIs for individual performance* are summarized.

Table 1: Table of software development teams' KPIs found in the literature.
Source: The author.

Category	KPI	Description	Advantages	Disadvantages	Source
Effectiveness	Defect density	Number of defects in a given software unit (e.g., lines of code, Story Points).	Can be applied to various software development projects, regardless of their size or complexity.	Does not account for severity of defects or the context in which they occur.	Huss et al. (2023:314).
	Defect leakage	Proportion of defects found after software delivery.	Specially effective in evaluating the effectiveness of testing processes.	Focuses solely on defects that escape testing, ignoring other potential quality issues.	Huss et al. (2023:314, 318).
	Change failure rate (CFR)*	The percentage of deployments that contain a bug.	Provides insights into the stability and reliability of code changes.	Difficult to calculate due to the challenge of matching the exact incidents to specific deployments.	(Wilkes et al. 2023:63), (Rüegger et al. (2024:39).
Efficiency	First pass yield (FPY)	Proportion of units passing a process without needing rework.	Highlights process efficiency and areas for improvement.	Provides a narrow view of process efficiency.	Armbruster Santiago (2015:3).
	Velocity deviation	Divides the standard deviation of velocity by the average velocity over a number of previous sprints.	Provides insights into the stability and consistency of the workflow.	Less effective at diagnosing the root causes of instability.	Menezes et al. (2024:9)
	Velocity	Amount of work completed per sprint (measured in story points).	Helps predict future capacity and set realistic goals.	It is subjective and needs to be used in combination with other KPIs.	Budacu and Pocatilu (2018:71), Pham and Neumann (2024:6), Natarajan and Pichai (2024:3), Menezes et al. (2024:9)

Cycle time*	The duration that individual work items spend in specific workflow states.	Provides visibility into workflow bottlenecks and delays.	Requires proper tool support to be effective.	Power (2014:5)
Lead time*	The time it takes for individual work items to move through a system, from the moment of request, idea, or report to the time the item is in the hands of users.	Shows the customer's perspective of how value is being delivered, including features and defect fixes.	Does not help visualize the causes and impacts of variability in software teams and organizations.	Power (2014:6)
Story points	Estimate of a task's complexity, uncertainty, and risk.	Flexible and adaptable to changes, avoids time-based estimations.	Subjective and potentially inconsistent across differ.	Pham and Neumann (2024:6).
User stories planned versus delivered	Measures completed number of story points by the planned number of story points.	Provides a straightforward way to assess how much of the planned work was actually completed	Does not provide a view of the quality of delivered work.	Visser and Hulstijn (2023:20)
Lines of code (LOC)	Measures the amount of lines of code written.	Easy to calculate.	Encourages writing more code, even if unnecessary, and does not reflect code quality or complexity.	Shah et al., (2015:105), Forsgren et al. (2021:18), Riihimäki (2024:12)
Halstead effort by time	Measures the cognitive effort involved in writing code, considering operators and operands.	Provides a more nuanced view of code complexity and developer effort.	More complex to calculate than LOC.	Oliveira et al. (2020:2)
Code owned by time	Number of source code files a developer has contributed the most to.	Identifies the key developers in a team who are essential to a software project's development.	May not accurately reflect effectiveness.	Oliveira et al. (2020:6 & 12)
Commits performed by time	Number of commits made by a developer in a given time.	Useful KPI for projects where developers follow strict and consistent commit policies.	Commit policies can vary across projects.	Oliveira et al. (2020:3)
Committed source lines of code by time	Number of code lines a developer commits in a given time.	Indicates the amount of work done from a developer.	Does not take code complexity into account.	Oliveira et al. (2020:3)

Committed characters by time	Number of characters a developer commits in a given time.	Reduces false positives of effectiveness that arise from simply counting lines changed.	Does not take code complexity into account.	Oliveira et al. (2020:3)
------------------------------	---	---	---	--------------------------

Table 1 is divided into the columns: Category – showing which of the KPIs are for efficiency and effectiveness. KPI - The name of the KPI. Description - A short description of what the KPI measures. Benefit - The benefits of the KPIs are described. Disadvantage - The flaws of the KPIs are described. Source - The source to the KPI was found is presented. Some KPIs are measured only at the team level, as far as the sources state, they have been added to the table due to some relevance to the study. These are marked with the character ‘*.’

2.4 The SPACE framework

There is current thinking in software engineering research on measuring developer efficiency multidimensionally and with self-reported data (Ziegler et al. 2022:22). Out of this interest, the SPACE framework has been developed, that defines five dimensions of developer performance, which are Satisfaction and well-being, Performance, Activity, Communication and Collaboration, and, Efficiency and flow (Forsgren et al. 2021:14). Forsgren et al. (2021:2) argue that efficiency can not be measured by one dimension or one KPI solely. The SPACE framework aims to create a solid ground for teams and organizations to understand better and make better decisions regarding individuals and teams. The recommendation is to use at least three dimensions of the SPACE framework for a balanced and insightful understanding of efficiency (Forsgren et al. 2021:17). If an organization, for example, already measures commits, they should not simply add pull requests to the KPI dashboard, because these are both activity KPIs.

The purpose of each dimension, along with example developer KPIs highlighted by Forsgren et al. (2021:14), are listed below and summarized in Table 2.

Satisfaction and well-being

The dimension *Satisfaction and Well-being* focuses on the developer’s overall happiness, health, and fulfillment in their work environment (Forsgren et al. 2021:6).

Developer satisfaction - This broad measure captures how content developers are with various aspects of their work, including their team, tools, and the overall work culture. Forsgren et al. (2021:6) suggest that high developer satisfaction correlates with high effectiveness and can act as a leading indicator for potential burnout. Using surveys to measure developer satisfaction is therefore recommended, focusing on employee satisfaction, developer efficacy, and burnout levels.

Satisfaction with code reviews assigned - This KPI assesses developers' feelings toward the code reviews they are tasked with (Forsgren et al. 2021:15). Dissatisfaction might arise if a developer feels consistently overburdened with a disproportionate amount of code reviews, impacting their time for other work.

Perception of code reviews - This qualitative KPI measures how developers view code reviews and if they see them as valuable opportunities for learning, mentorship, and shaping the codebase. Or are they perceived as tedious obligations? Forsgren et al. (2021:13) mean that understanding developers' perceptions can provide insights into their overall satisfaction and engagement with the code review process.

Performance

The dimension *Performance* focuses on the outcomes of development processes, the delivered software's quality and impact (Forsgren et al. 2021:8).

Code review velocity - This KPI tracks the speed at which code reviews are completed (Forsgren et al. 2021:15). It can reflect both individual efficiency and team dynamics. While an individual might

complete a review quickly, team policies (like leaving reviews open for 24 hours for broader feedback) can influence overall velocity.

Activity

The dimension *Activity* captures the volume of actions or outputs produced by developers (Forsgren et al. 2021:8). These KPIs offer insights into developer contribution and workload.

Number of code reviews completed - This KPI simply counts the number of code reviews a developer completes within a specific timeframe (Forsgren et al. 2021:15). It provides a quantitative measure of their direct contribution to the final product.

Coding time - This measures the total time developers spend coding or specific times of day dedicated to coding (Forsgren et al. 2021:16). It provides insights into their work patterns and how much time they dedicate to this core activity.

Number of commits - This measures the number of code changes submitted by a developer and reflects on their coding activity (Forsgren et al. 2021:16).

Communication and collaboration

The dimension *Communication and Collaboration* focuses on how developers interact and work together (Forsgren et al. 2021:9).

Code review score - This qualitative KPI assesses the quality of code reviews (Forsgren et al. 2021:14). It provides insights into how effectively developers are collaborating and communicating through the code review process (Forsgren et al. 2021:15).

Pull request merge times - This KPI ties into effective collaboration, as it reflects the efficiency of communication and coordination among team members (Forsgren et al. 2021:10).

Efficiency and flow

The *Efficiency and flow* dimension easures the smoothness of work, intending to minimize delays and interruptions in the work processes (Forsgren et al. 2021:11).

Code review timing - This assesses how code reviews are integrated into the workflow (Forsgren et al. 2021:15). Batching code reviews can minimize interruptions to developers' coding time, improving individual efficiency. However, delays in reviews can create bottlenecks, impacting system-level efficiency.

Effectiveness perception - This KPI relies on developers' self-reported perceptions of their effectiveness, providing a valuable indication of their individual efficiency and flow (Forsgren et al. 2021:17).

Table 2: Dimensions and example of KPIs for individual measuring from the SPACE framework. Source: Modification of Forsgren et al. (2021:14).

Level	Satisfaction & Well-being	Performance	Activity	Communication & Collaboration	Efficiency & Flow
	How fulfilled, happy, and healthy one is	An outcome of a process	The count of actions or outputs	How people talk and work together	Doing work with minimal delays or interruptions
Individual	<ul style="list-style-type: none"> Developer satisfaction Satisfaction with code reviews assigned Perception of code reviews 	<ul style="list-style-type: none"> Code review velocity 	<ul style="list-style-type: none"> Number of code reviews completed Coding time Number of commits 	<ul style="list-style-type: none"> Code review score Pull request merge times 	<ul style="list-style-type: none"> Code review timing Effectiveness perception

In Table 2, a modification of a table of the dimensions and example KPIs for individual measuring from the SPACE framework, created by Forsgren et al. (2021:14), is presented. Table 2 summarizes the content of subchapter 2.4 *SPACE framework*. The modification of the original table consisted of excluding KPIs that were not measuring on an individual level and excluding KPIs that were not described in the paper.

2.5 Measuring individual performance

The appropriateness of individual measurement may depend on specific context of an organization and project (De Ruyck et al. 2020:23). This raises the question of whether individual performance should be measured and rewarded. While individual pay-for-performance can drive speed and effort, equal team-based rewards may enhance collaboration and accuracy.

In agile organizations, where task interdependence is high, cooperation and communication are essential (De Ruyck et al. 2020:23). However, measuring individual performance in agile may conflict with the collaborative nature of agile principles (Beh et al. 2022:84, Al-Heyasi 2018:3). Agile emphasizes teamwork and shared responsibility, focusing on individual KPIs might therefore undermine team collaboration and create competition and divisipn amongs team members, rather than collaboration (Al-Heyasi 2018:3, Pinter et al. 2017:313).

Research offers mixed findings: Pearsall et al. (2010:188) suggest hybrid rewards, combining individual and team rewards, are most effective in interdependent teams by encouraging knowledge sharing while reducing free riding. However, Barnes et al. (2011:1627) argue that mixed incentives create a social dilemma, where individuals prioritize their own performance at the expense of team outcomes, leading to faster but less accurate work.

Cultural aspects, such as hierarchical processes and team opinions about peers, further influence the effects of individual KPIs (Mendes et al. 2018:187). In Nordic countries, teams tend to emphasize collective success, whereas in India, hierarchical progression and individual prestige play a more prominent role. These cultural variations influence how teams perceive themselves and their capabilities, making it essential to measure their impact on agile projects.

Furthermore, Almeida and Carneiro (2023:11) suggest that the role of an individual in Scrum is not a factor in the perception of the importance of KPIs, while years of experience is. This finding indicates that while organizational culture and team dynamics play a role in shaping attitudes toward performance measurement, individual factors such as experience level may also affect how KPIs are perceived and utilized.

2.6 Selecting and adopting KPIs

With many KPIs being discussed in studies, the process of adoption, selection of the right measurements, and contextual factors are referred to less (Boon et al. 2023:133). After conducting a systematic literature review, Boon et al. (2023:133) found that there is a need for selecting, prioritizing, and balancing the right metrics in measuring progress toward business goals and improving performance, but empirical studies are scarce.

Additionally, López et al. (2021:20) found that a lack of standardized models for KPIs and terminology in agile software development is a recurring issue.

Boon et al. (2023:130) researched the factors of actionability of KPIs, meaning the ability of a KPI to proactively drive awareness and action. Boon et al. (2023:130) emphasize that a crucial part of successfully leveraging value is understanding how a KPI dashboard contributes to the improvement and what the challenges are in adopting these dashboards with the rightful KPIs.

The goal of the study was to get a better understanding of how to operationalize a KPI and what impact KPIs have on improvement ambitions in agile teams (Boon et al. 2023:130). What Boon et al. (2023:142) found was that KPIs are most effective when they are actionable and stimulate dialogue and qualitative measures such as Employee and Customer satisfaction should be used in an interactive and shared format.

Furthermore, introducing KPIs requires careful guidance and integration into daily rituals (Boon et al. 2023:142). Teams should get continuous technical- and management support, and the dashboard should be customized iteratively, with a focus on “just enough” actionable KPIs to avoid overwhelming teams. Lastly, a higher quality of data is fundamental for using KPIs effectively, and this means efficient

data collection processes. Boon et al. (2023:142) emphasize using KPIs like Workflow health to improve these processes.

When adopting KPIs, they should always be linked to the team's specific goals (Budacu & Pocatilu 2018:71, Natarajan & Pichai 2024:2). Multiple KPIs should be used across different dimensions to create a balanced and comprehensive view of performance (Forsgren et al. 2021:2). Forsgren et al. 2021:19) mean that too many metrics can cause confusion, lower motivation, and make improvement goals feel unattainable.

Shah et al. (2015:105) suggest that development teams should strive for a balance between what should be measured and how much value the KPI adds. To improve the effectiveness of software KPIs, Budacu and Pocatilu (2018:71) further propose three key recommendations: The first is to use short measurement intervals, the second is to measure trends instead of exact numbers, and the third is to stop using a KPI when it doesn't drive any change.

Boon et al. (2023:132) also emphasize that a stakeholder-driven approach should be taken, which aims to address the needs and goals of both internal and external stakeholders, such as development teams, customers, suppliers, and users. Although not all groups are equally relevant in every context, balancing their perspectives creates a solid foundation for designing KPIs.

Boon et al. means that data quality and the efficiency of data collection are prerequisites for the use of KPIs. Low data quality can hinder the use of measurements due to a lack of trust in the numbers (Boon et al. 2023:143). Measurement data quality is a key factor influencing actionability (Boon et al. 2023:130). Data quality affects actionability in that low data quality impedes the use of measurements by a lack of trusting the numbers, and transparency improves adherence to processes and data maintenance and therefore stimulates action taking (Boon et al. 2023:143).

2.7 Benefits and drawbacks of dashboard implementation

Through their study, Boon et al. (2023:140) found many benefits perceived by scrum masters and teams. Introducing a dashboard offered transparency by teams being motivated to reflect on the results, improve the quality of their backlogs, and improve as a team. Scrum masters had also remarked that some of the information that was in the dashboards could not be found as easily otherwise, which led to the team being more aware of potential problems they needed to act on.

Additionally, scrum masters perceived that they found support in the dashboards when having a dialogue with team members regarding their work, as well as getting support for decisions or requests to the team (Boon et al. 2023:141). Similarly, Oliveira et al. (2020:23) argue that effectiveness KPIs can complement the subjective perceptions of team leaders.

Despite the benefits, implementing a KPIs system can present challenges (Oliveira et al. 2020:2). The disadvantages of this type of system are that it requires extra work, as new components need to be integrated (Budacu & Pocatilu 2018:77). It also needs to be adapted to the main project and process management tool, and new versions of project and process management tools in such cases require the need to change import modules. However, these drawbacks are avoided through a careful and well-thought-out design process, which can make the system more stable and flexible in future updates.

The biggest challenge for Boon et al. (2023:141) was the data quality of the KPI dashboard. For example, the Cycle Time KPI became useless when items were not registered timely, causing the KPI to be perceived as useless or not actionable. Another challenge was creating a default dashboard because each team had specific needs, demands, and preferences, which made the team use the dashboard in different ways.

Another challenge is that developers usually tend to not see any benefits for using KPIs (Teiber 2021:55), which makes it challenging with acceptance of implementing KPIs. KPIs can help improve workflows, increase transparency, and ultimately improve the efficiency and effectiveness of agile teams (Psarov & Druzhinin 2024:93). Teams with higher individual levels of contribution and impact have better final product outcomes (Gamble & Hale 2013:7).

Psarov and Druzhinin (2024:97) emphasize that stakeholders can refer to diagrams and dashboards to understand the status of work items, facilitating better communication and shared focus on project goals. If performance indicators are misused, such as by comparing teams and employees or focusing too much on just numbers, metrics may become meaningless as teams may start inflating them

intuitively (Ertaban et al. 2018:1). Furthermore, differences between the team's goals and top management's targets can be a problem (Ertaban et al. 2018:2).

Oliveira et al. (2020:2) conducted a study where a quantitative analysis was made to determine the correlation between leaders' rankings versus KPI-based rankings of the developers of their teams by efficiency as well as the team leaders' impressions about the chosen efficiency KPIs. The results of the study showed a greater correlation of leaders' perceptions with code-based KPIs than commit-based KPIs. The results also showed that team leaders had more positive impressions of the code-based KPIs and would potentially adopt them, compared to the commit practices. This is due to the great variation of commit practices when it comes to the developer and the project. The results of the study conducted by Oliveira et al. (2020:21) showed that team leads believed that using KPIs would provide fairness in team evaluations, help recognize individual contributions, and improve task allocation by aligning developers with work where they are most productive.

2.8 Design thinking

Design thinking is defined as an analytic and creative process wherein a person experiments, creates and prototypes models, gathers feedback, and redesigns (Razzouk & Shute 2012:330). Razzouk and Shute (2012:330) state that needs, dissatisfaction, and determination to act are the start of a design process.

Design thinking has been considered a learning cycle linking individuals with different learning styles to different phases and as a way to address cognitive bias (Carlgren & BenMahmoud-Jouini 2022:46). Descriptions of design thinking as a process often include three phases: inspiration/need-finding, ideation, and implementation/prototyping.

Furthermore, design thinking consists of five themes: user focus, problem framing, visualization, experimentation, and diversity (Carlgren & BenMahmoud-Jouini 2022:46). Design thinking is about how designers think and work, and has emerged as a practice since the early 2000s. At its core, design thinking relates to how designers perceive and think (Razzouk & Shute 2012:334). It is an iterative and interactive process where the designer looks at a problem and explores different ideas to solve it and to find the best solution.

The design process is flexible where the designer keeps changing and improving their ideas or requirements as they learn new things (Razzouk & Shute 2012:335). This helps fix mistakes and make sure the solution fits the problem well. Razzouk and Shute (2012:336) emphasize that essential characteristics of a design thinker include being considerate of human needs and environmental interests, having the ability to visualize ideas visually, having the ability to communicate across disciplines and work with others, and lastly, avoiding the necessity of choice and being able to combine the best possible choices.

3 Method

3.1 Approach and chosen methods

The process began with me taking contact with Team lead A, to explore problems that need solutions at the case firm Extenda retail. Together with Team lead A, we agreed during two probing dialogues (see Appendices 1 & 2) on a problem that I was going to help solve with this study.

The possibility of spending time at the office allowed for personal dialogues to follow up on work and to gather additional information about the software development process of the specific team from the case firm Extenda Retail, for whom I built the KPI dashboard. Team Lead A was a readily available contact person throughout the study, who provided me guidance on their team's software development process. This study is a case study, conducted with a mixed-method approach.

An entrepreneurial approach combined with an Action research approach was applied due to a verbalized wish from Team Lead A for improvement in the software development process by adopting performance KPIs (see Appendices 1 & 2). The entrepreneurial approach is relevant because of its focus on understanding the situation by pinpointing perceived problems through interview guides based on the SPIN method (Dynehäll & Ståhlberg 2014:36).

During the process of prototyping the KPI dashboard, an action research (AR) approach was applied, aiming to create change through direct involvement between the researcher and participants (Robson 2014:24). An AR approach is also especially suitable for the process of developing the KPI dashboard because of my direct involvement with the case firm participants during the prototyping process. The AR approach includes the steps diagnosing, action planning, action taking, and evaluating, with the entrepreneurial approach integrated in form of SPIN interviews and MVP tests. Through SPIN interviews, the primary empirical data will be gathered.

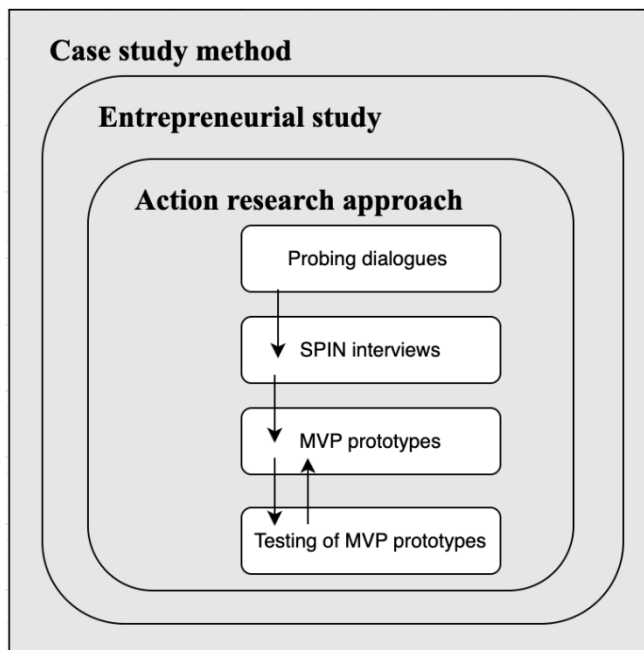


Figure 1: Visualization of overall method.
Source: Modification of Staake (2022:20).

3.2 Selection of interview participants

The interview participants were selected together with a contact person from Extenda Retail. To get a comprehensive, multidimensional perspective on the problem area, individuals with diverse roles were selected to participate in the SPIN-interviews. For the three MVP tests, two participants were replaced by new individuals who had not participated in the SPIN interviews. The test participants for the MVP tests were chosen based on availability since the test period fell under the period of holidays, and many were on vacation. See Tables 3-5.

Table 3: Probing dialogues/other dialogues

Role	Interview method	Time	Date	Appendix
Team lead A	Personal dialogue	20 min	2024-09-04	1
Team lead A	Personal dialogue	20 min	2024-10-02	2
Team lead A	Brainstorming	20 min	2024-10-02	3

Table 4: SPIN-interviews

Role	Interview method	Time	Date	Appendix
Engineering manager	Google meet	25 min, 29min	2024-10-18, 2024-11-22	7
Software developer	Google meet	41 min	2024-10-21	8
Software engineer	Google meet	19 min	2024-10-21	9
Director of human resources	Google meet	36 min	2024-10-22	10
Team lead A	Google meet	36 min	2024-10-22	11

Table 5: MVP-tests and Jira dashboard set-ups

Jira dashboard set-ups				
Role	Interview method	Time	Date	Appendix
Senior frontend developer	Google meet	15 min	2024-12-18	No protocol
Software engineer	Google meet	15 min	2024-10-19	No protocol
MVP 1				
Role	Interview method	Time	Date	Appendix
Senior software Developer	Google meet	23 min	2024-12-27	13
Team lead B	Google meet	9 min	2024-12-27	14
Engineering manager	Google meet	20 min	2024-12-27	15
Software engineer	Google meet	10 min	2024-12-30	16
Team lead A	On site	20 min	2024-12-30	17
MVP 2				
Role	Interview method	Time	Date	Appendix
Team lead B	Google meet	15 min	2025-01-03	18
Software engineer	Google meet	2 min	2025-01-03	19
Senior frontend developer	Google meet	10 min	2025-01-03	20
Engineering manager	Google meet	10 min	2025-01-06	21
Team lead A	Google meet	15 min	2025-01-08	22
MVP 3				
Role	Interview method	Time	Date	Appendix
Senior frontend developer	Google meet	5 min	2025-01-10	23
Software engineer	Google meet	2 min	2025-01-10	24
Team lead B	Google meet	11 min	2025-01-10	25
Engineering manager	Google meet	13 min	2025-01-10	26
Team lead A	On site	5 min	2025-01-10	27

3.3 The case study method

The study was conducted through a case study in collaboration with the case firm Extenda Retail, and a specific team, that I created the KPI dashboard for. This enabled me to go in-depth and capture complexities, relationships, and processes, as Robson (2014:25) describes. Further, Robson (2014:25 & 26) means that it is strongly advisable to use multiple methods of collecting data as well as to have multiple data sources when using this method. Owing to that, I have analyzed data related to the KPIs, relationships between data sources, and the software development process by accessing all existing data in the case of the company's Jira system, besides several spontaneous dialogues in the office and SPIN interviews.

Through the multiple methods of collecting the data for this study, *triangulation* can be achieved and can be used for two purposes in a case study (Pickard 2017:102). One purpose is to gather data from multiple sources to confirm the same facts or phenomenon, and the other purpose is that triangulation helps balance biases and gather various perspectives. In this study, a triangulation was achieved that fulfilled both these purposes. The first one is achieved through gathering data from multiple methods, and the other purpose is achieved by including a diverse set of roles for the SPIN interviews to gain a more complete and nuanced understanding of the subject.

Some benefits of the case study method are that the scope of the study, including timeframe and contextual coverage, is adjustable based on the time and resources available Robson (2014:26). It is also more closely tied to real-life contexts than methods like experiments and surveys, as well as being suitable for widely different types of studies. This suits especially well with my sub-purpose of creating a KPI dashboard since it involves an iterative and creative entrepreneurial approach, where concrete deadlines can negatively impact this process.

3.4 Entrepreneurial approach

During the exploratory talks, a wish for an improvement in the software development process was verbalized. Therefore, I chose to conduct the study with an entrepreneurial approach. An entrepreneurial approach consists of *design thinking*, the *LOOP method* and the *SPIN method*.

When it comes to the design thinking approach, the user and the users needs are a big part of the development process (Rainer & Prince 2021:411). It involves five stages: empathize, define, ideate, prototype, and test. The empathize stage focuses on understanding the user's perspective, while define shapes the problem based on this understanding. In ideate, creative solutions are generated, ensuring they align with the insights gained earlier. Prototyping and testing follow, refining the product through user feedback in multiple iterations.

The LOOP method emphasizes the importance of early customer feedback to avoid investing time and resources into a product that the market doesn't want (Dynehäll & Ståhlberg 2015:19).

The SPIN interview guides were formed with the SPIN method (see Appendices 4-6), often used in interviews with potential customers (Dynehäll & Ståhlberg 2014:35), which suits the chosen entrepreneurial approach well. The components of the SPIN method are situation questions, problem questions, implication questions, and need questions. The aim of the situation questions is to get an understanding of the interviewee's current situation or circumstances, and they are neutral questions. The problem questions go from understanding the situation to identifying perceived problems (Dynehäll & Ståhlberg 2014:36). The implication questions delve into the consequences of the identified problems to highlight the importance of addressing them. Lastly, the need questions focus on potential solutions.

3.5 Action research approach

When prototyping the KPI dashboard, I applied an action research (AR) approach, an approach that aims to change something together with a direct involvement between the researcher and the participants in the process (Robson 2014:24). The action research method is highly collaborative and is especially appropriate for practitioner-researchers, as it fosters both professional growth and personal development.

The researcher does not simply observe, they actively participate, and this presence directly influences what happens (Robson 2014:25). This involvement is seen as a valuable part of the research process rather than a flaw or limitation in the method. The direct engagement of the researcher means

tackling real-world problems and iterative problem-solving and when successful, it can make a real positive transformation within an organization.

The AR approach is a cyclical process, and Pickard (2017:159-162) describes the AR cycle with five steps: Identifying problems, action planning, implementation, evaluation, and reflection. Boon et al. (2023:133), on the other hand, describe AR with four steps: diagnosing, action planning, action taking, and evaluating. I have chosen to take inspiration from Boon et al. (2023:133) for my AR, adopting their process model with a few changes to fit my study (see Figure 2).

The first step of the AR approach, diagnosing (identifying problems), includes identifying a problem or an issue, especially when there is a desire to improve a practice (Pickard 2017:159). Pickard (2017:159) emphasizes that it also is of relevance to review relevant literature and documentation, as well as gathering knowledge from stakeholders is also important in the first step, often through interviews.

The second step of the AR approach is action planning, includes the design of an intervention which is based on the data from the first stage, and there is no predetermined length or nature of the intervention (Pickard 2017:160).

The third step of the AR approach is action taking (implementation), which includes the practical part of implementing the intervention within the research context and delivering a solution (Pickard 2017:160).

The fourth step of the AR approach is evaluating, which includes examining the intervention, which Pickard (2017:161) state can be done in various ways, as for example with questionnaires to participants in order to gather feedback.

The last step of the AR approach, which is not included in this study, is reflection, which Pickard (2017:161-162) explains as evaluating what happened and why it happened. Pickard (2017:162) means that usually no more actions follow this stage.

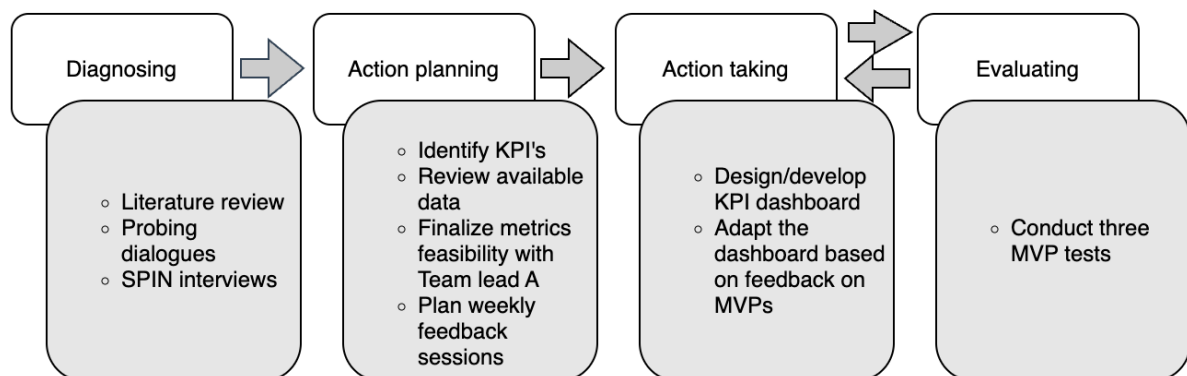


Figure 2: Phases and activities of Action research approach.

Source: Modification of Boon et al. (2023:133).

3.5.1 Diagnosing

The first step is called diagnosing, which included identifying existing KPIs for agile teams. For this, a literature study was conducted to identify and describe common agile KPIs in use on an individual level. To examine the current situation of the case team from the case firm, probing dialogues and SPIN interviews were conducted with a Team lead, an Engineering manager, a Software engineer, a Software developer and a Director of human resources (HR), who are further on referred to as stakeholders in this study.

3.5.2 Action planning

The purpose of the action planning step was to further identify relevant KPIs through literature study, to ensure that the KPIs that are selected for the KPI dashboard development are highly relevant, aligning with the theory and knowledge in literature as well as the case firms' needs.

3.5.2.1 Identifying KPIs

This stage involved reviewing literature on agile KPIs, and the SPACE framework, and aligning the goals with the organization's expectations, which was done through regular check-ins with Team lead A at the office (without protocol), a brainstorm session, and SPIN interviews. Additionally, identifying what data was available within Jira and EazyBI to support these KPIs was an important step in this phase, as well as analyzing how the data could be used.

Ethical considerations were addressed together with HR by discussing how individual-level data would be presented and ensuring measuring on individual level was non-invasive and focused on a higher degree of effectiveness insights rather than personal surveillance.

Based on findings from the mentioned data collection methods, I identified a set of candidate KPIs for the dashboard. Although, challenges arose and the process of identifying candidate KPIs for the dashboard became an iterative process.

3.5.2.2 Dimensions

The dimensions for the dashboard have been based on the SPACE framework, proposed by Forsgren et al (2021:14). The dimensions I used were Performance (P) and Efficiency, and flow (E). The dimension Satisfaction and well-being (S) have been left out of this study since it requires collecting and measuring qualitative data. This study focuses merely on leveraging automatically collected quantitative data and visualization in the form of a dashboard. It is worth noting that during the SPIN interviews, a gathering of qualitative data on satisfaction and well-being was described, which is done twice a year. Considering this, the dimension 'S' is being covered to some degree already.

3.5.3 Action taking

The action taking step involved executing the action plan, transforming the planned design into a functional KPI dashboard for measuring individual performance in agile teams. In this section, a detailed explanation of the process of prototyping the KPI dashboard is explained.

3.5.3.1 Tools and technology

The dashboard is created in the project-management tool called Jira, used for planning and tracking work across teams (Atlassian 2025a). Jira has a function specifically for creating dashboards, which is done by adding several gadgets, that, among other things, contains charts (Atlassian 2025b). A Jira gadget is a customizable module that displays information about projects and tasks and is used to visualize data with real-time updates. Jira has built-in gadgets for their dashboards, which I found limiting due to their limited customization options and inability to support advanced calculations or detailed data segmentation. Therefore, I have used Jira's native EazyBI plugin, a business intelligence tool integrated within Jira to visualize data with customizable reports (EazyBI 2025a).

Reports in EazyBI are created using drag-and-drop functionality, allowing users to filter and segment data based on dimensions, for example time, projects, users, and issue types (EazyBI 2025b). The dimensions can be dragged into three sections: columns, pages, and rows. Dragging dimensions into the rows and columns defines the layout of the report, whilst dragging dimensions to the pages section will provide report filters. For example, adding the Project dimension to the pages section will allow the user to select a particular project in which the report should display data (EazyBI 2025b).

In addition, EazyBI provides interactive data visualization and analysis with the opportunity to drill through other dimensions of the report. For example, Figure 5 shows a report of the percentage of bugs in production of the total Jira issues in production. When clicking on one of the values representing a period, in this case, one month, the user can further click on "Drill through" which will automatically present a table of the exact Jira issues that have been visualized in the report for that given period (see Figure 5.1 & 5.2). A Jira issue represents a work item or project task, in any form (e.g. epic or bug), that consists of detailed information about the given task (Atlassian 2025c). For example description of task, who is assigned to the task, when it is due, what the current status is, and more.

For advanced analysis, EazyBI supports custom-calculated measures that can be created using the query language Multidimensional Expressions (MDX).

3.5.3.2 Dashboard designing and prototyping

The data visualized in the dashboard is real-time data collected directly from Jira through its built-in reporting tools. Building the dashboard involved several steps:

- First, I identified the KPIs that needed to be tracked. The KPIs were selected based on findings in the literature study, SPIN interviews, and the data available to visualize, fitting within the dimensions of the SPACE framework.
- The next step included defining which dimensions and measures to include in the reports. (For example, *Project*, *Assignee*, and *Sprint*. These dimensions serve as the foundational categories for organizing the data, whereas KPIs, such as *Velocity*, are calculated values derived from these dimensions. For the *Velocity* KPI, it would be necessary to select the *Sprint Story Points Completed* measure).
- The third step was to design the visualizations.

The drag-and-drop functionality allowed me to experiment with different dimensions and KPIs that were relevant to measuring individual performance in agile teams. For example, the dimension *Assignee* helps break down performance per individual, while *Sprint* allows for trend analysis across different project iterations. I could also try out different graphical visualizations, such as bar charts, pie charts, line diagrams, and pivot tables, to determine which formats suit the different KPIs best. For example, bar charts were mostly useful for comparing individual performance across the same KPI, whilst line charts were effective for visualizing trends over time.

I evaluated and refined alternative approaches in the development process iteratively as various challenges emerged. Some of the initially planned KPIs could not be developed despite multiple attempts, often due to limitations in data availability. In several cases, the required data was either not configured within EazyBI or was not automatically tracked in Jira, which limited the range of KPIs that could be measured.

After creating the reports in EazyBI, I could add them as gadgets to the Jira KPI, as EazyBI is a plugin, and its reports appear as options within Jiras gadgets. The end result of the KPI dashboard is presented in Figures 3 & 4. Two views of the dashboard were created, one for the developers and one for the team leads/managers. The second view is different because it has additional reports that show the team leads or managers the KPIs for all individual team members.

The KPI dashboard (both views) ended up consisting of two KPIs on team level, these being Change failure rate (see Figure 5) and Cycle time by status (see Figure 6).

With the wish to measure effectiveness at an individual level, I initially planned to develop the Change failure rate KPI report on individual level in EazyBI, as it was a KPI requested by Tam lead A (see appendices 2 & 3) and well supported in literature. However, when presenting my idea to Team Lead A during a spontaneous dialogue in the office, I learned that bugs are not linked to the individual who caused them. Instead, all bugs are collected in a shared backlog and assigned randomly, meaning the person fixing the bug is not necessarily the one who introduced it. As a result, it was not possible to measure the Change failure rate KPI or any other bug-related KPI at the individual level.

The only KPIs that were identified in literature for measuring the effectiveness of individuals were those that measure defects or bugs when it comes to automatically collected data. I evaluated it to be important to include an effectiveness measure for measuring performance because I followed the SPACE framework as a foundation for a multidimensional performance measurement. Even though the KPI report could not be developed on an individual level, an effectiveness KPI report, the Change failure rate was developed for the KPI dashboard on a team level (see Figure 5).

As for the Cycle time by status KPI (see Figure 6), the KPI is usually measured on a team level, according to the literature study conducted in this study. However, I also decided to test the KPI at an individual level to explore whether it could provide additional insights, particularly for self-evaluation. While this approach was not directly supported by literature, I wanted to experiment with different perspectives on Cycle time KPI measurement. Therefore, I created two variants of the view: one showing Cycle time KPI at the team level and another displaying it for individual developers.

As the project progressed, challenges related to data availability made it difficult to develop several individual-level KPIs as initially intended. However, the Cycle time KPI emerged as a viable option since the necessary data was accessible. Given that the Cycle time KPI is commonly measured at the team level, developing it in this form first allowed for a structured foundation from which I could

develop the KPI on an individual level. Since the team-level KPI was already being developed, including it in the KPI dashboard provided a broader context for comparison, ensuring that individual performance could be analysed in relation to overall team trends.

3.5.3.3 Challenges

The main challenge in creating a dashboard with agile KPIs was data availability or data quality. This made it harder impossible to develop a sufficient number of KPIs that could have been good indicators of the performance of an individual in an agile team. For example, the Change failure rate KPI becomes useless in the case of measuring it at an individual level when the bugs in Jira are not connected to the team member that caused them.

At one point, in cooperation with the admins in the case firm, an attempt was made to configure data on pull requests to EazyBI (data of submitted code that wants to be merged to the development environment). The aim was to get historical data of code commits made by team members, to measure the Commits performed by time KPI that would have represented the Action dimension in the SPACE framework. Unfortunately, errors occurred and the correct data was not configured, which led to the KPI dashboard lacking the representation of a dimension.

3.5.4 Evaluating

In the evaluating phase, three feedback sessions were made in the form of MVP tests, followed by iterative corrections to the dashboard based on the feedback received. This iterative process lasted three weeks in total and included three MVP tests conducted by letting the MVP test persons think aloud.

A Minimum Viable Product (MVP) is the simplest form of a product and a quick way for an entrepreneur to test ideas (Ries 2011:93). Unlike traditional product development, which often involves long planning phases to create a perfect product, the MVP focuses on learning early in the process. It aims to test key business assumptions, not just design or technical details (Ries 2011:94). Ries (2011:95) means that MVPs vary in complexity, and there are no certain guidelines for how simplified or complex an MVP should be done.

As insights and limitations were discovered from the MVP test persons during the evaluation, adjustments were made accordingly to the dashboard. This feedback loop created a repeating cycle where insights from the evaluation phase directly guided the action-taking phase, leading to continuous improvements, which is illustrated in Figure 2, with the arrow going from evaluating back to action taking.

3.6 Theory and earlier studies

To gather relevant studies, I primarily used Google Scholar and the Karlstad University Library as search tools. In addition to academic articles and conference papers, I also consulted books, mainly in digital format.

The selection of sources was conducted with a critical approach, focusing on identifying research that aligns with the study's two purposes. Research articles were evaluated based on their key concepts and content to ensure that they provided a solid theoretical foundation. To achieve a well-rounded perspective, both recent and older sources were incorporated, allowing for a broader understanding of the subject. Searches were conducted using keywords related to agile performance measurement, KPI frameworks, and individual vs. team-level metrics.

Although every effort was made to access high-quality research, some limitations arose, mainly due to a lack of research on the topic of measuring at an individual level in agile teams, but also due to some research articles having restricted access. Following keywords have been used to search for and find relevant information to the *Theory and earlier studies* chapter:

Keywords: Measure efficiency agile team, Measure effectiveness agile team, Measure efficiency scrum team, Measure effectiveness scrum team, Measure performance, Measure individuals in agile teams, Measure individuals in scrum teams, KPIs for agile teams, KPIs for individuals in agile team, developer effectiveness, List of KPIs for measuring agile teams

3.7 Degree of validity and reliability of the study

According to Robson (2014:54), the data collection method is more reliable if the same measurement is repeated under the same conditions and provides the same data. Additionally, Robson (2014:56) emphasizes that using triangulation helps make the data collection method more reliable. This aligns with how multiple data collection methods have been used in this study, achieving triangulation, which is also emphasized by Pickard (2017:102). Firstly, data from different sources confirmed similar facts, strengthening the study's validity. Secondly, biases were reduced by incorporating diverse viewpoints from different roles of both the SPIN interviewees, and MVP test persons, reinforcing both validity and reliability.

A key element in enhancing the study's degree of reliability and validity was the iterative MVP testing process. The MVP prototypes were repeatedly tested and refined based on feedback from stakeholders, ensuring that the collected insights to a higher degree were both accurate and reflective of real-world conditions. The feedback loops included stakeholders iteratively reviewing and validating the proposed solutions, iteratively.

Additionally, to enhance the degree of accuracy and credibility of the collected data, interviews were recorded, transcribed, and approved by the participants. This validation process helped minimize misinterpretations and ensured that the empirical data material accurately reflected the participants' perspectives.

3.8 Degree of empirical saturation

According to Mason (2010:1), the principle of saturation should guide sample sizes in qualitative research. When new data no longer contributes additional insights into the topic under investigation, saturation is reached. However, there are several factors that influence how quickly or slowly saturation is achieved in a qualitative study, such as studies with modest claims may reach saturation faster than those aiming to describe processes across disciplines (Mason 2010:2). Additionally, a more diverse population may require a larger sample size to achieve saturation.

Since this study is centered on one specific workflow, the software development process of one team from the case firm Extenda Retail, saturation was reached relatively quickly. Additionally, the diversity of the SPIN interviewees and MVP test persons, spanning different roles, ensured a comprehensive perspective. The findings from the selected SPIN interviewees and MVP test persons were consistent rather than introducing entirely new insights. The MVP 3 test did not uncover significant new feedback, which suggests that key insights had already been captured.

Mason (2010:12) further emphasizes that both the researcher and their supervisor should understand the concept of empirical saturation early in the research process to establish minimum requirements. The decision regarding the number of participants was primarily affected by availability of participants, for both the SPIN interviews and the MVP tests, five persons participated, which highly promotes the degree of empirical saturation that has been reached within this study.

Mason (2010:4) points out that in studies with limited time, trying to reach empirical saturation can be risky because stopping data collection might miss out on valuable information. In this study, the timeframe and the case firm will mainly determine what is considered empirical saturation.

3.9 Ethical considerations and GDPR

The Swedish Authority for Privacy Protection (2021) states that the General Data Protection Regulation (GDPR) is designed to protect individuals' rights and freedoms. The Swedish Authority for Privacy Protection (2021) specify that personal data should be collected only for legitimate purposes, kept to a minimum, accurate and updated, not stored longer than necessary, protected from unauthorized access, and that there must be evidence of compliance with these principles.

To honor the principles of GDPR, in this study, all participants were informed verbally how their data will be managed and used, with the information sheet developed by Karlstad University, used as a framework. The consent was recorded in a separate audio file before the start of the SPIN interview or MVP test. To ensure anonymity, personal names were removed, though participants were referred to by their roles, which are relevant to the study's data collection. Throughout the research process, all data was stored on my personal computer and deleted once the thesis was completed. Only the data necessary

for the study's purpose was collected, and it has been updated to reflect accurate collection times and dates.

4 Results

In this chapter, the most prominent results of the SPIN interviews (see Tables 6-8) and three MVP tests (see Tables 9-11) are presented. Additionally, the prototype of the KPI dashboard is explained in detail and presented in Figures 2-12.

4.1 Results from SPIN interviews

In the following tables (see Tables 6-8), a selection of key findings from the SPIN interviews are presented, with both questions and answers. One of the SPIN interviews was conducted in Swedish, therefore, note that the answers from that specific SPIN interviewee have been translated to English in Table 8. For full SPIN interview transcripts, see Appendices 7-11.

Table 6 - Results from SPIN interviews with Team lead A and Engineering manager.

Source: Appendix 7 & 11.

Interview question	Team lead A	Engineering manager
Situational questions		
1.1 How are you currently working with measuring performance in terms of efficiency within development teams?	Measuring efficiency relies on observation, there are no methods for this.	Measuring efficiency relies on observation and feedback. It is feedback from the person itself but also, in the end, feedback from team-members. So you can say it is a reactive observation.
1.2 How are you currently working with measuring performance in terms of effectiveness within development teams?	We don't have any specific measurement for effectiveness. It's more of if we are missing deadlines, then we react on that.	Effectiveness is observed by comparing requirements and acceptance criteria in the end of a product delivery or an upgrade on a product. It's seen in the final product, by looking at whether the team delivered what they set out to do. I think it is even more important that the team is looking at itself, you know, because you are looking for self-driving teams.
1.3 How are you currently measuring the performance of separate individuals in teams?	It's also the same answer there. We don't have any specific measurements for individual team members.	Individual performance is observed informally without defined KPIs.
1.4 How often are follow-ups made regarding the teams' performance?	Performance is indirectly followed up through story points and informal observations, but there's no structured or systematic follow-up.	No formal practices, but there are HR assessments that focus on individuals. These are used to initiate discussions within the team to align expectations.
1.4.1 What methods do you use to follow up the work of a team?	Story points and capacity are used as rough indicators, but they are not direct measures of performance.	None, at least not formal practices.
1.4.2 What methods do you use to follow up the work of separate individuals in teams?	Observational methods are used, but no formal system is in place.	It is the HR assessments.

<p>1.5 How well do the current practices capture the performance of a team, in your opinion?</p>	<p>Current practices only partially capture team performance, as KPIs like story point velocity are always available but do not reveal efficiency or effectiveness. These KPIs indicate capacity trends, whether stable, increasing, or decreasing, but do not provide meaningful conclusions about performance quality. As a result, decisions are often based on quantity rather than quality.</p>	<p>Current practices do not effectively capture team performance, as there are no formal measurement processes. While Scrum and retrospectives aim to optimize workflows, they often lack depth. PI planning in SAFe provides some structure for cross-team coordination, but without proper KPIs, assessing and improving collaboration remains a challenge.</p>
<p>1.6 How well do the current practices capture the performance of separate individuals in teams, in your opinion?</p>	<p>The same applies to individuals. For individuals, other signals, such as observations from managers or team members, play a role.</p>	<p>The effectiveness of capturing individual performance depends on how focused the organization is on its processes and how individuals approach their roles. An annual survey in our organization revealed a lack of clarity around expectations and performance measurement, which impacts individual performance evaluation. Cultural differences also affect how individual performance is captured. In the Netherlands, people generally place less emphasis on formal expectations, while in other cultures, clear expectations and performance measurements are considered very important. This variation further influences how well individual performance is captured across teams.</p>
<p>Problem questions</p>		
<p>2.1 What problems do you experience in getting an overall picture of how productive the teams in your department are?</p>	<p>The main issue is the lack of a systematic approach to measuring effectiveness, making it difficult to draw objective conclusions or set measurable goals. This results in challenges when trying to identify areas for improvement or track progress constructively and productively.</p>	<p>The main problem is that the lack of clear and well-prepared work at the start of a sprint disrupts the flow for developers, leading to frustration and inefficiency. The reality often deviates from the ideal where developers can work smoothly and deliver on time. The uncertainty around requirements and constant changes increase stress, making it difficult for developers to maintain a productive flow.</p>
<p>2.1.1 What specific data are you missing today?</p>	<p>The main problem is the lack of a systematic approach to measuring effectiveness, which makes it difficult to set and track tangible goals. Without proper measurements, it's challenging to identify areas for improvement or track progress constructively. This often leads to decisions being based on intuition rather than data, and achieving goals becomes difficult due to the absence of active measurements.</p>	<p>The missing data can be categorized into a few key areas: clarity of goals and the impact of scope changes, the quality of preparation before work starts (including clarity in acceptance criteria and requirements), and the number of defects delivered to external teams. These KPIs are crucial for understanding why outputs fail or require rework. By tracking them, the team can identify pin points and focus improvement efforts effectively.</p>
<p>2.2 Have you experienced problems accurately identifying when a team is losing effectiveness?</p>	<p>Yes.</p>	<p>Yes.</p>

2.2.1 What causes the problem?	There is no structured system to identify or track effectiveness systematically.	It typically happens when there are disruptions, such as when a team member leaves or when external demands on priorities increase. These disruptions cause the team leader to struggle with goal setting, leading to slow progress, more defects, and additional stress within the team. Unclear or false requirements and poor priority setting also contribute to decreased output and overall team effectiveness.
2.3 What challenges have you encountered when measuring performance within teams?	I had to collect all the data myself, and while it gave me some new insights, it wasn't maintainable. It took too much time, and storage was limited. Any new KPI I wanted to track required a lot of manual input.	I struggled with measuring effectiveness due to unstructured data and inconsistent use of Jira across teams. For example, it's hard to link defects to specific features. Aligning how Jira is used would improve KPI accuracy.
2.4 What challenges have you encountered when measuring the effectiveness of separate individuals in a team?	Same challenges apply to both team and individual measures; lack of clarity about what to measure and lack of tools.	I have not tried that.
2.5 What would you like to improve in the current process for measuring performance?	In addition to data intake and presentation, the models need to be improved. It's important to understand what trends can be identified, possible causes, and solutions based on these trends. There is need for a better understanding of the models that form the foundation of the measurements.	The desire is to make KPIs more visible to help identify issues, even if they initially fail. Start with basic KPIs and refine them over time. Figure out how to gather and use these KPIs effectively.
2.6 Do you see any problems with measuring performance at an individual level versus the team level?	Measuring performance at an individual level can be sensitive, as it may create a feeling of constant surveillance. The goal should be to support individuals constructively, helping them improve productivity in a way that benefits both them and the team. Transparency and clear communication are essential to avoid the feeling of being monitored, with the focus on providing useful insights and support.	Measuring performance at the individual level can be insensitive, especially when highlighting negative aspects like bug counts, which can harm the team culture. It's more valuable to focus on team KPIs, such as cycle time, where team members can collaboratively address issues. Team-based discussions and improvements are more motivating and effective than focusing on individual performance.
Implication questions		
3.1 What do you think are the consequences of not having a clear method for measuring performance?	Without clear performance measurement, decisions are based on incomplete data, leading to poor choices, frustration, and inefficiency for both leaders and team members. It causes stress as expectations and assessments become unclear or subjective, impacting output in tight deadline environments. Clearer performance measurement could optimize workflows, reduce stress, and offer more flexibility in deadlines.	I would say frustration. Customers and external stakeholders become frustrated with unmet expectations, and employees feel stress from unresolved issues and lack of structured solutions, which may lead to good employees leaving. While it's not the sole reason, the absence of KPIs is a significant contributing factor.

3.2. What effects do you think an improved method of measuring efficiency would have?	Improved efficiency measurement would reduce stress, enhance team spirit, and strengthen leadership-individual relationships. It would lead to happier customers by ensuring timely delivery with fewer bugs. Even a few high-quality, actionable measurements would be a major improvement over having none.	An improved method would provide clearer visibility into how time and resources are being utilized.
3.3 What effects do you think an improved method of measuring effectiveness would have?	Same goes for effectiveness and efficiency.	It would ensure that teams are not only productive but are also delivering valuable outcomes.
Need-payoff questions		
4.1 Do you think that both aspects, effectiveness and efficiency, should be measured to the same extent?	Effectiveness is the more important one.	I would prioritize effectiveness first and then start looking into efficiency.
4.1.1 Why do you think so?	The most important thing is to ensure that you are doing the right things. You don't want to waste time or mental energy on the wrong tasks. Once you focus on effectiveness making sure the right work is being done, efficiency naturally follows because there's no unnecessary effort being spent. Additionally, maintaining effectiveness helps keep individuals and stakeholders in a productive mindset, which also tends to improve efficiency.	Because, when focusing on effectiveness, you automatically gain some efficiency as a result. Many KPIs overlap, contributing to both aspects, but focusing on effectiveness ensures that teams deliver value rather than just working efficiently on the wrong tasks.
4.2 How would a tool that measures performance improve the way you manage projects?	It would simplify planning, help detect disruptions early, and enable proactive communication with stakeholders. It would also improve problem forecasting, facilitate productive discussions, and prevent blame games or unspoken frustrations. Additionally, it could reduce burnout by promoting a healthier work environment and increasing job satisfaction.	It will provide insights when things are not going in the right direction and provide a starting point to talk to people and address issues, thus indirectly helping manage projects by providing clarity and structure.
4.3 What types of information would help you make better decisions about team performance?	Insights into individual and team performance trends, quality of work, and behaviors of team members.	*Already answered above*
4.4 How do you think that an improved measurement of effectiveness affects the team's long-term growth?	It strengthens team building by fostering collaboration, shared ideas, and mutual support. This synergy can make a team significantly more productive than individuals working alone. With good measurement tools providing actionable feedback, team spirit improves, leading to better results, long-term growth, and a more meaningful, enjoyable work environment.	It will aid in optimization of the team.

4.5 What would you like to see in a tool that provides a more detailed picture of the performance of a team?	I think, some actionable KPIs, those that we discussed during the brainstorm are really good and we did discuss why they could be good for our team also.	Cycle time is important because it predicts when a task will be completed, and scope changes should be tracked since sudden shifts cause efficiency loss. Measuring how clearly work is defined, including acceptance criteria, is also valuable. The focus is on the team as a whole, and there are various KPIs to consider.
---	---	--

Table 7 - Results from SPIN interviews with Software developer and Software engineer.
Source: Appendix 8 & 9.

Interview question	Software developer	Software engineer
Situational questions		
1.1. How are you currently working with measuring performance in terms of efficiency within development teams?	We measure efficiency based on time, ensuring delivery on schedule. The focus is on making implementations functional, even if not perfect, to keep clients satisfied. However, for important features, we postpone delivery to ensure a better implementation and avoid future issues.	I think it's in Story Points and Velocity.
1.2. How are you currently working with measuring performance in terms of effectiveness within development teams?	Measuring effectiveness is not constant, as we adjust our approach to maintain efficiency. It's a balance, with both parts at a good level.	I think, how much repetitive work is done or how many times a ticket is going back. Actually, there is no measure for it to review it, there's just like a small interpretation.
1.3. How often are follow-ups made regarding the teams' performance?	Follow-ups on team performance happen during retrospectives, where we discuss what went well and what didn't, then try to implement improvements in the next sprints. We also use Jira tools like burndown and velocity charts to track progress. Additionally, we have sprint planning and sprint review meetings at the end of every two-week sprint.	Once in half a year. It's in the form of a discussion.
1.3.1. What methods do you use to follow up the work of the team?	*Already answered above*	Not sure, no. It is just verbally.
1.4. How well do the current practices capture the performance of a team, in your opinion?	Senior members provide guidance through discussions and support. An onboarding plan is being developed to help newcomers integrate into the project and workflow more easily. Management is in place and continuously working to simplify the process.	There is a lack of history, lack of like references and no action plans. Actually, we have sprint-review, where we are discussing basically what was wrong and what should be done better and so on, this is done like every three weeks or at the end of every sprint.

1.5. How well do the current practices capture the performance of separate individuals in teams, in your opinion?	Individual performance is not formally measured, but some estimations are made based on past data and discussions. The Scrum Master calculates how many Story Points the team should handle, as Jira's charts do not account for factors like vacations or task complexity.	It is only done on team level. I think if there are any issues then those should be solved inside the team not by some KPIs.
Problem questions		
2.1. What challenges have you encountered when measuring performance within teams?	Every sprint is challenging because team members have different opinions on the number of Story Points to use, requiring compromises. Unclear or missing information can make estimation difficult, leading to additional discussions with managers or backend teams, which takes time.	Not me, because I think I'm not the right person to like make this particular reviews and decisions.
2.2 What challenges have you encountered when measuring the performance of separate individuals in a team?	The estimations are done verbally and on paper rather than through Jira.	*Already answered above*
2.3 What would you like to improve in the current process for measuring performance?	Having well-described Jira tickets is essential, as poor descriptions make estimation difficult and lead to inaccurate "guesstimates." Additionally, effectiveness in achieving goals depends on teamwork, when developers collaborate and discuss issues with backend and managers, solutions are found faster and more efficiently. Team-programming sessions or brainstorming meetings ensures better alignment and prevents rework based on unexpected client feedback.	Not sure, maybe yes but not sure what it should be.
2.4 Do you see any problems with measuring performance at an individual level versus a team level?	Not really, as individual performance can be observed from past sprints, where completed Story Points per person provide a basis for prediction. Team performance is essentially a summary of individual contributions, though factors like vacation days and sick leave must be considered. Estimating one person is harder than estimating a team, but the process is fundamentally the same.	So, as I had mentioned previously, I prefer team level more.
2.4.1. In what ways is the work being affected by measuring performance at an individual level, in your opinion?	If you say that "okay, no worry I will finish the thirteen Story Points in this sprint" and then for some reason you have headaches or I don't know something happened, it's a psychological part, it doesn't affect the real work, it's a pressure. It is much easier to say "okay, we as a team we will try to accomplish that much of Story Points" and everyone can like do as much as he can because in this way there is not any conflict between the developers.	Then it will affect the work so it's some kind of pressure added.

Implication questions		
3.1. What do you think are the consequences of not having a clear method for measuring performance?	Without measuring performance, it's unrealistic to track how much work can be delivered in a sprint. While individual measurement isn't a big deal, knowing past performance helps estimate team capacity. However, individual tracking adds pressure, while team-based responsibility reduces blame and allows for support in case of personal issues.	Money loss, the correlation between the delivering and the costs is affected directly. If the team itself delivers, if there are any issues in the team it is better for the team to make, at least I think when the issues are made on team level team themselves know where the problem is and they should have the option to solve it. So measuring individuals wouldn't matter that much.
3.2. What effects do you think an improved method of measuring efficiency would have?	One thing we don't like is having meetings for meetings and statistics for statistical data. the primary focus would be on the effectiveness, efficiency and the delivery. Data is good, but if developers spend too much time creating it instead of coding, performance is impacted. A dedicated person handling statistics would be fine, but usually, companies want to spend less and expect developers to do both.	I'm not seeing it as a good thing, more KPIs. In my opinion, more KPIs add stress and don't show the real illustration of how we're working, as every sprint or block has its own context. More charts are just a reference for managers to ensure everything works, but without context, they lack essence.
3.3. What effects do you think an improved method of measuring effectiveness would have?	*Already answered above*	It depends on what the goals are. So it should be related to goals, otherwise it's just to make sure that the graphs look okay.
Need-payoff questions		
4.1. Do you think that both aspects, effectiveness, and efficiency, should be measured to the same extent?	Yes.	Yes, but maybe efficiency is more is more important, maybe.
4.1.1. Why do you think so?	I mean this is why, in the beginning I didn't separate them. Because agile is like we must be adaptable, in one sprint we must achieve one goal, in another sprint make some code improvement and some documentation. So, it's kind of both 50/50 but it depends on the sprint by sprint and what kind of goals we have to achieve. Both are important yes.	Because again, if we are differencing to error reports or any follow-ups by history, the efficiency is easier to demonstrate and it's easier to show or to illustrate. Effectiveness, I'm not sure about that. It's more about how people interpret, it's more subjective.
4.2. Specifically, in what way, do you think such tools help prevent problems, such as effectiveness dip?	Tools can help improve effectiveness, but the current ones don't always provide realistic or precise data for the team. The tools are useful for tracking the sprint's progress, but their output often doesn't match the team's specific needs. Ultimately, the team has the most insights into its effectiveness, as they are directly involved with the product and communication.	In some cases it can help.
4.3. How do you think that an improved measurement of effectiveness affects	Improved measurement of effectiveness, in terms of some charts, not really, but in terms of retrospective, sprint planning, and sprint-review, of course. We have a meeting where we must discuss our	I think it's also important how or what actions are taken after those measurements because sometimes those measurements could be taken silently with some overall reviews. Maybe the period of reviews should

the team's long-term growth?	problems, but it's internal. Managers don't know about it, they know all the charts and statistics, but for us, we don't need statistics, we know we failed in some part and need improvement.	be bigger and not on every sprint or on every small period.
4.4. What would you like to see in a tool that provides a more detailed picture of the performance of a team?	As I mentioned earlier, this data is for managers. For us as developers, statistics like Story Points completed don't provide practical insights for solving problems. We handle problems directly, know how to solve them based on experience, and don't rely on external data to guide us.	I'm not sure about this, but I think the timelines and when things were promised versus when they were finished could be useful. We can achieve this by combining some charts, but we're sometimes missing that specific measurement. We don't have a chart for it, only a pipeline.

Table 8: Results from SPIN interview with Director of HR.

Source: Appendix 10.

Interview question	Director of Human Resources (HR)	
Situational questions		
1.1. How does the organization currently measure performance in software development teams?	The organization currently measures performance across the entire company using the same process. We use a system called Bamboo HR, which has an assessment module where both employees and their managers perform assessments. These assessments occur twice a year and focus on questions such as how valued the employee feels, whether they have what they need to do a good job, areas for improvement, and personal strengths. Additionally, we ensure the assessments are linked to goals and personal development plans, and consider individual growth trajectories, such as those new to the company or in new roles.	
1.2. How often are team performance reviews conducted?	Team performance is followed up twice a year, where it's documented in the system.	
1.2.1. How are these results used?	One-to-one meetings are held to discuss the results, identify any gaps, and ensure alignment on plans. These follow-ups also help to support progress and adapt development plans or goals as needed throughout the year.	
1.3. How often are follow-ups made regarding the individual's performance?	The performance of individuals is followed up twice a year, and additional discussions occur in one-to-one meetings.	
1.3.1. How are these results used?	*Already answered above*	
1.4. How well do you perceive the current methods capture software developers' performance fairly?	The current methods are generally filled out based on job descriptions, but there hasn't been feedback suggesting the need for a different assessment tool. The feedback hasn't highlighted the need for a more specialized competency assessment module. Overall, it seems to be working well within the given framework.	
1.4.1. Do these methods capture performance in a reliable way, in your opinion?	These methods capture performance fairly if both parties are engaged in the process. The involvement in development and the manager's commitment to maintaining the process are key. Historically, performance evaluations were more formal and felt like a screening rather than a meaningful dialogue.	
Problem questions		
2.1. What problems have you encountered when measuring performance at the team level?	Measuring at the team level is something we also do in our assessments to add pragmatism. If a team consists of only high-performing, highly experienced members who can carry the role, there is a risk that they will all want to move on and need more stimulation. If a team is structured with a spread of experience levels for natural succession, that is something we can analyze at the team level. However, measurement cannot be done only at this level; the individual must also be considered. Culture must also be assessed because if only	

	competence is rated without measuring culture, relationships, and communication, it can create the wrong team dynamics, where someone highly skilled still causes significant discord. It is essential to see the individual behind team performance as well.
2.2. What problems have you encountered when measuring performance at the individual level within teams?	If a team carries someone experiencing a performance dip, the employer cannot see what the individual needs support with. It could be life circumstances or health issues creating obstacles, and these are things we want to help with and guide in the right direction. But it can also lead to demoralization if others feel that "he/she always gets away with it, so why should we try?", creating a negative team culture. There have been cases where strong collegial support covered for others, leading to unfortunate outcomes. We would have wanted to step in earlier to provide support and be proactive. This is just an example of why it is important to connect the individual with the team as a whole.
2.3. What ethical risks do you see in measuring individual performance within teams?	The ethical risks of measuring individual performance within teams include ensuring proper data handling, access control, and compliance with privacy laws such as GDPR. It's important to clarify ownership of data, how it will be used, and obtain consent from users. Transparency about the purpose of the tool, how data will be shared, and the ethical use of assessments are key to managing these risks.
2.3.1. How could these risks be managed sustainably?	To handle these risks sustainably, it's important to ensure compliance with GDPR, including data retention policies, data removal requests, and how long trends remain relevant. Building trust and transparency is key, ensuring individuals understand how their data will be used and giving them visibility into the process. The more transparent the system is, the more it can foster a sense of security, allowing individuals to share feedback and learn from each other. A transparent, secure system can provide valuable insights and be used effectively in team discussions, reflections, and lessons learned.
Implication questions	
3.1. What do you believe are the consequences of not having a clear method for measuring performance in terms of effectiveness?	Not having a clear method for measuring performance makes it easy to draw incorrect conclusions. While you don't need a system for everything, having relevant examples and data, such as KPI trends, is important for making informed decisions and providing constructive feedback. Trust is key, if there is no clarity or understanding of what is being measured, motivation and engagement can be lost.
3.2. What do you believe are the consequences of not having a clear method for measuring performance in terms of efficiency?	The consequences of not having a clear method for measuring productivity are similar to those of not having a clear method for measuring performance. To find the right approach, it's important to include all parties involved and clarify what is being measured and what the data is saying.
3.3. What impact do you think a performance measurement tool would have on the work environment?	The consequences are similar to those of not having a clear method for measuring performance. To find the right approach, it's important to include all parties involved and clarify what is being measured and what the data is saying.
3.4. What impact do you think a performance measurement tool would have on employee well-being?	I would say that we go back to the idea of anchoring everyone and explaining that we use it. I can't assess how exactly this measurement tool could affect things, but I can say how to work to make it have the best effect.
Need-payoff questions	
4.1. What are your views on measuring individual performance?	I would say, without going into too many details, that it's a good complement to look at how the teams, development, and individuals progress. We can definitely find areas in the business where we need to improve processes or focus more on quality. But I also believe that the individual needs a broader assessment, looking at what more is required in their role to be successful, which might not be visible in these evaluations.

4.1.1. Why do you believe that?	*Already answered above*
4.2. What methods would enable a good balance between the need to measure individual performance and ethical considerations, in your opinion?	*Already answered above*
4.3. What features would you like to see in a tool that helps measure performance in a way that considers individual contributions?	I have too little insights into it to provide an answer, and that's not a negative thing, it's something I can get into later.

4.2 Prototype

The Prototype chapter begins with a presentation of the final versions of the KPI dashboard prototypes, presented in Figure 3 (Team Lead/Manager view) and Figure 4 (Developer view), which also are the results of MVP test 3. Two views of the KPI dashboard are created since data on other team members is not allowed to be shown amongst developers. Furthermore, the different KPI reports are explained in detail in Figures 5-9.

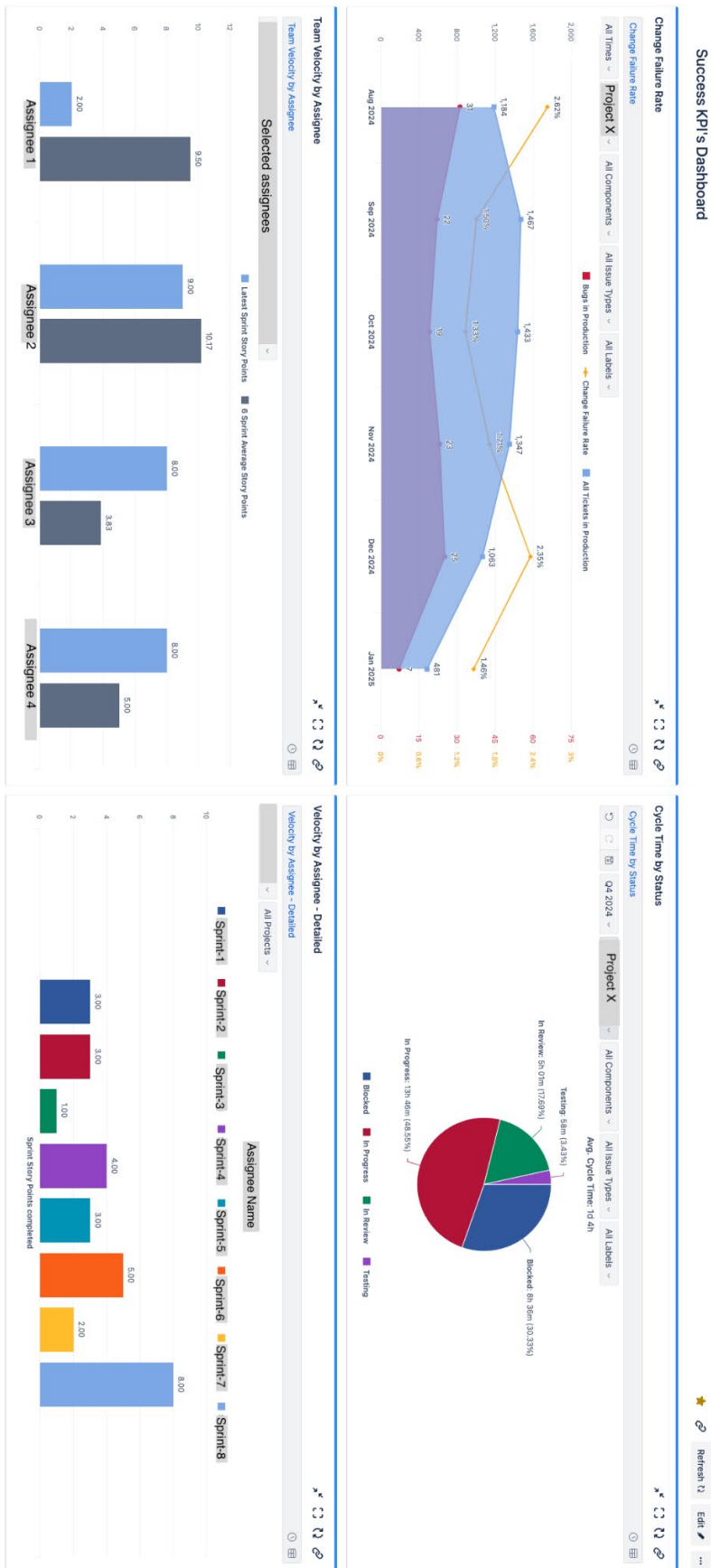


Figure 3: MVP 3. Screenshot of the team lead/manager view of the KPI dashboard.
Source: Table 10.

In Figure 3, a screenshot of the actual KPI dashboard in Jira is presented, with the view for a team lead or manager. The KPI dashboard consists of the KPIs Change failure rate (see Figure 5), Cycle time by status (see Figure 6), Team velocity by assignee (see Figure 8), and Velocity by Assignee – detailed (see Figure 9.1). The first two reports present measures on team level and the Team velocity by assignee present measures for the separate individuals in a view for the whole team in one report. The last one, Velocity by assignee – detailed, presents measures for only one individual at the time.

The given view of the KPI dashboard is designed to provide insights into individual performance by presenting key efficiency and effectiveness KPIs in a structured and accessible format, with two team-level KPIs for a broader understanding of team dynamics and overall performance. The KPI dashboard allows team leads or managers to track trends of their team members, identify potential performance bottlenecks, and compare individual contributions over time.

Success KPI's Dashboard



Figure 4: MVP 3. Screenshot of the developer view of the KPI dashboard. Source: Table 10.

In Figure 4, a screenshot of the actual KPI dashboard in Jira is presented, with the view for a developer. The KPI dashboard consists of the KPIs Change failure rate (see Figure 5), Cycle time by status (see Figure 6), Cycle time by status – current assignee (see Figure 7), and Current assignee velocity (see Figure 9). The first two reports present measures on a team level. The last two KPIs present the developer with a view of their own performance measures. The developers do not have the ability to see other individual team members’ performance measures.

The given view of the KPI dashboard is designed to provide insights into individual performance by presenting key efficiency and effectiveness KPIs in a structured and accessible format. The KPI dashboard allows developers to track trends of themselves, identify potential performance bottlenecks, and self-evaluate over their performance over time.

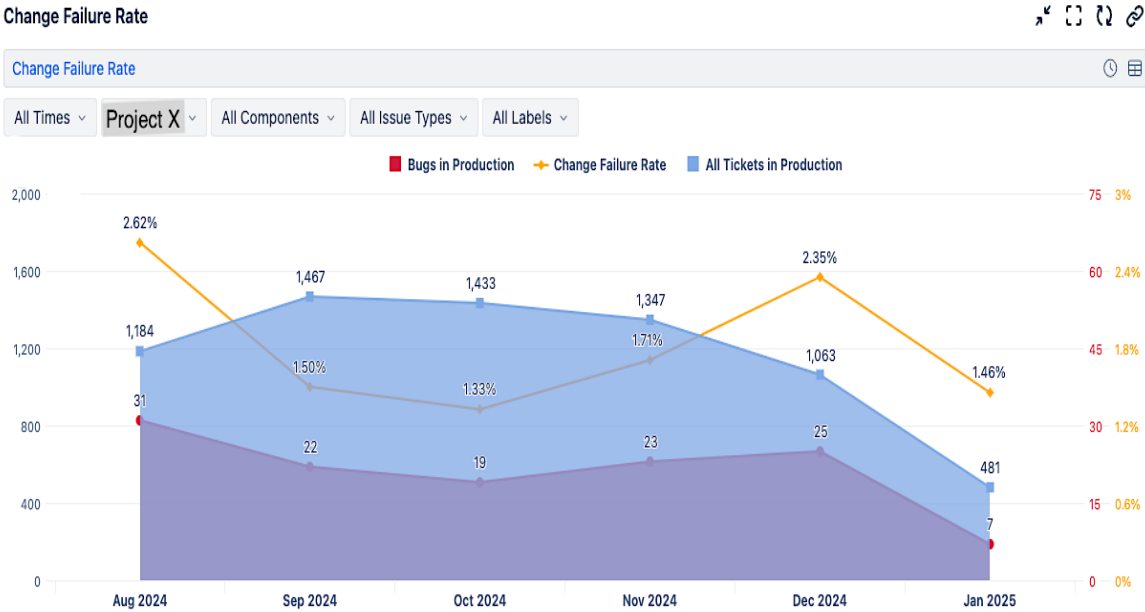


Figure 5: Change failure rate

In Figure 5, the Change failure rate KPI is presented with an area line chart that presents the percentage of deployments that caused a failure (bug) in production. Three important components are presented: The total number of all Jira issues (also named Jira tickets) that have went to production (blue area line), how many of those Jira issues were actual bugs (red area line) and the Change failure rate (yellow line) that shows what percentage of Jira issues in production that are bugs. For example, if the Change failure rate is 2.35%, it means that out of all Jira issues in production, 2.35% of them were bugs.

The user has several options to alter the chart by clicking on the drop-down menus: the option to choose what time frame the report should display, making it easy to spot trends, and the option to choose a specific project, specific component, issue types, and labels. In Figure 5, the report is set to present a view over the last 6 months, broken down by each month for a project that has been sensured.

In the y-axis to the left, a scale of the number of Jira issues is presented, and on the right side, a scale of the number of issues as well as a scale of the percentage of Change failure rate is presented. A lower Change failure rate generally means better quality of releases and more stable software for end users.

It is possible to “drill through” all the specific bugs that contribute to the values displayed in the report, as presented in Figure 5.1, by right-clicking on one of the values in the red line for bugs. However, the report only presents bugs found after deployment within the selected timeframe, meaning the report does not immediately indicate which specific deployment each bug belongs to. To determine the corresponding deployment, the user can manually check the Jira issue created for that specific bug.

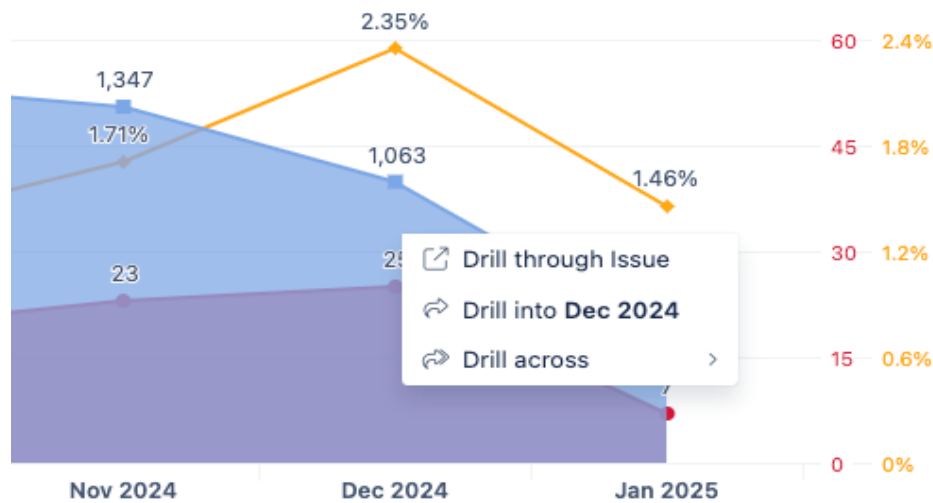


Figure 5.1: Presentation of the drop-down that allows users to drill through the Jira issues when right-clicking.

After clicking on the first option to drill through the issue, presented in Figure 5.1, a table will appear with all the specific Jira issues that have been included for that particular value (see Figure 5.2). For example, if clicking on the number of bugs existing for December 2024 in Figure 5 (which is 25 bugs), it is possible to easily see a table of all the bugs that appeared in December 2024.

Drill through Issue. Total value: 25; row count: 25

Filters: All Times | All Projects | All Components | All Issue Types | All Labels

Dec 2024	Jira Issue Name	Bugs in Production
	Jl-0102 Calendar does not show tasks from older dates	1
	Jl-1124 Templates do not follow the permissions	1
		1
		1
		1
		1
		1
		1

Buttons: Open all in Jira Issue Navigator | Close

Figure 5.2: Table of specific Jira issues that have been included for a particular value.

In Figure 5.2, The real names of the Jira issues are scensored, but consist of two fictive Jira issues to illustrate how the appearance of the Table is. Jl-0201 and Jl-1124 are fictive Jira issue codes, followed by the Jira issue names.

Cycle Time by Status

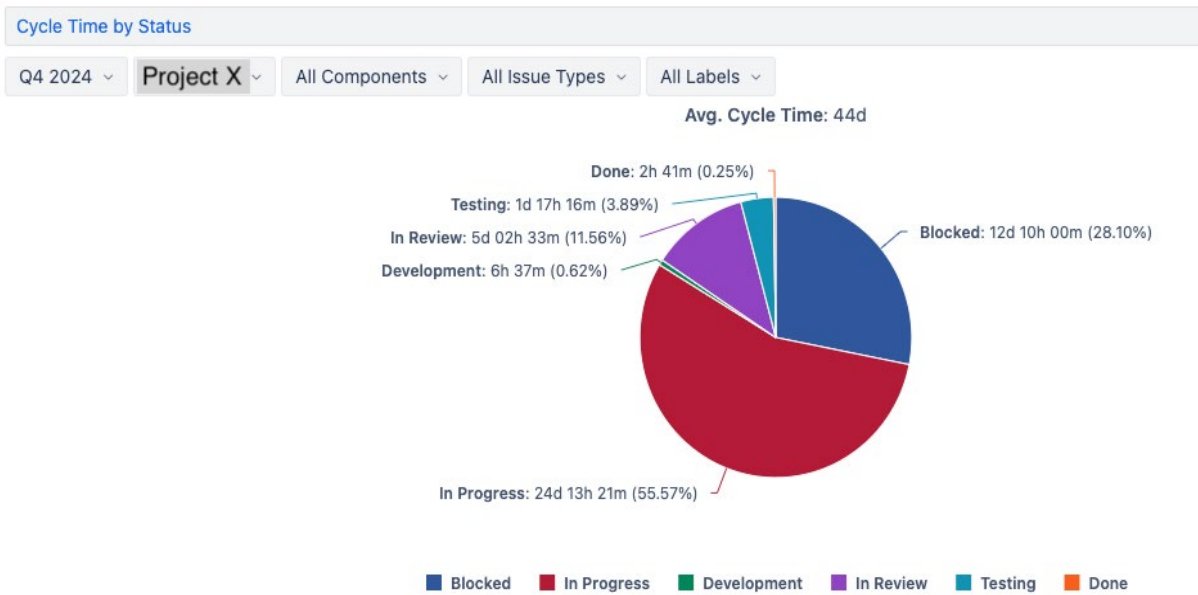


Figure 6: Cycle time by status

In Figure 6, the Cycle time by status KPI is presented, which shows how much time Jira issues spend in each stage of the workflow. In the pie chart, the average time in workdays is displayed, with the statuses *blocked*, *in progress*, *development*, *in review*, *testing*, *delivered*, and *done*. What statuses are represented depend on what statuses the teams use in their Jira project board. Therefore, the visualization of what the report displays may vary from team to team.

The labels show both the actual time values and percentages. Currently, the report is showing data for Q4 2024, but the user can change the time period using the filters. The user can also filter the data by Specific projects, components, Issue types (like bugs, stories, tasks), and Labels.

The percentages help you quickly see which statuses are taking up the largest portions of your overall cycle time, making it easier to spot where issues might be getting stuck or taking longer than expected.

Cycle Time by Status - Current Assignee

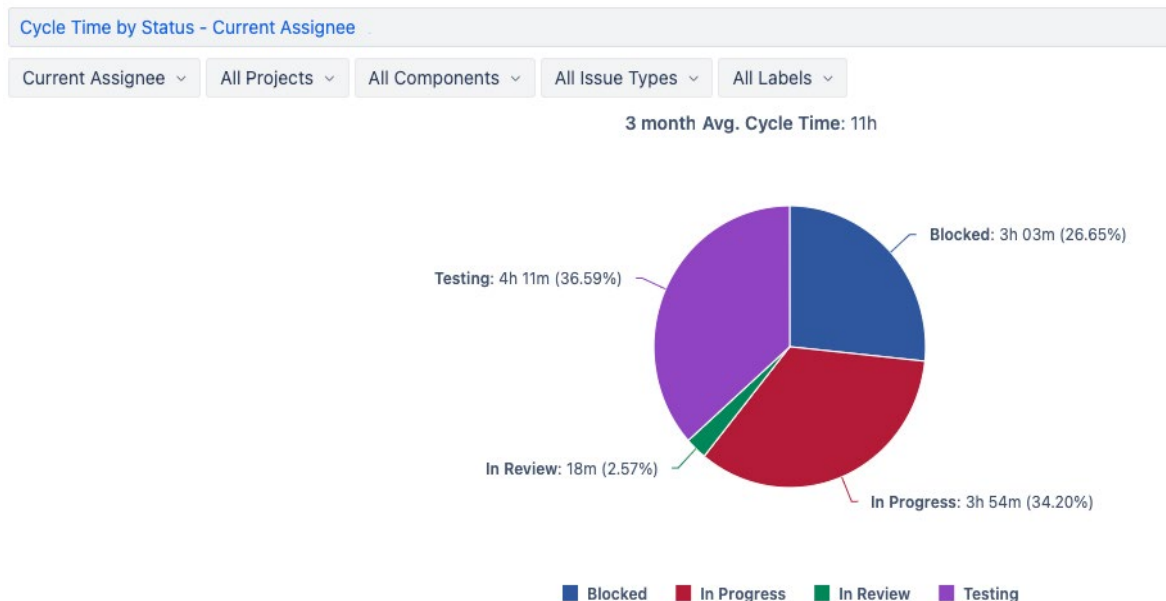


Figure 7: Cycle time by status – current assignee

In Figure 7, the Cycle time by status KPI is presented, which shows the same content as the Cycle time by status KPI in Figure 6. The only difference is that this report is set to only present data of the user that is logged in, in Jira (Current assignee). It is not possible to choose another team member in the filters.

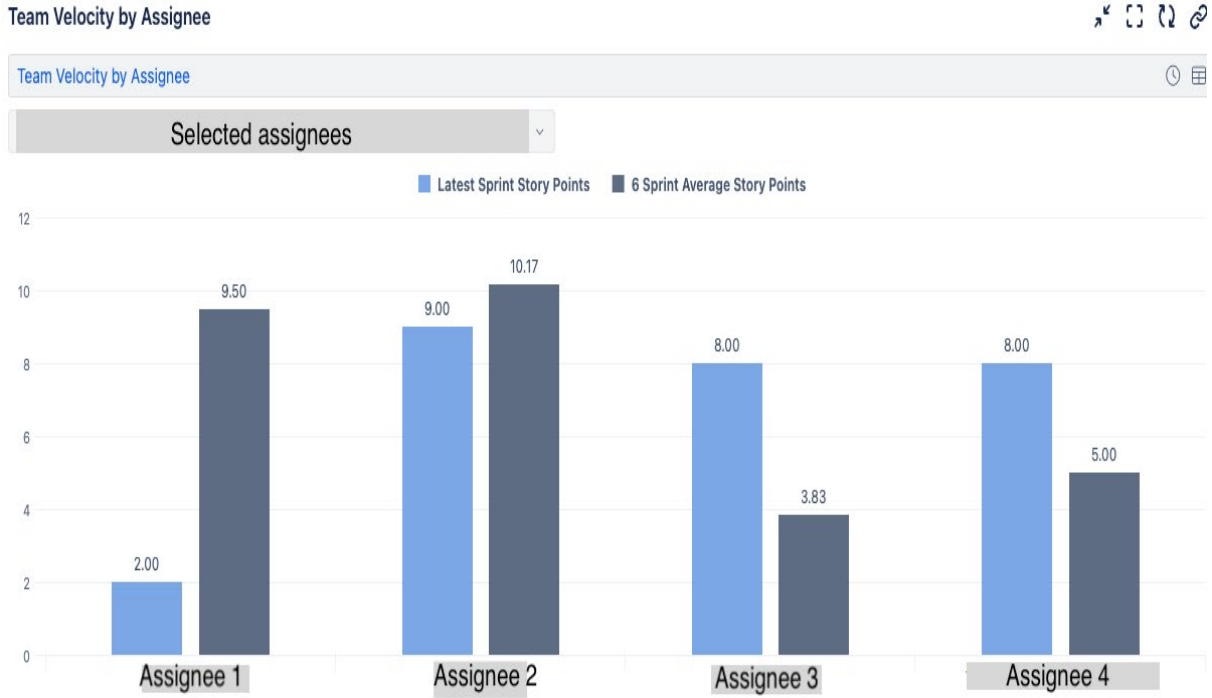


Figure 8: Team velocity by assignee

In Figure 8, Team velocity by each team member (assignee) is presented with a bar chart. A report for the team lead or manager to see the team’s Velocity by each team member in one report. This is a Velocity report with a view of the whole team that helps track individual team member performance, all in one report. It compares two key KPIs: Latest sprint story points (blue bars) - shows how much work each team member completed in their most recently completed sprint. 6 Sprint average story points (gray bars) - shows each team member’s average performance over the last six completed sprints. This provides context about team members’ typical delivery capacity. The comparison between these two KPIs helps identify if someone is performing above or below their usual capacity, who might be overloaded or have the capacity for more work, and the consistency of delivery across the team. The vertical bars make it easy to compare both current and average performance across team members at a glance. The labels above the bars are the number of story points completed. In the y-axis, a scale of a number of story points is presented, and in the x-axis, the names of different team members are presented. The current report shows the value of the most recent sprint at that time, which corresponds to a time span of two weeks at the case firm.

Current Assignee Velocity

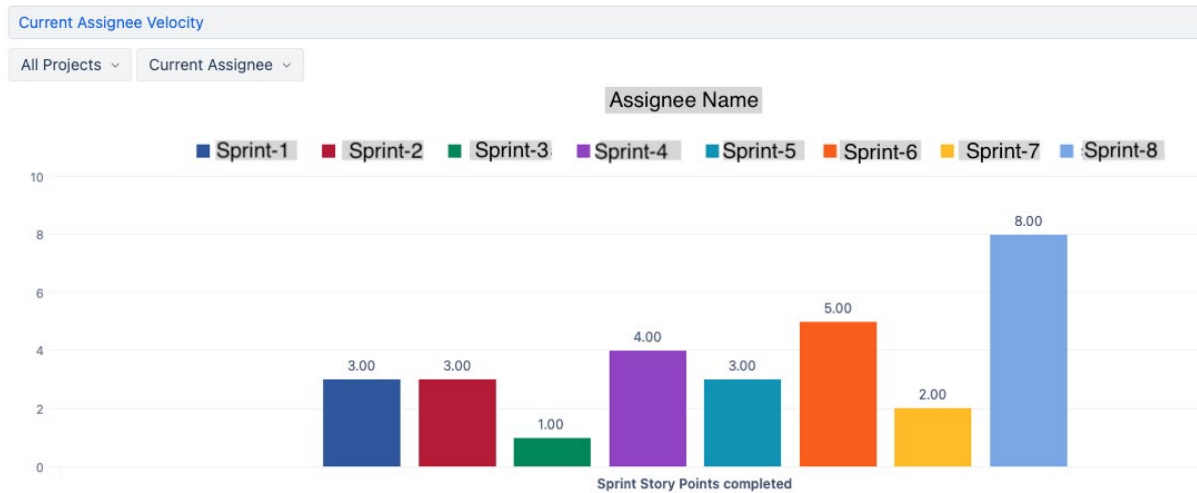


Figure 9: Current assignee velocity

In Figure 9, a measurement of the Current assignee velocity is presented with a bar chart. This report visualizes and how many Story points a team member has completed in different sprints. Higher numbers indicate more complex work or a larger amount of work completed. The report is set to Current assignee, which the user can not change. Meaning, the report only shows data for the individual who is currently logged in, in Jira, and watches the KPI dashboard. This ensures that team members can not see each other's measures. The report shows performance across multiple sprints, which helps identify patterns in individual velocity over time. The data labels show exact values. The sprints are represented with different colors and fictive names, such as 'Sprint-1'. The current report shows 8 sprints, which corresponds to 18 weeks, as one sprint has a duration of two weeks at the case firm Extenda Retail.

Velocity by Assignee - Detailed

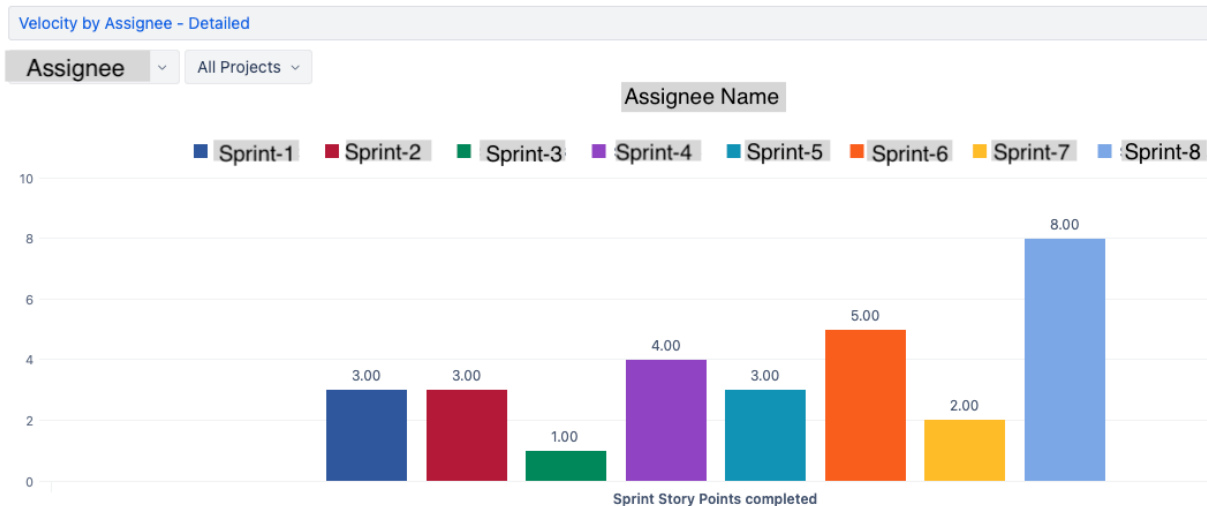


Figure 9.1: Velocity by assignee – detailed

In Figure 9.1, the Velocity by assignee – detailed KPI is presented, which shows the same content as the Current assignee velocity KPI in Figure 8, but with the possibility for team leads and managers to choose specific team members whom they want to analyze more deeply.

4.3 MVP 1

The first MVP is developed based on literature study and SPIN interviews (See Tables 6-8), as well as a probing dialogue (see appendix 2) and a brainstorm (See appendix 3). MVP 1 consists of two views, one for the team leads/managers (see Figure 10) and one for the developers (see Figure 11).

In the first prototype (see Figures 10 & 11), the calculation for the Change failure rate KPI (see figure 5) was different, using a special filter in Jira issues named “discovered in”, with the option to choose at what phase the Jira issue was discovered. This is a crucial filter to use when measuring bugs in production since the only way to detect bugs in production is by using the Jira issue filter correctly. In MVP 1, the calculation was to also measure all Jira issue types discovered in production. Additionally, the name of the blue area line was Total issues in production and the timeframe was set to 6 months without the option to choose timeframe.

The Cycle time KPIs (see Figures 6 & 7) started off by being set to show the measure for three months, without the option to choose timeframe (see Figures 10 & 11).

The team lead/manager view initially consisted of three KPIs (see Figure 11).

Success KPI's Dashboard



Figure 10: MVP 1. A screenshot of the developer's view of the KPI dashboard. Sources: Tables 6-8, Appendices 2 & 3.

Success KPI's Dashboard

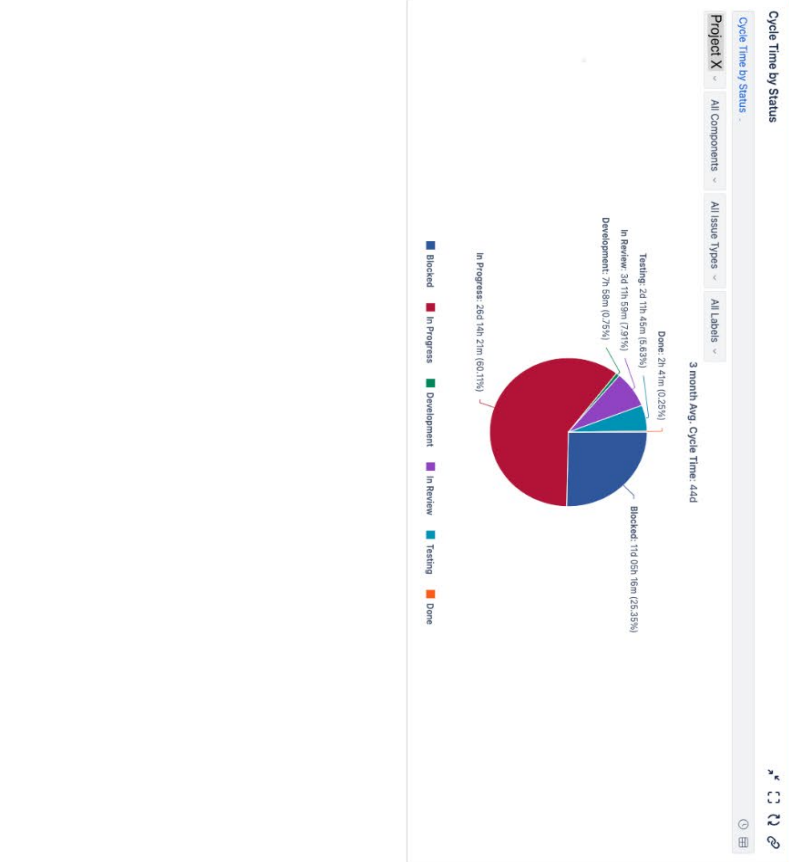


Figure 11: MVP 1. A screenshot of the team leads/managers view of the KPI dashboard. Sources: Tables 6-8, Appendices 2 & 3.

Table 9: Results from the MVP 1 tests.
Sources: 13-17.

Role	Feedback
Team lead B	<ul style="list-style-type: none"> ● Likes the ability to look at the Change failure rate, but wants the ability to adjust time range. ● Acknowledges that the KPIs don't measure team members who spend time aligning others or sharing knowledge. ● Wishes to see more sprints than just the last one and average of six sprints for the Team velocity by assignee KPI
Senior frontend developer	<ul style="list-style-type: none"> ● Found incorrect data in the Velocity by assignee KPI due to a Jira issue not being marked as "Done." ● Requested the ability to filter data by individual for the Change failure rate KPI. ● Pointed out that the Cycle time KPIs helps identify bottlenecks. ● Suggested adding a global filter for projects in the KPI dashboard, that could be applied to all reports. ● Found the Velocity by Assignee and Cycle time KPIs very useful, but had concerns about data accuracy. ● "I can see that a developer is delivering two story points, but maybe the issue is not that a developer is working only on two story points, but the fact that the developer finishes 16 story points, yet out of those 16 story points, the QA [Questions & Answers) team can only test two story points. Then I will know that the bottleneck is not on this developer's side but on the QA team." ● Would like to be able to compare their performance with the rest of the team to know at what standard they are performing.
Engineering manager	<ul style="list-style-type: none"> ● Recognizes the usefulness of the Team Velocity by Assignee KPI and that in combination with observation it can be really useful. But cautions that it can be misleading if taken out of context. ● Emphasized that velocity doesn't necessarily equal value and that the KPI should not become the main focus of the team. ● Found the "In Progress" status in the Cycle Time KPI confusing and suggested adding more detail by breaking down other Jira statuses. ● Pointed out that the Velocity by Assignee KPI was not working for one team and suggested documenting how to use Jira correctly to ensure accurate data. ● Found the Change failure rate measure concerning and expressed a need to better understand the KPI. ● Overall, found the dashboards very useful and emphasized the importance of having accurate data and a proper understanding of how to read the KPIs.
Software engineer	<ul style="list-style-type: none"> ● Did personally not find the dashboard useful. Found it more useful for management. ● Found the Change failure rate graph confusing.
Team lead A	<ul style="list-style-type: none"> ● Believes that smaller tasks would lead to better data for the Individual Velocity KPI. ● Thinks the Individual Velocity KPI should be considered alongside other factors and not become the main goal of developers. ● Wishes to see more sprints than just the last one and average of six sprints for the Team velocity by assignee KPI. ● Emphasized the importance of understanding trends in the KPIs, not just snapshots. ● Believes the Cycle time by status KPI and the Velocity KPIs can together help identify team spirit problems or codebase complexities. ● Thinks the KPIs can improve communication with stakeholders by providing a clearer picture of team performance and potential delays.

- Sees the potential for using the KPIs to set goals and track progress in one-on-ones.
- With individual KPIs, it is possible to see when a problem may arise when someone is overburdened, causing stress that becomes visible in interactions, like frustration in daily meetings. This can lead to a sense of needing to compensate for others.
- Overall, sees the dashboard as a "massive improvement" compared to having no analytics.
- Thinks the KPIs are valuable because they not only drive improvements in performance but also enhance the teams work processes. By visualizing the data through graphs, the team can identify ways to optimize their workflow while ensuring quality. This insight encourages to continuously refine our processes for better outcomes.

4.4 MVP 2

In the second version of the prototype, changes have been made based on the feedback from MVP 1 tests (see Table 9). The changes included adding the function to be able to adjust the time range in the Change failure rate report, which previously was set to six months.

Some data accuracy issues were addressed, like story points not being registered in the Current assignee velocity and Team Velocity by assignee reports, if the Jira issues are not marked “Done” before a sprint ends. For example, if the status for a Jira issue is set to “deployed” or “quality assurance” by the time a sprint ends, it will not be counted for that sprint. This means that the Velocity reports may not show accurate amount of story points completed for a specific sprint. Unfortunately, this situation could not be fixed within EazyBI because the tool relies on the data provided by Jira, and it doesn’t have control over how Jira statuses are set or tracked during the sprint lifecycle.

Another data accuracy issue was identified for the Cycle time by status report. The time did not seem to be accurate, indicating a data configuration issue between Jira and EazyBI or poor data logging practices.

There was also some confusion about the statuses that the Cycle time by status report presented. This problem was addressed by giving additional information verbally to the test persons about what determines the statuses that are shown, (see explanation in Figure 6).

There was also a wish from Senior frontend developer to see the Change failure rate on an individual level, but this has proven to not be possible because the software development process of the team who the dashboard was created for, do not include connecting the occurring bugs to the individuals who caused them.

The Senior frontend developer suggested adding a global filter for projects in the KPI dashboard that could be applied to all reports. Unfortunately, this is not possible to apply in Jira dashboards because such a function does not exist.

Ultimately, there was a wish amongst Team leads to see more sprints than just the average of six sprints together with the last sprint (see figure 8). Therefore, I added the same report as Current Assignee Velocity to the team leads/managers view of the KPI dashboard (see figure 9) and made the small change to disable the Current assignee function. This made it possible for team leads and managers to analyse deeper into separate team members’ Velocity.

Based on the feedback from MVP 1 tests, MVP 2 was developed (see Figures 12 & 13).

Success KPI's Dashboard



Figure 12: MVP 2. A screenshot of the developers view of the KPI dashboard. Source: Table 9.

Success KPI's Dashboard

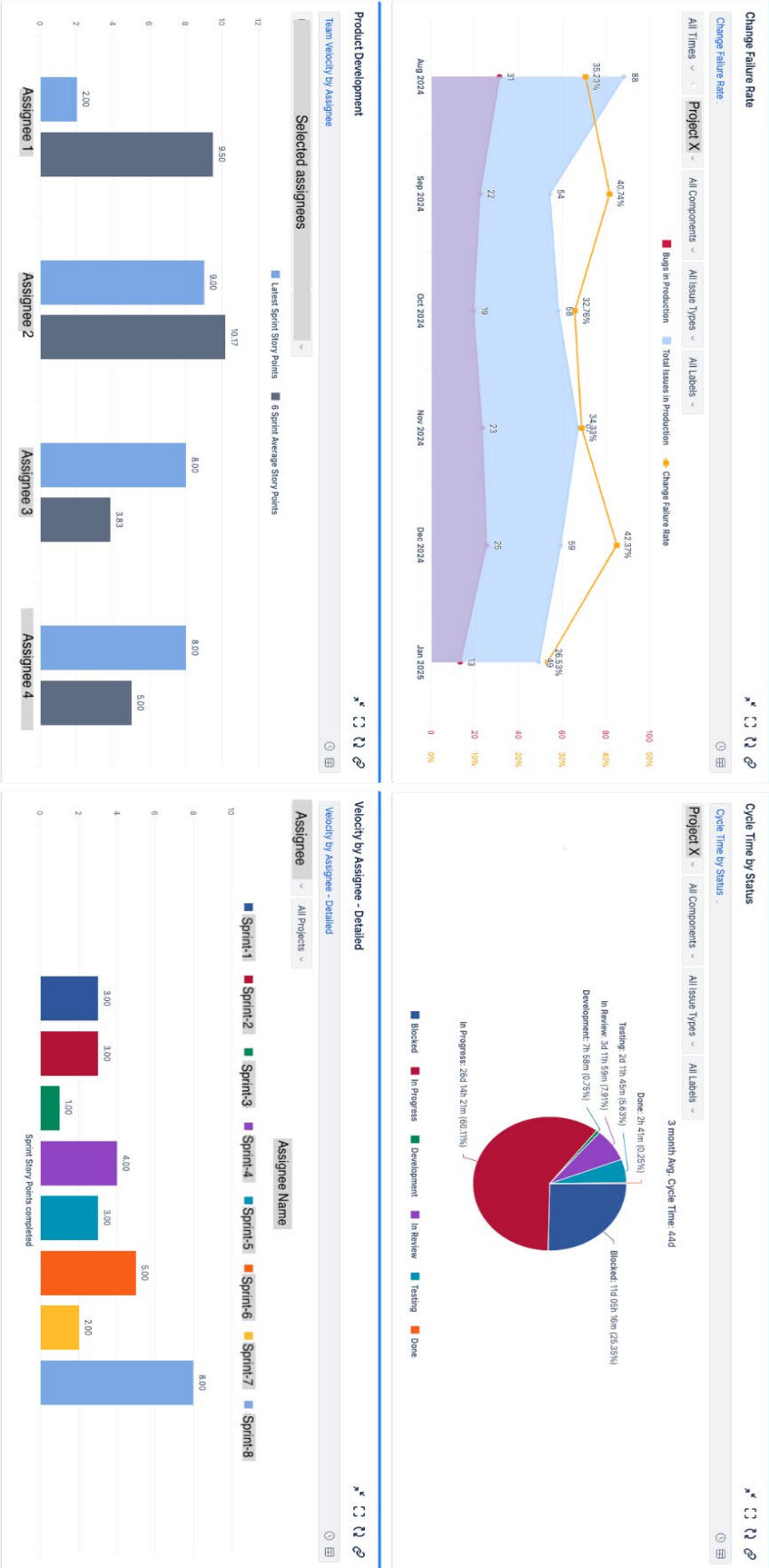


Figure 13: MVP 2. A screenshot of the team leads/managers view of the KPI dashboard. Source: Table 9.

Table 10: Results from the MVP 2 tests.
Sources: Appendices 18-22.

Role	Feedback
Team lead B	<ul style="list-style-type: none"> Was surprised that the Cycle time by status measure wasn't "all in progress," as their workflow is simplified. They expected some time to be logged on the backlog, but this wasn't reflected in the data. Was initially confused about the terminology used in the Change failure rate KPI. After better understanding of the Change failure rate KPI, explained interest in seeing the relationship between the number of Jira issues completed and the number of bugs. Explained that all all Jira issues go to production and are deployed when they are marked as done, indicating that the "Discovered in production" filter does not have to be used when it comes to Jira issues in production. Liked that the dashboard layout could be customized.
Senior frontend developer	<ul style="list-style-type: none"> Did not see the benefit of the Change failure rate KPI if the bugs are not traceable to the individual causing it. Thought the data in the Cycle time by status was not accurate because it did not reflect time spent on Jira issues correctly. Wished to see working days divided by the delivered story points. Believed the dashboard could be very helpful with some finetuning for data accuracy.
Engineering manager	<ul style="list-style-type: none"> The grouping of "In Progress" statuses was too broad in Cycle time by status KPI. A feature in Jira goes through many stages and extracting these individual states would be useful. This can help identify where to focus efforts to improve effectiveness. States that consistency across projects using the same schema is essential for the dashboard to make sense.
Software engineer	<ul style="list-style-type: none"> Had not engaged with the dashboard since the last test and did not have any feedback to give.
Team lead A	<ul style="list-style-type: none"> Thought the changes made to the dashboard were good and that the small updates allowed them to dig deeper and find more information. Have started using the dashboard to organize thoughts and as material for morning meetings. Will be using the dashboard for one-on-one meetings with team members to discuss personal growth and setting quantifiable milestones. Points out that data accuracy is an area for improvement to gain better insights from the dashboard, as the current data is a bit messy, for example the Cycle time KPIs. Points out the need to ensure that everyone that are involved with the dashboard use Jira in a consistent way to collect the needed data.

4.5 MVP 3

In MVP 3, changes have been made based on the feedback from MVP 2 tests (see Table 10).

During MVP 2 tests, I learned that all Jira issues that are marked as "done" are in fact in production and deployed (see Table 10). Before, I understood that all Jira issues had to use the filed "Discovered in production" in order for the Change failure rate KPI to show accurate measures. Therefore, I recalculated the Change failure rate KPI and the measures became instantly more realistic.

Additionally, several MVP test persons found the Change failure rate KPI hard to grasp, likely to the miscalculation before, showing unrealistic values, but also because of the word 'issues' that was confusing. The sub-KPI Total issues in production was changed to *Total tickets in production*, with "tickets" having the same meaning as Jira issues.

There was also a wish to see working days divided by the delivered story points. After carefully analyzing the data availability for this kind of measurement, there was no automatically collected data on what days individuals worked or not. Therefore, this could not be developed.

Ultimately, since adding the option to be able to choose a timeframe for Change failure rate was highly appreciated, I decided to do the same for the Cycle time in status KPIs (see Figures 6 & 7).

Based on the feedback from MVP 2 tests, MVP 3 was developed (see Figures 3 & 4).

Table 11: Results from MVP 3 tests.
Sources: Appendices 23-27.

Role	Feedback
Team Lead #1	<ul style="list-style-type: none"> ● Thinks that the Change Failure Rate KPI made more sense after the changes. They found the new name for the KPI, “All Tickets in Production,” to be clearer. ● Liked the ability to customize the timeframe for the Cycle time by status KPI, noting that it was useful to drill down or look at a broader picture. ● Found the dashboard helpful, particularly the historical view for Velocity and the Change Failure Rate KPI. ● Also noted that the measures were more realistic now for the Change Failure Rate KPI.
Senior Frontend Developer	<ul style="list-style-type: none"> ● Did not have any new feedback to give and their feedback remained the same as the previous MVP test.
Engineering Manager	<ul style="list-style-type: none"> ● Found that the Change Failure Rate KPI was clearer after the changes made. ● Found the dashboard very useful, noting that it allowed them to look back at projects, see what’s coming back from production, and have discussions with teams on how to improve.
Software Engineer	<ul style="list-style-type: none"> ● Thinks that the new naming ‘Total Tickets in Production’ was clearer than "issues" for the Change Failure Rate KPI. ● Found the ability to choose a timeframe for the Cycle time by status KPI beneficial. ● Had not started looking into the live data yet.
Team Lead #2	<ul style="list-style-type: none"> ● Was pleased with the changes to the Change Failure Rate KPI and the addition of the ability to choose a timeframe for the Cycle time by status KPI. ● Felt the changes were necessary and that the dashboard was becoming more useful. ● Will use the KPIs in the dashboard in the future for sprint planning purposes. ● States that evaluation is now backed by data and not just an observation. ● Is really happy and pleased with the dashboard.

Since there were no improvements requested by the test persons after MVP 3 tests (see Table 11), MVP 3 remained the last version of the KPI dashboard prototype, and MVP 4 was not created.

5 Analysis

In this chapter, the primary empirical data presented in the chapter *Results*, including SPIN interview answers (see Tables 6-8), results from MVP tests (see Tables 9-11), and probing dialogues (see Appendixes 1-2), are analyzed. These findings are compared to knowledge and earlier experiences presented in the *Theory and Earlier Studies* chapter. The analysis follows a structured approach: firstly, key insights identified in the empirical data are outlined and then evaluated in relation to existing theory and earlier studies on performance measurement in agile teams. The aim of the structured approach is to identify both confirming evidence and potential gaps in current knowledge.

5.1 Key performance indicators

The main purpose of this thesis is to identify and describe KPIs for measuring efficiency and effectiveness in agile software development teams. A sub-purpose is to create a dashboard to measure individual performance using data from a project management tool.

The KPIs that were ultimately developed for the KPI dashboard were:

- Velocity: Team velocity by assignee, Current assignee velocity, Velocity by assignee - detailed (see Figures 8, 9 & 9.1).
- Cycle time: Cycle time by status, Cycle time by status - current assignee (see Figures 6 & 7).
- Change failure rate (see Figure 5).

Velocity is a well-established KPI for measuring effectiveness in agile teams, according to Budacu and Pocatilu (2018:71), Pham and Neumann (2024:6), and Natarajan and Pichai (2024:3). Pham and Neumann (2024:6) state that the Velocity KPI helps teams predict capacity and set realistic goals for future sprints. On the other hand, Almeida and Carneiro (2023:9 & 10) state that Velocity should not be used in isolation and should, therefore, be combined with other KPIs for a more comprehensive assessment of performance. Both of the Team leads and the Engineering manager said during the MVP 1 tests (see Table 9) that the Team velocity by assignee KPI (see Figure 8) has great value but within its context. They also addressed that Velocity can not be relied upon as the sole KPI for evaluating performance, aligning with what Almeida and Carneiro (2023:9 & 10) stated about not using the Velocity KPI alone for performance measurement. Team lead B further describes that the three variants of Velocity KPIs do not provide a more comprehensive view of individual contributions, as it excludes time spent assisting others or performing tasks outside assigned work items (See Table 9). This aligns with Teiber (2021:57) explaining that co-workers at Microsoft meet a similar challenge when using Velocity as a KPI for task completion. Although it effectively measures whether requirements were fulfilled to a higher degree within an iteration, velocity alone can not pinpoint the underlying causes of process variations.

A senior frontend developer mentioned that one of the benefits of using the Cycle time by status – current assignee KPI is its ability to identify bottlenecks in the workflow (see Table 9), which aligns with Power's (2014:5) statement that analyzing Cycle time helps to identify bottlenecks in the workflow. However, it is important to point out that the Cycle time KPIs (see Figures 6 & 7) do not always capture the correct value, as stated by Team Lead B, the senior frontend developer, and Team Lead A (see Table 10). This could have influenced the scarcity of feedback on what value the Cycle time by status (see Figure 6) and Cycle time by status – current assignee (see Figure 7) KPIs provided the MVP test persons.

During the literature review, no study of the Cycle time KPI at the individual level was identified. Through testing the Cycle time KPI at an individual level (see figure 7) in this study, the potential correlation between the Cycle time by status – current assignee KPI and the Velocity KPIs (see Figures 9 & 9.1) is identified. During MVP 1 test, a Senior frontend developer provided valuable insights, stating (see Table 9):

“I can see that a developer is delivering two story points, but maybe the issue is not that a developer is working only on two story points, but the fact that the developer finishes 16 story points, yet out of those 16 story points, the QA team can only test two story points. Then I will know that the bottleneck is not on this developer's side but on the QA team.”

This observation indicates that KPIs like Cycle time and Velocity, when analyzed together, can help identify bottlenecks and inefficiencies in the development workflow, such as delays occurring in specific teams like QA, rather than being linked to individual team members.

Team lead A also gives the feedback that the Cycle time KPI is a good one to include together with a Velocity KPI, as it enables detecting if the development in progress time starts to go up, and the team's velocity goes down, it might indicate a problem with the code base, project complexity, or team spirit (see Table 9).

Rüegger et al. (2024:39) acknowledge that the Change failure rate KPI can be difficult to calculate due to the challenge of matching incidents to specific deployments, which aligns with the challenge that I encountered in my study. I managed to make it possible to see the value of the Change failure rate for a chosen period (see Figure 5), and with the possibility to drill through all the specific bugs that are measured (see Figures 5.1 & 5.2), but that is the bugs found after deployment for that chosen timeframe and not does not directly show what deployment it belongs to at first glances. However, to see what deployment the bugs belong to, the user can manually check what deployment that specific bug belonged to in the Jira issue for that specific bug.

Overall, Team lead A states that the KPI dashboard (see Figure 3) is of use for sprint planning, which aligns with Pham and Neumann's (2024:7) argument that KPIs support future sprint planning.

5.2 SPACE framework

The Engineering manager noted that the Velocity KPIs (see Figures 8 & 9.1) could be misleading if used without context (see Table 9), which aligns with the SPACE framework's assertion that efficiency cannot be measured by a single dimension or KPI alone (Forsgren et al. 2021:2). Similarly, Team lead A emphasized the importance of analyzing trends rather than snapshots of data (see Table 9), which reinforces the need for a multidimensional approach to performance measurement, which the SPACE framework aims to provide (Forsgren et al. 2021:2).

Additionally, the needs of the case firm Extenda Retail was to get insights in the efficiency and effectiveness of the performance of individuals in teams (see Appendices 1 & 2), which aligns with the SPACE framework's goals of providing a balanced and insightful perspective on multiple dimensions, including efficiency and effectiveness.

The results of this study confirm that combining KPIs across different dimensions is necessary for a comprehensive understanding of performance, supporting the SPACE framework's recommendation to use at least three dimensions for a balanced and insightful evaluation of performance (Forsgren et al. 2021:17). In this study, the SPACE framework was used to guide the selection of dimensions for the KPI dashboard, focusing on the dimensions Performance, and Efficiency and flow. Due to data availability challenges (discussed in Chapter 3 *Methods*, section 3.5.3.3. *Challenges*), additional dimensions could not be incorporated. A planned Commits performed by time KPI, could not be developed due data limitations, leading to the KPI dashboard only representing two dimensions with the SPACE framework, which is less than recommended by Forsgren et al. (2021:17). The lack of representation of a third dimension may affect the effectiveness of the KPI dashboard by not providing highly sufficient depth for analysing performance.

The Performance dimension is represented by individual Velocity KPIs (see Figures 8, 9 & 9.1), and these KPIs effectively captured the outcome of the software development process, with the condition that Jira issues with story points are consistently marked as "Done" in Jira (see Table 9). It is worth noting that teams who have a different software development process, and more specifically teams who close their Jira issues by marking them as, for example, "Deployed" will either have to adjust the calculation of the Velocity KPI according to their software development process or change their software development process.

The Efficiency and flow dimension in the SPACE framework is represented by Cycle time KPIs (see Figures 6 & 7) and Change failure rate (see Figure 5). The Senior frontend developer stated that the Cycle time KPIs help identify bottlenecks (see Table 9). Furthermore, Team lead A states that team spirit problems or code complexities can be identified with the help of the Cycle time KPIs (see Table 9). The results of this study indicate that if there is no issues with the data accuracy, the Cycle time KPI is proven to be a good indicator of the dimension Efficiency and flow.

Similarly, the Change failure rate KPI effectively reflects efficiency, aligning with the dimension's goal of capturing how smoothly work progresses with minimal disruptions.

To summarize, the Cycle time KPIs, both on team level and individual level (see Figures 6 & 7) provided valuable insights into the overall workflow and identified bottlenecks, while individual Velocity KPIs (see Figures 8, 9 & 9.1) provided insights on individual contributions and outcomes of the software development process.

To the best of my knowledge, no prior studies have examined the application of KPIs using the SPACE framework or its effects. In this study, the SPACE framework was applied as a foundation for KPI selection in a dashboard. However, one of the main challenges encountered was that the SPACE framework is not fully suited to KPI measurement that is based on automatically collected data. Incorporating subjective KPIs, which are emphasized in the SPACE framework, proved difficult.

In conclusion, the SPACE framework provided a structured approach into KPI selection for individual performance measurement. However, fully leveraging the framework in practice requires a higher degree of data availability, reliable data sources, and consideration of subjective factors that are challenging to quantify automatically.

5.3 The degree of need for performance measurement

The Team lead A emphasizes that KPIs are highly essential for measuring progress, identifying issues, and making informed decisions (see Appendix 2). During MVP 1 test, Team lead A further explains the issues that are identified through measuring KPIs, which are team spirit problems or when a team member is overburdened (see Table 9). This aligns with what Almeida and Carneiro (2023:2) and Pham and Neumann (2024:7) emphasize, that performance measurement is a critical component of successful software development.

The director of HR at Extenda Retail emphasizes the importance of building trust and transparency in how KPIs are used, including ensuring that the KPIs are seen as tools for improvement, not surveillance (see Table 8). To ensure what perceptions different stakeholders of the KPI dashboard have, SPIN interviews (see Tables 6-8) and three MVP tests (see Tables 9-11) were conducted in this study with multiple stakeholders. Similarly, Boon et al. (2023:132) cautions that a nuanced approach should be taken, that is sensitive to the needs and perspectives of all stakeholders when adopting a KPI dashboard.

Team leaders often rely on their own observations and perceptions to assess developer performance, but these perceptions run a great risk of being biased and subjective (Oliveira et al. 2020:1-2). The same case stands for Extenda Retail, where currently, the only way to measure individual performance is by observation (see Table 6). During MVP 1, the Engineering manager states that the combination of the Team velocity by assignee KPI (see Figure 8) in the dashboard, together with observation, gives more meaningful insights (see Table 9). This relates to Oliveira et al. (2020:23) emphasizing that efficiency KPIs can complement the subjective perceptions of team leaders and that KPIs can reveal aspects of developer effectiveness that were not previously known or were underestimated. The results of this study clearly indicate that there is a degree of higher need for leveraging automatically collected data for performance measurement to support the perceptions of team leads and managers in the software development team investigated in this entrepreneurial study.

As for developers, the Software engineer believes that a team itself is best positioned to understand where issues lie, and measuring individuals, in the Software engineers view, is not important (see Table 7). Additionally, the Software engineer expresses concern that more metrics could add stress and lead to a focus on following the metrics rather than improving the work (see Table 7). Furthermore, the Software engineer do not find the dashboard useful, stating it was more useful for management, and do not find the KPIs insightful for their work (see Table 9).

The Software developer has the same opinion that team-based responsibility is better since they themselves already know how it is going without KPIs (see Table 9). Although the Software developer sees a risk of creating psychological pressure if measuring performance, the Software developer explained that they understood the necessity for team leads to see this kind of data and do not personally feel bothered about performance being measured. In contrast, the Senior frontend developer finds the KPI dashboard useful and would rather compare their performance with the rest of the team for self-evaluation (see Table 9).

The results of this study mostly confirm what Teiber (2021:55) indicates about developers usually not seeing any benefits of using KPIs, but at the same time a new perspective has been explored where a developer sees the usefulness and benefit of KPIs. The perception of the Senior frontend developer aligns more with what Almeida and Carneiro (2023:11) explain about how years of experience working with Scrum is an important factor in the perception of the importance of KPIs. A possible explanation could be that the more experience a developer has, the more the individual perceives KPIs as important.

5.4 Measuring on individual level versus team level

While having the first probing dialogue with Team lead A at Extenda retail, a problem was identified, which was a lack of performance measurement of teams and individuals in teams (see Appendix 1). The Team lead A stated that both individual and team levels are important to follow how each individual is performing over time and to get an overview of how the whole team is performing as a unit. Although, to narrow down the focus of this study, a decision was made that a study focusing on the individual level was more relevant as there currently were no existing KPIs at all on individual level. This study resulted in mixed opinions on measuring at individual level.

The Team lead A at Extenda retail emphasizes that with individual KPIs, it's possible to spot issues, for example, if someone is doing more than they usually do because they are overburdened (see Table 9). The Team lead A believes that when individuals want to move towards a senior developer role, goals with KPIs can be set (see Table 9), and the Team lead A will be using the dashboard for one-on-one meetings with team members to discuss personal growth and setting quantifiable milestones (see Table 10).

Similarly, The Director of HR at Extenda Retail emphasizes that measuring performance at the team level allows for insights into the distribution of experience and performance levels within the team (see Table 8). The Director of HR at Extenda Retail means that a team of high performers might indicate a risk of members seeking advancement elsewhere, while a team with varied experience levels can support natural succession.

When it comes to individual performance, The Director of HR at Extenda Retail means that a strong team culture might mask individual performance dips, which the company would want to address to offer support so it does not impact other team members. Quoting the Director of HR, "But it can also lead to demoralization if others feel that "he/she always gets away with it, so why should we try?", creating a negative team culture." (see Table 8). The Director of HR further state that this case has occurred before, where strong collegial support covered for other, and it ended in unfortunate outcomes. This further reinforces the need for a visualization of individual performance, like the KPI dashboard created in this study (see Figures 3 & 4). The Director of HR confirms this by saying "This is just an example of why it is important to connect the individual with the team as a whole." (see Table 8). Despite the fact that the case firm has individual assessments twice a year for following up on individual's performance, The Director of HR states that the individual also requires a broader assessment, looking at what else is needed in their role for success, which may not be apparent in evaluations.

In contrast, the Engineering manager at Extenda Retail addresses that there is a sensitivity around individual KPIs and less so around team KPIs (see Table 6). The Engineering manager also emphasizes that highlighting negative aspects like bug counts can be insensitive and not helpful in fostering a creative culture, and team-based discussions are more valuable and motivating than focusing on individual performance, stating "I think it is even more important that the team is looking at itself, you know, because you are looking for self-driving teams." (See Table 6).

Similarly, the Software engineer and the Software developer think that if there are any issues, then those should be solved inside the team and not by some KPIs and prefers measurement at team level because measuring at an individual level will add pressure (see Table 7).

Team lead B states that factors like role differences (e.g., product owners) and collaborative efforts, such as knowledge sharing, can impact Velocity scores but are not reflected in the statistics (see Table 9). This highlights the need to consider team dynamics when measuring individual performance through KPIs.

While individual KPIs can provide insights into personal performance, there is a clear need for a balance between individual and team-based measurements. This study suggests that combining both could offer a more comprehensive understanding of performance, benefiting both the individual and the

team. What the Team lead, Engineering manager and Director of HR stated about how both individual- and team based KPIs are needed for a comprehensive view, aligns with research on hybrid models, where both individual and team rewards are used to encourage collaboration (Pearsall et al. 2010:188).

Cultural factors further complicate the application of individual KPIs. As noted by the Engineering manager at Extenda Retail, cultural differences affect how individual performance is captured (see table 7). An example is provided of people in the Netherlands, who, according to the Engineering manager, tend to place less emphasis on formal expectations, while in other cultures, clear expectations and performance measurements are considered more important. Likewise, Mendes et al. (2018:187) stated how in the Nordic countries, teams tend to emphasize collective success, whereas in India, hierarchical progression and individual prestige play a more prominent role. Meaning, these cultural variations influence how teams perceive themselves and their capabilities, making it essential to measure their impact on agile projects.

5.5 The impact of data quality on KPI reliability and actionability

One of the key findings of this study is that data quality is crucial for KPIs to be actionable. A major challenge has been the inconsistency and lack of structured data, particularly when measuring KPIs at the individual level (discussed in Chapter 3, Methods, section 3.5.3.3. Challenges). This aligns with Budacu and Pocatilu (2018:77), who highlight poor data quality as a common barrier to effective KPI adoption. Without reliable data, stakeholders struggle to extract meaningful insights, diminishing the KPIs' effectiveness.

The Engineering Manager at Extenda Retail described early difficulties in measuring effectiveness due to inconsistent Jira usage and unstructured data across teams (see Table 6). Similarly, a Senior Frontend Developer noted inaccuracies in the Velocity by Assignee KPI, caused by Jira issues not being properly marked as "Done." Such inconsistencies reduce trust in the dashboard and hinder its usability for decision-making. Boon et al. (2023:143) emphasize that low data quality undermines confidence in KPIs, whereas improving transparency and adherence to data logging standards enhances their effectiveness.

In this study, inconsistencies in Jira data usage emerged as a key factor affecting the reliability of certain KPIs. For example, the Cycle Time KPIs (see Figures 6 & 7) displayed misleading timing due to a configuration issue between Jira and EazyBI. These discrepancies illustrate how data inconsistencies can distort KPI interpretations, making them less actionable for teams. Variations in how tasks were tracked and completed further contributed to these issues, as different teams used Jira in different ways (see Table 9).

To address these challenges, standardizing Jira practices is essential. Team Lead A emphasized that for the dashboard to be fully effective, teams need to consistently update statuses and correctly utilize newly introduced fields (see Table 10). This supports Boon et al. (2023:142), who argue that high-quality data is a prerequisite for actionable KPIs. Despite these obstacles, the study suggests that applying KPIs can serve as a catalyst for process improvements. The dashboard was developed iteratively, incorporating feedback from three MVP tests (see Tables 9-11). This process has already led to better data collection practices and increased awareness among developers about the importance of maintaining accurate Jira entries, as noted by Team Lead A (see Table 9). These improvements demonstrate that while data quality remains a challenge, a structured approach to KPI adoption can drive meaningful enhancements in both processes and data integrity.

6 Conclusions

The main purpose of this bachelor's thesis in Information Systems is to identify and describe key performance indicators for measuring efficiency and effectiveness in an agile software development team from individual team members' perspectives.

The sub-purpose is to develop a KPI dashboard that measures individuals' performance based on data from a digital project management tool.

The two most important conclusions for the main purpose are presented below in sub-chapters 6.1 to 6.3, and for the sub-purpose, in sub-chapters 6.3 and 6.4, followed by contributions of this study and suggestions for future studies.

6.1 Deeper contextual analysis enhances the meaning of KPIs

The findings from this entrepreneurial study confirm that KPIs such as individual Velocity and Cycle time (both on individual and team level) and Change failure rate provide valuable insights into individual and team performance in agile software development processes. However, these KPIs work best if used in combination to obtain a comprehensive view of performance in terms of efficiency and effectiveness.

Velocity is useful for measuring work completed but does not account for activities such as assisting other team members or resolving other tasks outside of the assigned work. The Cycle time KPI proves to be effective to a greater extent in identifying workflow bottlenecks, and the Change failure rate KPI gives insights into the degree of effectiveness of the agile software development process.

One important finding from this entrepreneurial study is that analyzing multiple KPIs together, particularly individual Velocity and Cycle time on both individual and team levels, reveals process inefficiencies, such as bottlenecks caused by dependencies on other teams. These bottlenecks often arise from existing dependencies between teams that require input from other teams, which slows down the software development process.

Velocity, which tracks the amount of work completed over a certain period, and Cycle time, which measures the time taken to complete a task from start to finish, are both essential in understanding how smoothly work is progressing. However, by focusing solely on these metrics in isolation, it's easy to miss crucial factors such as the impact of team interdependencies.

This emphasizes the need for contextual analysis when using KPIs for measuring performance, consisting of considering team interdependencies and workload distribution amongst team members. For instance, if an individual team member's degree of Velocity appears lower, and a higher degree of dependency from the QA team is causing delays, it might not be an accurate reflection of that specific team member's performance. By looking at the Cycle time KPI, these dependencies are detected.

Additionally, measuring individual Cycle time proves to be a highly effective self-evaluation KPI for developers, one aspect that has not been identified in earlier studies.

6.2 Measuring individual performance offers both perceived opportunities and risks

The findings of this study contribute to the ongoing discussion on agile performance measurement by demonstrating how individual-level KPIs are applied in practice. The demonstration provides insights into measuring individual performance in agile software development, which there is a lack of in earlier studies.

A key observation is that developers hold mixed views on individual performance measurement. While some perceive value being created from measuring personal productivity to identify areas for growth, others express concerns about potential misuse, such as fostering competition instead of collaboration. The findings in this study suggest that while some individuals appreciate a higher degree of performance transparency, others worry that individual performance measurement could create unnecessary higher stress of being over watched.

The preference of the SPIN interviewees appears to lean toward a balanced approach, combining team level and individual level KPIs to provide a more comprehensive view of the performance of a team.

6.3 The SPACE framework works in theory, but needs adaptations in practice

To the best of my knowledge, this study is the first known attempt to apply the SPACE framework for developing a KPI dashboard. In addition, no prior studies have attempted to leverage automatically collected data from a digital project management tool for the purpose of developing a KPI dashboard with the SPACE framework as the foundation.

The results of this study show that applying the SPACE framework in practice poses challenges due to limitations in the degree of data availability and quality. A key takeaway is that while the SPACE framework provides a good theoretical foundation for selecting KPIs, applying it in practice, especially with automated data collection, requires modifications such as including more objective KPIs.

The SPACE framework can not be applied as it is when leveraging automatically collected data due to the challenges of capturing subjective KPIs in, for example, the dimension Satisfaction and well-being dimension in the SPACE framework. The SPACE framework emphasises including subjective KPIs, such as well-being and collaboration. These KPIs prove difficult to quantify using the available data sources, since subjective data are not automatically collected in the digital project management tool.

This conclusion sub-chapter contributes to both the main and the sub-purpose, as it provides insights into applying the SPACE framework in practice, supporting the main purpose. Additionally, the challenges with the SPACE framework and adaptations that were needed to develop the KPI dashboard are addressed above, which aligns with the sub-purpose of creating the KPI dashboard.

6.4 The degree of data quality is paramount for KPIs to be more actionable

The effectiveness of KPIs relies heavily on the degree of quality of the underlying data. A key challenge identified in this study is the lower degree of inconsistency in Jira data usage, which directly impacts the degree of reliability of certain KPIs. However, despite these challenges, the introduction of KPIs accelerates process improvements. Following the entrepreneurial approach, I developed the KPI dashboard iteratively, incorporating feedback from three MVP tests. This feedback process is already a practical contribution to enhanced data collection practices at Extenda Retail and increases the degree of awareness among team members regarding the importance of a higher degree of accurate Jira data entries. Acknowledging the degree of inconsistencies and gaps of data logging encourages more precise reporting, which in turn enhances the reliability of performance measurements.

Ensuring a higher degree of data accuracy remains a critical factor in attaining the effectiveness of KPI reports. For instance, the accuracy of the Cycle time KPI depends on how consistently team members log work in the project management tool Jira. Differences in workflow practices, such as marking Jira issues as "Deployed" instead of "Done", necessitate adjustments in KPI calculations to ensure more meaningful measurements. Without standardized data logging practices, the reliability degree of performance insights is compromised.

In this sense, where data accuracy and standardized logging practices are crucial for reliable KPI measurements, this study highlights that a standardised KPI framework is difficult to achieve. It is difficult to achieve due to the fact that software development processes and data practices vary significantly between teams. Instead, organizations should select KPIs based on their specific processes and ensure that data collection practices support more meaningful and reliable measurement of performance.

6.5 Contributions of the study and future studies

One knowledge contribution from this study is that it provides a broader discussion on performance measurement in agile teams by demonstrating identified KPIs on the individual level.

Additionally, during this study I developed the presented KPI dashboard for measuring individual and team performance, which is in use since end of December 2024 for the co-workers at Extenda Retail. This KPI dashboard demonstrates how individual performance can be measured in practice, bridging the gap between theory and practice. While much of the existing literature focuses on KPIs on a team level, this study provides empirical insights into the possibilities and hindrances of measuring individual performance. Additionally, the findings from this study clearly indicate the importance of a higher degree of data quality.

This study primarily contributes practical knowledge to the case firm by providing insights into which individual-level KPIs exist in literature and the description of the identified KPIs, as well as what individual-level KPIs I recommend the specific team at the firm that I worked closely with to use. It also offers valuable contributions to information systems students, practitioners, and companies seeking a deeper understanding of individual performance measurement in agile teams.

From a methodological perspective, the study is conducted with an entrepreneurial approach, which has been a beneficial approach for this study. The KPI dashboard was developed through iterations with the key stakeholders, resulting in a tangible, visual archetype. The entrepreneurial approach proves to be more effective in refining the KPI dashboard. This study serves as an important knowledge contribution for future similar entrepreneurial studies.

Future studies are recommended to explore how subjective factors, such as developer satisfaction, psychological safety, and collaboration, to be integrated into automated performance measurement tools. While quantitative KPIs provide valuable insights, they do not fully capture the complexities of software development work, where creativity, teamwork, and problem-solving play critical roles.

Further studies should also examine perspectives from individual team members in greater depth, ideally involving a larger sample of developers, team leads, and managers from various firms and also from different countries.

Additionally, future studies should analyze how the long-term effects of individual level performance measurements within agile software development teams impact the degree of collaboration, motivation, and overall team dynamics over time.

7 Recommendations

To effectively adopt the KPIs dashboard developed in this study, Team leads or equivalents at Extenda Retail should begin by analyzing their software development processes and see how the KPIs dashboard can be applicable to their specific team. Since this study concludes that a standardized framework for measuring performance is more difficult to achieve, teams most likely need to adjust the KPI reports so they match the team's workflow more.

The first step is to determine whether the existing calculations of the KPI reports in the dashboard to a greater extent accurately reflect the team's agile software development process. If needed, teams should either adjust the KPI report calculations in EazyBI or align the software development process with better data logging practices for more effective measurement in the KPI dashboard. Between these approaches, adjusting the software development process is more preferable, as it contributes to a higher degree of standardized workflows across the company, making KPI measuring more consistent across teams. One concrete adjustment is for teams to add more statuses in the Jira project boards for the Cycle time KPIs to show higher degree of accurate and detailed measures. For example, instead of using only the statuses "Unassigned," "In Progress," and "Done," teams may benefit from introducing statuses like "In QA" and "In Review" to better capture the different phases of the development cycle.

Additionally, to maintain accurate Velocity KPIs measures, it is essential that all Jira issues are properly closed once completed, ideally with the status "Done" in order to use the developed Velocity KPI reports that are developed in this study. Without this step, the Velocity KPIs may be more misleading and fail to reflect actual individual performance.

This brings us to the second recommendation, which is that Team leads should guide their teams on why and how to input data more correctly. One key practice includes ensuring that Jira issues move through all necessary statuses at the right time rather than skipping steps. This is particularly important for the Cycle time KPIs, where missing transitions can lead to more inaccurate measures.

Once teams have adapted their workflows and improved their data logging practices, I recommend teams to gradually expand their KPI dashboard to incorporate additional dimensions from the SPACE framework. This leads to a more holistic performance analysis by integrating more factors.

A key challenge in this study was the lack of automatically collected data available for measuring KPIs in EazyBI, especially at the individual level. Teams looking to adopt additional KPIs should first select suitable ones from Table 1 that I created in Chapter 2, which summarizes individual-level KPIs. Then, they should determine what data needs to be logged and how this will affect the software development process in order to make a plan before developing the new KPI report in EazyBI or another data analysis tool. Analyzing the existing data and planning the development before developing the KPIs will save a lot of time that may go to waste if there turns out to be data availability issues later on.

List of references

Written sources

Al-Heyasi, A. (2018). Individuals Performance Measurement in Agile Software Development. *Eurasian Journal of Social Sciences*, 6(1), pp.1-6.

Almeida, F., & Carneiro, P. (2023). Perceived importance of KPIs for agile Scrum environments. *Information*, 14(6), 327.

Armbruster Santiago, E. (2015). *First Pass Yield KPI Implementation for Validation Documents*. Master's thesis, Engineering in Manufacturing Engineering. Polytechnic University of Puerto Rico. https://prcrepository.org/xmlui/bitstream/handle/20.500.12475/613/Articulo%20Final_Eric%20Armbruster.pdf?sequence=1&isAllowed=y

Atlassian (2025a). *Great outcomes start with Jira*. <https://www.atlassian.com/software/jira> [2025-01-04].

Atlassian (2025b). *Build dashboards to analyze data*. <https://support.atlassian.com/analytics/docs/build-dashboards-to-analyze-data/> [2025-01-22].

Atlassian (2025c). *Jira issues overview*. <https://www.atlassian.com/software/jira/guides/issues/overview#what-is-the-anatomy-of-an-issue> [2025-01-22].

Barnes, C., Hollenbeck, J., Jundt, D., DeRue, D. & Harmon, S. (2011). Mixing individual incentives and group incentives: Best of both worlds or social dilemma? *Journal of Management*, 37(6), pp. 1611-1635.

Beck, K., Beedle, M., van Bennekum, A., 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. & Thomas, D. (2001). The Agile Manifesto. *Agile Alliance*. <https://agilemanifesto.org/iso/sv/manifesto.html> [2024-10-01]

Beh, H.C., Jusoh, Y.Y., Abdullah, R. & Hassan, S. (2022). Dimensions in Measuring Performance of Agile Software Development Projects: A Literature Review. In *2022 Applied Informatics International Conference (AiIC)*, Serdang, Malaysia 18-19 May 2022, pp. 83-87.

Boon, G.C., Stettina, C.J., Visser, J. & El-Baz, Y. (2023). Beyond Dashboards: Operationalising a Measurement Framework for Agile Teams. *Springer*, 130-146.

Budacu, E. & Pocatilu, P. (2018). Real Time Agile KPIs for Measuring Team Performance. *Informatica Economica*, 22(4), 70-79.

Carlgren, L. & BenMahmoud-Jouini, S. (2022). When cultures collide : What can we learn from frictions in the implementation of design thinking? *The Journal of Product Innovation Management*, 39(1), pp. 44-65.

CIO (2023). *Is it worth measuring software developer productivity? CIOs weigh in*. <https://www.cio.com/article/1255774/is-it-worth-measuring-software-developer-productivity-cios-weigh-in.html> [2024-10-10].

Davis, CWH. (2015). *Agile KPIs in Action - How to Measure and Improve Team Performance*. Manning. Available: <https://www.perlego.com/book/2682475>

De Ruyck, B., Quataert, S., Vandenbroucke, A., Van Steerthem, A., Baeten, X. & Dewettinck, K. (2020). Performance and Reward Management in an Agile Environment. *Vlerick Business School*.

Dynehäll, M. S. & Ståhlberg Lärk, A. (2015). *LOOPA: A Business Development Method for Entrepreneurs*. Roos & Tegner. www.roostegner.se

Dynehäll, M. S. & Ståhlberg Lärk, A. (2014). *Loopa: affärsutveckling för entreprenörer: så driver du din affärsidé till kundsuccé*. Stockholm: Liber.

EazyBI. (2025a). *The Leading Jira Reporting App*.
<https://EazyBI.com/products/EazyBI-reports-and-charts-for-jira> [2025-01-02].

EazyBI. (2025b). *EazyBI Features*.
<https://EazyBI.com/features> [2025-01-02].

Ertaban, C., Sarikaya, E., & Bagriyanik, S. (2018). Agile performance indicators for team performance evaluation in a corporate environment. In *Proceedings of the 19th International Conference on Agile Software Development: Companion (XP '18)*. New York, NY, USA, pp. 1-3.

Extenda retail 2025. *About us*.
https://www.extendaretail.com/about-us/?utm_source=chatgpt.com [2025-02-14].

Extenda retail 2025. *Solutions*.
<https://www.extendaretail.com/solutions/> [2025-02-14].

Forsgren, N., Storey, M.-A., Maddila, C., Zimmermann, T., Houck, B., and Butler, J. (2021). The SPACE of Developer effectiveness. *ACMQueue*, 19(1), pp. 20-48.

Gamble, R.F. & Hale, M.W. (2013). Assessing individual performance in Agile Undergraduate Software Engineering Teams. In *Proceedings - Frontiers in Education Conference*. Oklahoma City, OK, USA, 23-26 October 2013, pp. 1678-1684.

Huss, M., Herber, D.R. & Borcky, J.M. (2023). Comparing Measured Agile Software Development KPIs Using an Agile Model-Based Software Engineering Approach versus Scrum Only. *Software*, 2(3), pp. 310-331.

López, L., Burgués, X., Martínez-Fernández, S., Vollmer, A.M., Behutiye, W., Karhapää, P., Franch, X., Rodríguez, P. and Oivo, M. (2021). Quality measurement in agile and rapid software development: A systematic mapping. *Journal of Systems and Software*, 186(5), pp. 111187.

Machuca-Villegas, L., Gasca-Hurtado, G.P. & Muñoz, M. (2021). Measures related to social and human factors that influence effectiveness in software development teams. *International Journal of Information Systems and Project Management*, 9(3), 43-67.

McKinsey (2023). *Yes, you can measure software developer effectiveness*.
<https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/yes-you-can-measure-software-developer-effectiveness>. [2024-10-10].

Mendes, E., Viana, D., Datta Vishnubhotla, S. and Lundberg, L. (2018). Realising individual and team capability in agile software development: A qualitative investigation. In *44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. Prague, Czech Republic 29-31 August 2018, pp. 183-190.

- Menezes, R., Marinho, M.L.M., & Sampaio, S. (2024). KPIs in Large-Scale Agile Software Development: A Multivocal Literature Review. In *Proceedings of the 27th Ibero-American Conference on Software Engineering*. Curitiba, PR, Brasil 6 May 2024, pp. 106-120.
- Mason, M. (2010). Sample Size and Saturation in PhD Studies Using Qualitative Interviews [63 paragraphs]. *Forum Qualitative Sozialforschung / Forum: Qualitative Social Research*, 11(3), Art. 8.
- Natarajan, T. & Pichai, S. (2024). Behaviour-driven development and KPIs framework for enhanced agile practices in Scrum teams. *Information and Software Technology*. 170.
- Oliveira, E., Fernandes, E., Steinmacher, I., Cristo, M., Conte, T., & Garcia, A. (2020). Code and commit KPIs of developer effectiveness: A study on team leaders' perceptions. *Empirical Software Engineering*, 25(4), 2519-2549.
- Pearsall, M., Christian, M., & Ellis, A. (2010). Motivating interdependent teams: Individual rewards, shared rewards, or something in between? *Journal of Applied Psychology*, 95(1), pp. 183-191.
- Pham, K.P. & Neumann, M. (2024). How to measure performance in agile software development? A mixed-method study. *ArXiv*.
- Pickard, A. J. (2017). *Research Methods in Information*. 2nd edn. Facet Publishing. Available: <https://www.perlego.com/book/3259640>
- Pinter, R., Čisar, S.M. & Čisar, P. (2017). Measuring team member performance in Scrum - Case study. In *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*. Subotica, Serbia 14-16 September 2017, pp. 309-314.
- Power, K. (2014). KPIs for Understanding Flow. *Agile Alliance*.
- Psarov, O. & Druzhinin, E. (2024). Enhancing Agile team productivity with metrics. *Scientific Journal of the Ternopil National Technical University*, 113(1), pp. 93-99.
- Rainer, R.K. & Prince, B. (2021). *Introduction to Information Systems*. Ninth edition, John Wiley Sons, Hoboken, New Jersey.
- Ramírez-Mora, S.L. & Oktaba, H. (2017). effectiveness in Agile Software Development: A Systematic Mapping Study. In *2017 5th International Conference in Software Engineering Research and Innovation (CONISOFT)*. Merida, Mexico 25-27 October 2017, pp. 44-53.
- Ramírez-Mora, S.L., Oktaba, H., & Patlán Pérez, J. (2019). Group Maturity, Team Efficiency, and Team Effectiveness in Software Development: A Case Study in a CMMI-DEV Level 5 Organization. *Journal of Software: Evolution and Process*, 32(4), e2232.
- Razzouk, R. & Shute, V. (2012). What Is Design Thinking and Why Is It Important? *Review of Educational Research*, 82(3), pp. 330-348.
- Ries, E. (2011). *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. New York: Crown Currency.
- Riihimäki, R. (2024). *effectiveness KPIs and their integration into DevOps*. Master's thesis. Software Engineering. University of Turku. https://www.utupub.fi/bitstream/handle/10024/178759/Riihimaki_Rasmus_Thesis.pdf;jsessionid=62E269D849A23EF78B8DAF7FB5A1304F?sequence=1

Robson, C. (2014) *How to do a Research Project*. (2nd ed.). Wiley. Available: <https://www.perlego.com/book/3866009>

Rüegger, J., Kropp, M., Graf, S. & Anslow, C. (2024). Fully Automated DORA KPIs Measurement for Continuous Improvement. In *ICSSP '24: Proceedings of the 2024 International Conference on Software and Systems Processes*. New York, NY, USA 4-6 September 2024, pp. 36-45.

Sallin, M., Kropp, M., Anslow, C., Quilty, J.W. & Meier, A. (2021). Measuring Software Delivery Performance Using the Four Key KPIs of DevOps. In *Proceedings of the 22nd International Conference on Agile Software Development (XP 2021)*. Trondheim, Norway 14-18 June 2021, pp. 103-119.

Shah, S.M.A., Papatheocharous, E. & Nyfjord, J. (2015). Measuring productivity in agile software development process: a scoping study. In *Proceedings of the 2015 International Conference on Software and System Process (ICSSP '15)*. Tallinn, Estonia 24-26 August 2015, pp. 102–106.

Staaake, F. (2022). *En entreprenöriell studie av den interna försäljningsprocessen i ett stålverk: en fallstudie inom Björneborg Steel*. Kandidatuppsats i Informatik. Handelshögskolan vid Karlstads Universitet.

Swedish Authority for Privacy Protection (2021). *The GDPR fundamental principles*. <https://www.imy.se/en/organisations/data-protection/this-applies-according-to-gdpr/the-gdpr-fundamental-principles/> [2025-02-13].

Teiber, N. (2021). *Goals and KPIs in Large-scale Agile Development: A Systematic Literature Review*. Master's thesis, Information Systems. School of CIT, Technical University of Munich. <https://www.matthes.in.tum.de/pages/1gkpczw297xtt/Masterarbeit-Nadine-Teiber>

Visser, T. & Hulstijn, J. (2023). Measuring Agile/DevOps team performance. *Proceedings of the Second International Workshop on Agile Methods for Information Systems Engineering (Agil-ISE 2023)*. Zaragoza, Spain, 13 June, pp. 17-23.

Whiteley, A., Pollack, J. & Matous, P. (2021). The origins of agile and iterative methods. *The Journal of Modern Project Management*, 8(3), pp. 20-29.

Wilkes, B., Milani, A.M.P. & Storey, M.A. (2023). A Framework for Automating the Measurement of DevOps Research and Assessment (DORA) Metrics. In *2023 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. Bogotá, Colombia 01-06 October 2023, pp. 62-72.

Ziegler, A., Kalliamvakou, E., Simister, S., Sittampalam, G., Li, X. A., Rice, A., Rifkin, D. & Aftandilian, E. (2022). effectiveness Assessment of Neural Code Completion. In *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming (MAPS '22)*, San Diego, CA, USA 13 June 2022, pp. 21-29.

Oral sources

Team lead A Extenda Retail, Stockholm. Probin dialog 2024-09-04, 2024-10-02.

Engineering Manager at Extenda Retail, Google Meet. SPIN-Interview 2024-10-18, 2024-11-22.

Software Developer at Extenda Retail, Google Meet. SPIN-Interview 2024-10-21.

Software Engineer at Extenda Retail, Google Meet. SPIN-Interview 2024-10-21.

Director of Human Resources at Extenda Retail, Google Meet. SPIN-Interview 2024-10-22.

Team lead A at Extenda Retail, Google Meet. SPIN-Interview 2024-10-22.

Senior Frontend Developer at Extenda Retail, Google Meet. MVP 1 Test 2024-12-27, MVP 2 Test 2025-01-03, MVP 3 Test 2025-01-10.

Software Engineer at Extenda Retail, Google Meet. MVP 1 Test 2024-12-30, MVP 2 Test 2025-01-03, MVP 3 Test 2025-01-10.

Team lead B at Extenda Retail, Google Meet. MVP 1 Test 2024-12-27, MVP 2 Test 2025-01-03, MVP 3 Test 2025-01-10.

Engineering Manager at Extenda Retail, Google Meet. MVP 1 Test 2024-12-27, MVP 2 Test 2025-01-06, MVP 3 Test 2025-01-10.

Team lead A at Extenda Retail, On site & Google Meet. MVP 1 Test 2024-12-27, MVP 2 Test 2025-01-08, MVP 3 Test 2025-01-10.

Appendices

Appendix 1: Probing dialogue 1

Date: 04-09-2024

Duration: 20 min

Location: On site

Participants: Laura-Liisa Lillemäe (Student) & Team lead

Team lead:

What ideas do you have for your thesis work, what do you want to write about?

Me:

My idea has been to create a machine learning model that can predict how successful a project will be, based on previous project data from, for example, Jira and retrospectives. I am open to your suggestions and any changes to the theme to better suit your company's needs.

Team lead:

Your idea is very interesting. However, I think it can be difficult to access the amount and type of data required to build a machine learning model. Another area where we've seen a need is in measuring the effectiveness of our team members. At present, we do not measure effectiveness in any way at all.

My suggestion is that you instead develop a tool that can be integrated into Jira dashboards, where we can visualize effectiveness data with the help of JQL. I would like to have a solution that clearly demonstrates effectiveness at both the individual and team levels. For example, I want to be able to identify when an employee's performance deviates from normal, so that we can have a dialogue and understand what is behind it. In addition, it is important to be able to analyze why effectiveness is falling and what we can do to fix the problems.

Me:

Is the idea that the tool will be tested on your team?

Team lead:

Yes, that's right. We would use it in my team to see how well it works and if it can be beneficial to our workflows.

Me:

How do you view the issue of measurement at the individual level and team level? Which level is most interesting to focus on?

Team lead:

Both individual and team level are important. I want to be able to follow how each individual is doing over time, but also get an overview of how the whole team is performing as a unit. It would be good if there were different views where you can switch between individual level and team level for a more complete picture of effectiveness.

Me:

It may be that I have to choose either whether it should be at the individual level or at the team level. If so, what do you think you would most like to get out of these two?

Team lead:

I think individual level in such case, because we do not have any metrics there. We already have two metrics on team level, Velocity and Burndown chart.

Appendix 2: Probing dialogue 2

Date: 02-10-2024

Duration: 20 min

Location: On site

Participants: Laura-Liisa Lillemäe (Student) & Team lead

Me:

On a previous occasion, we have mainly discussed effectiveness, i.e. efficiency, in relation to the work of the team and the individual. However, I would like to examine why we have not included effectiveness, that is, effectiveness, in our analyses. How do you view this?

Team lead:

That's a good question. In fact, on reflection, we can say that we're indirectly looking at both effectiveness and effectiveness through some of the key KPIs we're considering, even if we haven't articulated it explicitly before.

Me:

In order to collect more specific and relevant data, I plan to conduct SPIN interviews as part of my work. The aim is to better understand what is most important to measure, both in terms of effectiveness and effectiveness. I would need more interviewees to get a broader perspective. Do you have any suggestions on which people might be suitable to interview?

Team lead:

Yes, I think HR, our teams manager and developers from the team could help. I can help you get access to these people.

Me:

Thank you very much, I think these perspectives can add a lot.

Me:

I've gone through a lot of research papers and noticed that most studies focus on the team level when it comes to measuring effectiveness. Unfortunately, there is very limited material that addresses the individual level. However, the most common key KPIs mentioned in the literature are velocity, cycle time and lines of code, but I feel that lines of code is not a good measure because it can drive negative behaviors, such as writing redundant code just to increase one's measurable output, I don't think we run on that.

Team lead:

I agree. Measuring lines of code can definitely skew effectiveness and can be counterproductive. Instead, I think we can look at how often a developer makes commits per day. Frequent commits indicate that the person is active and moving forward. If a developer is making fewer commits than usual, it can signal that the person is stuck or running into problems.

Me:

Good point!

Team lead:

I also think that the Change failure rate (CFR) would have been a good KPI. CFR measures how often code changes lead to errors or failures in production, and what code changes caused it. While it's mainly useful at the team level, I think we can also gain some insights at the individual level.

Team lead:

Using tools and key KPIs such as commit frequency and CFR, we can get a better picture of when a developer's work deviates from normal. This would allow us to quickly identify any problems such as lack of motivation or technical barriers. By having this information available, we can bring it up in our regular one-on-one meetings and work on solving the problems together. We have such meetings twice a week for most developers, but the frequency can vary depending on their needs.

Me:

It sounds like a good setup. By picking up on these signals early, you can offer support and solutions in a timely manner, improving both individual and team effectiveness.

Me:

During the first exploratory talk, you said that you hope that the tool will be able to be used in your workflows, plural. I want to double-check which workflows you had in mind, ideally, I'll start by focusing on one.

Team lead:

Well, I meant the software development process and that at a later stage it might be applicable in other development teams as well.

Me:

Okay so the workflow that you wish I focus on is the software development process?

Team lead:

Yes, exactly.

Appendix 3: Brainstorming with Team Lead

Date: 02-10-2024

Duration: 20 min

Location: On site

Participants: Laura-Liisa Lillemäe (Student) & Team lead

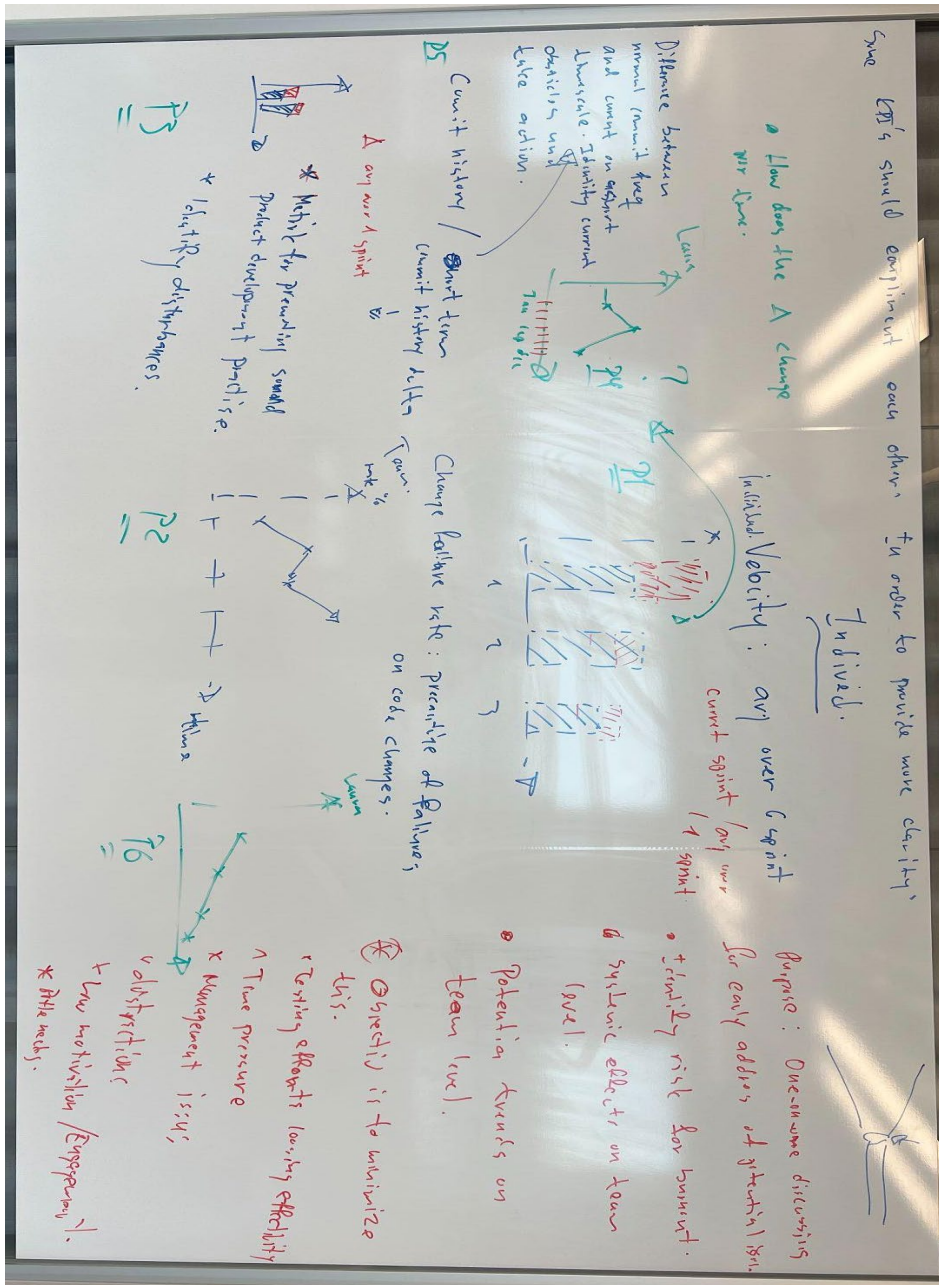


Figure 14: The results of brainstorming, with suitable KPIs and the purpose of them. Source: The author and a Team lead Bt Extenda retail.

In Figure 14, the results of brainstorming together with the team lead, whose team I initially built the dashboard for, on suitable KPIs for measuring individual performance and descriptions of purposes of the KPIs are presented. They were also categorized by importance, presented with values P1-P6, with the “P” standing for “priority”.

The brainstorming session was spontaneously initialized after probing dialogue 2 (see Appendix 2).

Appendix 4: SPIN-Interview guide for Engineering Manager/Team Lead

Before the interview, provide information on: information letter, consent form and recording of the interview.

1. Situational questions

- 1.1. How are you currently working with measuring performance in terms of efficiency within development teams?
- 1.2. How are you currently working with measuring performance in terms of effectiveness within development teams?
- 1.3. How are you currently measuring the performance of separate individuals in teams?
- 1.4. How often are follow-ups made regarding the teams' performance?
 - 1.4.1. What methods do you use to follow up the work of a team?
 - 1.4.2. What methods do you use to follow up the work of separate individuals in teams?
- 1.5. How well do the current practices capture the performance of a team, in your opinion?
- 1.6. How well do the current practices capture the performance of separate individuals in teams, in your opinion?

2. Problem questions

- 2.1. What problems do you experience in getting an overall picture of how productive the teams in your department are?
 - 2.1.1. What specific data are you missing today?
- 2.2. Have you experienced problems accurately identifying when a team is losing effectiveness?
 - 2.2.1. What causes the problem?
- 2.3. What challenges have you encountered when measuring effectiveness within teams?
- 2.4. What challenges have you encountered when measuring the performance of separate individuals in a team?
- 2.5. What would you like to improve in the current process for measuring performance?
- 2.6. Do you see any problems with measuring performance at an individual level versus the team level?

3. Implication questions

- 3.1. What do you think are the consequences of not having a clear method for measuring performance?
- 3.2. What effects do you think an improved method of measuring efficiency would have?
- 3.3. What effects do you think an improved method of measuring effectiveness would have?

4. Need-payoff questions

- 4.1. Do you think that both aspects, effectiveness, and efficiency, should be measured to the same extent?
 - 4.1.1. Why do you think so?
- 4.2. How would a tool that measures performance improve the way you manage projects?
- 4.3. What types of information would help you make better decisions about team performance?
- 4.4. How do you think that an improved measurement of effectiveness affects the team's long-term growth?
- 4.5. What would you like to see in a tool that provides a more detailed picture of the performance of a team?

Appendix 5: SPIN-Interview guide for Developer

Before the interview, provide information on: information letter, consent form and recording of the interview.

1. Situational questions

- 1.1. How are you currently working with measuring performance in terms of efficiency within development teams?
- 1.2. How are you currently working with measuring performance in terms of effectiveness within development teams?
- 1.3. How often are follow-ups made regarding the teams' performance?
 - 1.3.1 What methods do you use to follow up the work of the team?
- 1.4. How well do the current practices capture the performance of a team, in your opinion?
- 1.5. How well do the current practices capture the performance of separate individuals in teams, in your opinion?

2. Problem questions

- 2.1. What challenges have you encountered when measuring performance within teams?
- 2.2. What challenges have you encountered when measuring the performance of separate individuals in a team?
- 2.3. What would you like to improve in the current process for measuring performance?
- 2.4. Do you see any problems with measuring performance at an individual level versus a team level?
 - 2.4.1. In what ways is the work being affected by measuring performance at an individual level, in your opinion?

3. Implication questions

- 3.1. What do you think are the consequences of not having a clear method for measuring performance?
- 3.2. What effects do you think an improved method of measuring efficiency would have?
- 3.3. What effects do you think an improved method of measuring effectiveness would have?

4. Need-payoff questions

- 4.1. Do you think that both aspects, effectiveness, and efficiency, should be measured to the same extent?
 - 4.1.1. Why do you think so?
- 4.2. Specifically, in what way, do you think such tools help prevent problems, such as effectiveness dip?
- 4.3. How do you think that an improved measurement of effectiveness affects the team's long-term growth?
- 4.4. What would you like to see in a tool that provides a more detailed picture of the performance of a team?

Appendix 6: SPIN-Interview guide for Director of Human Resources (HR)

Informera om innan intervjun: informationsbrev, samtyckesblankett och inspelning av intervju.

1. Situationsfrågor

- 1.1. Hur arbetar organisationen för närvarande med att mäta prestation hos mjukvaruutvecklingsteam?
- 1.2. Hur ofta genomförs uppföljningar av teamens prestation?
 - 1.2.1. Hur används dessa resultat?
- 1.3. Hur ofta genomförs uppföljningar av enskilda individers prestation?
 - 1.3.1. Hur används dessa resultat?
- 1.4. Hur väl uppfattar du att de nuvarande metoderna fångar upp mjukvaruutvecklarnas prestation på ett rättvist sätt?
 - 1.4.1. Fångar dessa metoder upp prestation på ett tillförlitligt sätt, enligt din mening?

2. Problemfrågor

- 2.1. Vilka problem har ni stött på när ni har mätt prestation hos team?
- 2.2. Vilka problem har ni stött på när ni har mätt prestation hos enskilda individer i team?
- 2.3. Vilka etiska risker finns det med att mäta prestation på individnivå hos team, enligt din mening?
 - 2.3.1. Hur skulle dessa kunna hanteras på ett hållbart sätt?

3. Inverkansfrågor

- 3.1. Vad anser du är konsekvenserna av att inte ha en tydlig metod för att mäta prestation i form av ändamålsenlighet?
- 3.2. Vad anser du är konsekvenserna av att inte ha en tydlig metod för att mäta prestation i form av produktivitet?
- 3.3. Vilka effekter tror du att ett verktyg för att mäta prestation har på arbetsmiljön?
- 3.4. Vilka effekter tror du att ett verktyg för att mäta prestation har på personalens välbefinnande?

4. Nyttifrågor

- 4.1. Vilka är dina uppfattningar om att mäta prestation på individnivå?
 - 4.1.1. Varför anser du så?
- 4.2. Vilka metoder skulle möjliggöra en bra balans mellan behovet av att mäta enskilda individers prestationer och de etiska överväganden, enligt din mening?
- 4.3. Vilka egenskaper skulle du vilja se i ett verktyg som hjälper till att mäta prestationer på ett sätt som tar hänsyn till enskilda individers bidrag?

Appendix 7: Protocol of SPIN-Interview with Engineering Manager

Informed before the interview: information letter, consent form, and recording of the interview.

Date: 18-10-2024, 22-11-2024

Duration: 25 min, 29 min

Location: Google Meet

Participants: Engineering manager & Laura-Liisa Lillemäe (Student)

1. Situational questions

1.1. How are you currently working with measuring performance in terms of efficiency within development teams?

Talking about measuring efficiency, today, there is no structured approach. It is based on observation and feedback. Based on observation, from a certain understanding of what you expect from a developer. I have more or less a definition for that. I can share that with you if you like. And also based on feedback, it is feedback from the person itself but also, in the end, feedback from team-members. So you can say it is a reactive observation.

1.2. How are you currently working with measuring performance in terms of effectiveness within development teams?

Just to talk about the definition of efficiency and effectiveness, I see effectiveness as achieving a goal and efficiency in how you achieved your goal. Both are definitely observed and just zooming in to one example, basically when a new software product or an upgrade on a software product is delivered, then there are certain inputs. We call this the input requirements and the acceptance criteria. At the end of the delivery, where the requirements are discussed, the development is done, the testing is done etc, you are looking into a product and say, okay, what did we discuss at the beginning and what do we discuss at the end. Also how many bugs and flaws have been introduced etc. That is definitely observed. But also the efficiency and one of the methods when you observe efficiency is to say, okay, the team or the individual has a certain estimation of the amount of work. So let's say one week. If it then takes three weeks, then we have an efficiency problem or problem in the predictability of the efficiency. So there are things in that sense when you are looking into a team, they are working according to Agile and Scrum, so you see that they are using a method where they discuss the work they have to do: requirements and acceptance criteria. Then they perform an estimation. We are working with points, points VS time estimation. Then they actually do the work and they deliver. All that work is based on, or not all the work, but there is some kind of way of working in Scrum and the things you apply from Scrum, where you look into the effectiveness of the team. Not only from a management perspective, I think it is even more important that the team is looking at itself, you know, because you are looking for self-driving teams. So they use these estimations and look into the amount of bugs etc, and find a way for optimization. But I think having KPIs would be fantastic.

1.3. How are you currently measuring the performance of separate individuals in teams?

When it comes to the individual, there you are definitely observing. But yet again it is a bit of an unstructured approach. You try to onboard a certain person when he or she joins the team and then you have your observations. Based on the experience from different people in the team and also for myself and from Team Lead you understand how it is going. But again, there are no defined KPIs currently. But you could definitely also look into the team. Because it needs to work as a team. Also there is no structured approach trying to measure the team as a whole. That's why I'm also happy to have this conversation today. But you would look into how they are collaborating and how the work is prepared, what the quality of the work is, etc. I would like to add something else there as well because yes, you are looking into an individual, yes you are looking into a team, but you are also looking for cross-team collaboration. That is much more difficult to identify, in the end, if you have a couple of good functioning teams, but they are not collaborating well cross-team. So, we are trying to look at three levels. And then I look to managers and team leads, most of them more or less have their own approach, which I believe is structured.

1.4. How often are follow-ups made regarding the teams' performance?

Currently, HR assessments focus on individuals rather than the team as a whole. I use these assessments as a historical starting point to initiate discussions within the team. These discussions aim to clarify what is expected of each role, what individuals expect from their roles, and to establish a baseline for performance and communication. My goal is to ensure that team members talk to each other about their expectations and align on a shared understanding.

1.4.1. What methods do you use to follow up the work of a team?

None at all, at least not formal practices.

1.4.2. What methods do you use to follow up the work of separate individuals in teams?

It is the HR assessments.

1.5. How well do the current practices capture the performance of a team, in your opinion?

the current practices do not effectively capture team performance, as there are no formal processes in place to measure it. While there is some attention given to individual performance, there is no structured way to assess or optimize team-level collaboration. Teams are expected to be somewhat self-organizing, following frameworks like Scrum, which includes retrospectives for process optimization. However, these efforts often lack the intent and depth needed to address team-level challenges effectively. To address cross-team collaboration, we have introduced PI (Program Increment) planning from the SAFe framework. This occurs every 12 weeks and involves teams working on newer products. During PI planning, we set business, product, and technical goals, identify dependencies, and create a high-level plan. Weekly follow-ups on progress and blockers are held every Thursday to ensure alignment. While this provides some structure, it focuses more on coordination rather than measuring or improving collaboration within and between teams. Overall, there is an acknowledgment that the current approach is not working effectively, but the lack of KPIs makes it difficult to identify and address these issues systematically.

1.6. How well do the current practices capture the performance of separate individuals in teams, in your opinion?

It depends on how focused the organization is on its processes and how individuals approach their roles and responsibilities. In our organization, we conduct an annual survey, and the feedback has highlighted that expectation management is lacking. People feel there isn't enough clarity about what is expected of them or how their performance is measured. This gap in expectation management impacts individual performance evaluation. Cultural differences also play a role. For instance, in the Netherlands, people generally place less emphasis on formal expectations, while in other cultures, clear expectations and performance measurements are considered very important. This variation further influences how well individual performance is captured across teams.

2. Problem questions

2.1. What problems do you experience in getting an overall picture of how productive the teams in your department are?

I think it will be very insightful for you if you just join a couple of meetings and observe. If someone explains to you a bit, and I will prefer in this case that Team Lead would help you look into teams. Just look at "okay, what is going on" and I will try to describe it a bit and then you can extract it and write it down. But then I suggest that you show it to me so I can review it, and not because it concerns stuff etc, but it is about seeing if I explained the story well enough. Let me talk about the dream of a software developer. The dream of a software developer is that you have an agile sprint of two weeks for example. Then at the beginning of the sprint, you know what needs to be done and then you start two weeks of coding. After two weeks you deliver exactly what has been requested. Two weeks and done. It's a dream. As a developer, you have your favorite music on, then you are just coding, some would say that sounds really boring, but you are actually creating stuff. Many people don't understand this but you see things when really being creative, out of your mind. If at a certain level you would come into a certain state, and it is called flow. If you are in the flow you are just happy to be working and when it is five o'clock, its sometimes hard to stop because you are in the flow and you are really enjoying what you are doing. The cool part of this is that as a developer if you are in that state, then

you are producing the best software. Because you are happy, you are doing your thing. Now, reality check, because I already told you it's a dream. The period of two or three weeks is a long time and especially in retail. You know, we provide software to retail and that means that the reality of one week or two weeks ago can be completely different to where we are now. It can happen that the work is not prepared enough and the developer gets a task and needs to think "what do I do now?" and gets asked difficult questions. Then you become frustrated because you have to talk way too much and try to figure out what should I do and if the beginning is unclear, if the requirements are not written then you know that you are going to deliver something in the end, but it is not what they expect, but I can't do anything about it because I'm not exactly sure on what to build. This might sound a bit strange but it is the reality in many cases. Then you get feedback and then finally a requirement, but then the problem is that you are much closer to the delivery time with the customer and then there is stress and pressure. Then you are not in the flow anymore. So there are many things based on that, that you can say, that we are drifting away from having a developer in the flow.

2.1.1. What specific data are you missing today?

I think the most important data I feel is missing can be categorized into three or four key areas. First, whether the goals are clear enough for the team within a given period and how many goals are being introduced during the same period, essentially looking at the scope changes and their impact on the team's focus. Another area is the quality of preparation before work is picked up. This includes clarity in the acceptance criteria and requirements, ensuring we understand what to build and how to test it properly. Additionally, the number of defects delivered from the team to external teams and whether the features are immediately functional is also crucial. These KPIs, goal clarity, scope changes, preparation quality, and defect rate, are important because they provide insights into why outputs might fail or come back for rework. For instance, if scope changes occur frequently, the team needs to reset its focus, which can reduce efficiency. If requirements are poorly written or include false information, it becomes challenging to deliver a good output since the input itself is flawed. Understanding these factors through these KPIs would help identify pin points and determine where to focus improvement efforts effectively. I think these are the most important areas to address.

2.2. Have you experienced problems accurately identifying when a team is losing effectiveness?

Absolutely. I have experienced problems accurately identifying when teams lose effectiveness, and it is usually tied to disruptions. A disruption can occur when a team member leaves, for example, or when too many people outside the team are demanding priorities. In such situations, the team leader struggles with goal setting, trying to address too many tasks at once. This leads to all tasks progressing slowly, taking more time, resulting in more defects, and creating additional stress within the team. Proper priority setting and focusing on a few core goals at a time are much more efficient than trying to respond to everyone demanding attention. Disruptions, incorrect goal setting, and unclear or false requirements also lead to decreased output. When requirements are unclear, deliveries are often met with dissatisfaction, causing further disruptions as people complain about the results. Essentially, all the KPIs I mentioned earlier, if they indicate poor performance or issues result in decreased team velocity. These disruptions are always a looming risk, so even with processes in place, it's essential to protect the team, ensuring goals are clear and requirements are solid before beginning work.

2.2.1. What causes the problem?

2.3. What challenges have you encountered when measuring effectiveness within teams?

I tried it a bit, but I struggled. At one point, I tried to understand the defect rate, for example. However, imagine you're developing a feature, delivering it to production or pre-production, and then a bug comes back. If the bug isn't connected to the feature, how do you know the bug is related to that feature? So, when it comes to KPIs, the first issue I encountered is that the data underlying these KPIs is unstructured. Today, that data remains unstructured. I think part of the solution needs to involve aligning how we work. I didn't test or try EazyBI, but I struggled to get the KPIs right. To get them right, things need to be more aligned on how Jira is used, for example. That's a big problem for accuracy. This misalignment can already be an outcome to address. We need insights into how Jira is used to ensure teams are more consistent, which will also help us extract more effective KPIs. For

instance, some teams work with time estimates, while others don't. Some teams estimate time on stories, while others estimate time on issues. There are all kinds of different approaches, and this inconsistency makes it difficult to get proper KPIs out.

2.4. What challenges have you encountered when measuring the effectiveness of separate individuals in a team?

I have not tried that.

2.5. What would you like to improve in the current process for measuring performance?

I think we've already touched on this a bit in my document. I would really like to figure out if there's a way to make KPIs more visible. Having a starting point would allow us to begin with some basic KPIs and identify issues, even if those KPIs show things failing initially. This can serve as a first iteration, helping us refine and improve over time by understanding why things are failing and gradually fixing the underlying problems.

Ultimately, I hope the company can work toward figuring out how to gather and use these KPIs effectively.

2.6. Do you see any problems with measuring performance at an individual level versus the team level?

I think here you have three goals, you have the individual and you have the team and you have cross teams. There is a bit of sensitivity around individual KPIs and it is less sensitive when we talk about team KPIs. Because I'm just giving this as an example, imagine you are a developer and at one point I walk into the office and say "hey you produced 13 bugs this month", that is insensitive and it is definitely not helping in a culture where you want to create creativity and make sure that people feel comfortable. At one point when you see someone who is really not performing you have a couple of conversations. But the more important thing for me is how is the team functioning as a whole? Because I think that when you talk and analyze the thing that you work on in the process and also enable the team members based on the feedback and the KPIs they see, to take a measure and adapt and make sure that there are no issues and improvement is as a team effort. And I think that as in most cases there is much more value to looking at a team than looking at the individual level. Because if you look at the team level and you start discussing with team members, let's say the cycle time, the cycle time is for example when a request for a change is coming in until the change is actually delivered. It's abstract but if you see big differences in the cycle time, then you can start talking to teams and say "hey guys, what can we do about this?" and this is about team effort but also for the guys that are working around a team. Then they start talking about this and collaborate on this and will do a couple of things you know. They are looking for how they can optimize and in the end in most cases the team is really capable of customizing the way of working. They are in the end the persons that are doing the work. If a manager is too much involved in this optimization, then the risk is that the developers are going to say that the managers are not doing the work, they are just observing. As a manager you are a person that is a bit more outside the team, so you have a different point of view and you can give them different insights. So team based KPIs and then discussing them as a team is for me the most important thing if we talk about a structured approach of measuring. I think it also delivers much more positivity than going to an individual and saying "hey, you produced too many bugs". That is really not motivating.

3. Implication questions

3.1. What do you think are the consequences of not having a clear method for measuring performance?

Lets say frustration. Customers become frustrated because the company fails to deliver on its promises. External stakeholders feel frustrated because the company isn't performing as expected. Employees experience frustration due to stress, witnessing issues without seeing corrective actions, or only seeing ad hoc responses rather than structured solutions. This can lead to good employees leaving the company. While not having KPIs isn't the sole reason for these problems, it's definitely a significant contributing factor.

3.2. What effects do you think an improved method of measuring efficiency would have?

I think an improved method for measuring efficiency would provide clearer visibility into how time and resources are being utilized.

3.3. What effects do you think an improved method of measuring effectiveness would have?

It would ensure that teams are not only productive but are also delivering valuable outcomes.

4. **Need-payoff questions**

4.1. Do you think that both aspects, effectiveness, and efficiency, should be measured to the same extent?

I think that currently, the KPIs I'm proposing are more focused on effectiveness than efficiency. At the same time, delivering your step right the first time is really efficient. But I would prioritize effectiveness first and then start looking into efficiency because, when focusing on effectiveness, you automatically gain some efficiency as a result. For example, if you have proper requirements and deliver with fewer defects, that's very effective in making the customer happy, but it also boosts efficiency because the velocity of the team increases. I've also noticed that some measures overlap, they contribute to both effectiveness and efficiency in some way. However, for me, effectiveness always comes first because you can be very efficient at doing all the wrong things.

4.1.1. Why do you think so?

4.2. How would a tool that measures performance improve the way you manage projects?

It will provide insights and show when things are not going in the right direction. This creates a starting point to talk to people and address issues. For example, if you can point out wrong requirements, you can begin improving them. After working on those requirements for a few weeks, you can check the KPIs again to see if they've improved. If they have, you can identify the next area to optimize. For me, it's a starting point for a more structured approach to optimization. This indirectly helps manage the project because these KPIs are already focused on how the team collaborates. The side effect is that project management becomes easier as we see what's actually being done. So, while the tool doesn't directly manage the project, it has an indirect effect by providing clarity and structure.

4.3. What types of information would help you make better decisions about team performance?

4.4. How do you think that an improved measurement of effectiveness affects the team's long-term growth?

I think it will aid in optimization of the team.

4.5. What would you like to see in a tool that provides a more detailed picture of the performance of a team?

So there is cycle time, how much time does it take when something comes in, before it is resolved. That is interesting because that means that if I ask something then I already have some kind of prediction of when it is going to be delivered. And the scope change, as I was discussing - how many things are injected or changed when I am doing my work. What do I mean with change? Imagine they would ask me "build a self-scan with X feature" and one day later there is a big change in scope with another component. So one day you were doing something but now you need to change your scope. That results in real efficiency loss. In my example, a couple of minutes ago I was talking about clear requirements and acceptance criteria. So you would like to measure how clear is the work actually defined. So you start talking about how we can find a way, how to measure these kinds of acceptance criteria. And there are a couple of these measures as well. Again, the team as a whole here is more important. I will send the document to you with some KPIs.

Appendix 8: Protocol of SPIN-Interview with Software developer

Before the interview, provide information on: information letter, consent form and recording of the interview.

Date: 21-10-2024

Duration: 41 min

Location: Google Meet

Participants: Software developer & Laura-Liisa Lillemäe (Student)

1. Situational questions

1.1. How are you currently working with measuring performance in terms of efficiency within development teams?

At least in our case, we are based on time, because we have to deliver on time. We usually do it in such a way that it needs to be implemented at least in a way that it works. Maybe it is not the greatest solution, maybe it is a little bit dirty, but if it is working, that means it is good and we can deliver and the clients are happy. Which is great. But we don't do that usually, if it is an important feature we postpone it so we can implement it better and so we don't face the same problem in the future again.

1.2. How are you currently working with measuring performance in terms of effectiveness within development teams?

I mean, it is not like a constant thing. We usually change the way we are working because we have to be efficient. So it is kind of 50/50. Both part are at good level.

1.3. How often are follow-ups made regarding the teams' performance?

We do have retrospectives and that one is a pretty good meeting to discuss about what went well and what didn't. Basically, in the end we try to find ways to improve and make notes based on that. Then, for the next couple of sprints we are trying to implement those notes. If it is successful then great, if not, then maybe we can rethink it a little bit. Maybe dive into some details. It is usually related to, not development itself, but the manager, from the manager perspective. Like how we plan sprints. It needs to be properly calculated, how many tickets we should handle during that period and it's kind of hard to do. There is also a lot of dependency because not every issue is depending on us. There is different connection between mobile team, backend team, management and clients as well. So, it's pretty hard to do a good sprint. But yes, we usually use retrospective to improve the situation. Also, you mentioned tools from Jira, we also use those. We have some charts like burndown-chart and velocity. There you can see how many Story Points, it's like an alternative to time. So, there you can see how many we planned, how many we completed and how many we didn't manage to complete. It is pretty useful. We have another meeting than retrospective to look at these charts. We have sprint planning and sprint review, so actually two meetings. These meetings are in the end of every single sprint. The sprints are two weeks.

~~1.3.1. What methods do you use to follow up the work of the team?~~

1.4. How well do the current practices capture the performance of a team, in your opinion?

Well, that is a pretty good question because now a days, in my team there were some big changes in terms of new colleagues, new people basically. But we elders do some talks on different topics and help. We want to do an onboarding plan so that every single person who joins the team, it will be much more easier to dive into the project and into the workflow as to say. Because hey, we have a lot of meetings, we have our own way of working and it is hard to dive into. But now we achieved good results I would say with the onboarding. But yes, management is already on place and always trying to make it easy.

1.5. How well do the current practices capture the performance of separate individuals in teams, in your opinion?

No, it is not measured. Well, we have some kinds of charts for that but when you let's say try to estimate how many Story Points our team is able to do or to finish in that sprint, we kind of have some

calculation on our side. Like how many Story Points I will be able to handle, how much Story Points will you be able to handle. Because every single experience from one developer to another is different. Like one guy, in one sprint, can finish eight Story Points, another guy can finish four Story Points. So yes, we do use some kind of calculation, but I haven't seen any charts from jira so that you can do a calculation there. So, it is done verbally but it is based on some statistic, data from the past. And we have one person who does that.

2. Problem questions

2.1. What challenges have you encountered when measuring performance within teams?

Yes of course. I would say every single sprint is challenging. One guy says "no, we have to use twenty Story Points", another guy says "no, we have to use twenty-five Story Points". So we have to make some compromise on that. But we are trying to get any ideas on why we have to put more or less and you can say like "look, this issue is kind of complicated and we don't have enough information". So it may be easy but what if key part of information is missing. Then we have to talk again with managers or from the backend team and we have to add that data back and it takes time. So it is always challenging, but yes it is part of our job.

2.2. What challenges have you encountered when measuring the performance of separate individuals in a team?

It is the Scrum Master who counts everyones Story Points on paper. He does some calculation how many Story Points we should use in one sprint, because it is usually not following the charts from the Jira. Because from the Jira, it can give you, lets say, if this sprint you finished like four Story Points and previous sprint you finished like twenty Story Points, it will suggest you that next sprint, measure thirteen Story Points, which is like medium. But it ignores the fact that some guy can be in vacation, some are sick or have some days off. Maybe next sprint we have some difficult tasks which may require more. During development whatever can happen, and you don't know. Because it is a little bit unclear and in this way, we can adjust it and put it lower or higher. Which Jira does not allow. So yes, we do some calculation on paper and trying to estimate this stuff. That guy knows enough so he has some equations and calculations for estimation. I don't know that Jira can do that.

2.3. What would you like to improve in the current process for measuring performance?

To have a good estimation, first of all we have to have a good described Jira ticket. Without the good described Jira ticket, it is really hard to estimate it. Well, if we can see that this ticket is three Story Points, then we can estimate it really easy, then it will take "this" time. But if there is no information, we are trying to "guestimate it", to guess some Story Points on that, which is wrong. It's ok if you have one ticket, but if you have two, three or five, then it's a mess. So yes one key is to have good described Jira tickets. And in forms of effectiveness, when it comes to achieving goals, it is a matter of team work I would say. Because if everyone is focused and working together on fixing a specific occasion, it is kind of, not easy to fix but much faster. It is more effective. Because we have a lot of developers focused on the same issue and there is a lot of discussion with backend and managers. Then it is effective because everyone is involved. Because if you finish that ticket and deliver it, the clients may say that "we didn't expect that, you have to do it again". The whole team should be focused on the problem itself. This would be the most effective ticket to deliver. To do that, we usually have meetings like team-programming sessions or we create a room where everyone joins the meeting and we brainstorm the solution.

2.4. Do you see any problems with measuring performance at an individual level versus a team level?

Not really. Well at least in our case it is pretty straightforward because, solution for that you can see a couple of past sprints. You can see like, that guy finished that sprint with four Story Points. Another finished sixteen. So based on some data you can predict user Story Points. And again, based on those Story Points you can summarize them and you can get a team performance. So it's kind of the same. But in some points you have to make consideration on vacation days, sick leaves and national holidays and stuff like that. So yes, it is harder to estimate one person than a team. But they're kind of both, the same, it's like just a summary of every single guy's performance.

2.4.1. In what ways is the work being affected by measuring performance at an individual level, in your opinion?

If you say that "okay, no worry I will finish the thirteen Story Points in this sprint" and then for some reason you have headaches or I don't know something happened, it's a psychological part, it doesn't affect the real work, it's a pressure. It is much easier to say "okay, we as a team we will try to accomplish that much of Story Points" and everyone can like do as much as he can because in this way there is not any conflict between the developers. So, if also I'll take four Story Points so even if I finished those two tasks let's say, then what I can do, I'm not going to say like "okay whatever, next Story Points are for you". So yes, it's a more stressful but I don't know, if maintainable. But again, it's a work-related thing and why we are getting paid.

3. Implication questions

3.1. What do you think are the consequences of not having a clear method for measuring performance?

I don't know if it's possible. I mean how you're going to do the work without measuring how much you can deliver in one sprint, it's unrealistic. In comparison let's say you are a car service, and you will not take ten cars per day to fix, it's like it doesn't make sense, you'll be able at least finish one and then it's good. I don't think that there are big consequences if individuals are not measured. For the individuals, there are more precise estimation like why your team performs only with two Story Points. Well because I can do ten and that guy also ten, so we have twenty, but it's not on us. why then it doesn't matter? I mean we know how much we can accomplish during the last spring and knowing how much every single person does it's like additional. Now the formation of course is useful but it's not mandatory it's like not a big deal. It really puts pressure, that's like "oh no, I can finish four Story Points this sprint". You can easily put on blames if you promise to complete five Story Points and then you didn't. If it's a whole team responsibility, then you blame the whole team but it's like not specifically you so it's less pressure than you which. I think is great and we are not judged by performance. I mean okay if you don't like the performance maybe do some talks, maybe you are not too well or what happens, maybe you have some problems in life. Such psychological things, it really makes a big impact on effectiveness. If you feel bad or maybe have some problems like you can't focus, a good gesture from the company is to help. If it happens that that guy is simply lazy then fine, just fire him. But I don't know, it depends on the case.

3.2. What effects do you think an improved method of measuring efficiency would have?

One thing that we don't like is just doing meetings for meetings and the statistic for statistical data. For data, I mean, data is always good, but it impacts on performance. So if you'll spend a lot of time just doing the creating the data, okay, congrats, you have some data but the question is, is the client happy? I mean our primary focus is to, not just making clients happy, but to create a good product and that good product will make that transit because there will be no issues in the store and the customer is happy. If the client is happy, everyone is in a good mood. So, the primary focus would be on the effectiveness, efficiency and the delivery. So, if we have a lot of space, I mean in time and we have a lot of persons who can be dedicated or created some data, good. But if you are taking that developer to do some data and also write code, that would be bad. We do that sometimes because, we discussed about Story Points, we do charts and we have retros, I mean we have some stuff from Jira which will require estimation. It's a good thing because you must do at least some estimations but in your case as you are asking if more data is good data, I think yes, but if there is a dedicated person for that. One person who don't do development and do only statistics, sits in the background and gets all this data, that's fine. But usually, companies want to be spending less money and getting more productive and usually it's the developer who then does that.

3.3. What effects do you think an improved method of measuring effectiveness would have?

4. Need-payoff questions

4.1. Do you think that both aspects, effectiveness, and efficiency, should be measured to the same extent?

Yes.

4.1.1. Why do you think so?

I mean this is why, in the beginning I didn't separate them because they are both. Because agile is like we must be adaptable, in one sprint we must achieve one goal, in another sprint make some code improvement and some documentation. So, it's kind of both 50/50 but it depends on the sprint by sprint and what kind of goals we have to achieve. Both are important yes.

4.2. Specifically, in what way, do you think such tools help prevent problems, such as effectiveness dip?

I mean if there will be some tools that can help, of course, but for now I don't see any, I mean well at least I can't give you some examples on what kind of tool to use. But of course, any tool which can help in effectiveness is a good tool. We usually are trying to use tools, but I mean, usually the output from that tool it's not usually realistic. We can't adapt it in our way because there's a lot of constants which the apps are not measuring. It's a lot of stuff which we can use from Jira, but it's kind of a raw product if I can say that. It can't give you a precise data on output. This is why we were trying to do some calculation on our part on paper so that it's specifically for our team. Maybe for another team it's working exactly but I don't know, by the end burndown-chart and stuff like this is easy to use. It's always there I mean with a couple of clicks you can see how the sprint went. I mean yes, it's a good tool. Well, it's more for the sprint-review than for improvement. But for us the effectiveness is clear. This is why I'm saying like for effectiveness and efficiency, we are the team, we have all the information because we are all on the product, we as a team. We are talking with each other, and we know the status. All this data I think is for manager or I don't know, for the clients, to see how productive we are. It's kind of an overview for the team, but it's like from our side, we as a team, we know our effectiveness.

4.3. How do you think that an improved measurement of effectiveness affects the team's long-term growth?

In terms of some charts, not really, in terms of retrospective, sprint planning and sprint-review, of course. Because we have a meeting which we must discuss about our problems but it's internal. Managers don't know about it. They know all the charts and all the statistics. You know so we know but the people from outside the team of course they need some charts and data. In this case we can say like more data is good data, because we get more precise information about everything, but again it's not for the team. It's for the people who is outside of the team, and they want to know the status. But for us we don't need statistics, we know that we failed in some part, and we need that information, and we need that improvement.

4.4. What would you like to see in a tool that provides a more detailed picture of the performance of a team?

You see you're asking the developer, but as I mentioned earlier this data is for managers. Maybe managers want to know how much this developer can do and maybe it's somehow related to costs and efficiency, if that developer is efficient enough to pay him that salary. For us as developers, statistics like Story Points completed don't provide practical insights for solving problems. Knowing the number of Story Points, for example, gives only an overview, not the detailed understanding needed to address specific issues. We handle problems directly, know how to solve them based on experience, and don't rely on external data to guide us. KPIs like Story Points, time spent, and complexity are useful for seeing our team's status from the outside, but we already know our teams. We discuss, take notes, and adjust our approach in the next iteration as needed.

Appendix 9: Protocol of SPIN-Interview with Software engineer

Before the interview, provide information on: information letter, consent form and recording of the interview.

Date: 21-10-2024

Duration: 19 min

Location: Google Meet

Participants: Software engineer & Laura-Liisa Lillemäe (Student)

1. Situational questions

1.1. How are you currently working with measuring performance in terms of efficiency within development teams?

I think it's in Story Points and Velocity.

1.2. How are you currently working with measuring performance in terms of effectiveness within development teams?

I think, how much repetitive work is done or how many times a ticket is going back. Actually, there is no measure for it to review it, there's just like a small interpretation.

1.3. How often are follow-ups made regarding the teams' performance?

Once in half a year. It's in the form of a discussion.

1.3.1. What methods do you use to follow up the work of the team?

Not sure, no. It is just verbally.

1.4. How well do the current practices capture the performance of a team, in your opinion?

There is a lack of history, lack of like references and no action plans. Actually, we have like this sprint-review, where we are discussing basically what was wrong what should be done better and so on, so this is done like every three weeks or at the end of every sprint.

1.5. How well do the current practices capture the performance of separate individuals in teams, in your opinion?

It is only done on team level. I think if there are any issues then those should be solved inside the team not by some KPIs.

2. Problem questions

2.1. What challenges have you encountered when measuring performance within teams?

Not me, because I think I'm not the right person to like make this particular reviews and decisions.

~~2.2. What challenges have you encountered when measuring the performance of separate individuals in a team?~~

2.3. What would you like to improve in the current process for measuring performance?

Not sure, maybe yes but not sure what it should be.

2.4. Do you see any problems with measuring performance at an individual level versus a team level?

So, as I had mentioned previously, I prefer more team level.

2.4.1. In what ways is the work being affected by measuring performance at an individual level, in your opinion?

Then it will affect the work so it's some kind of pressure added.

3. Implication questions

3.1. What do you think are the consequences of not having a clear method for measuring performance?

Money loss, the correlation between the delivering and the costs is affected directly. If the team itself delivers, if there are any issues in the team it is better for the team to make, at least I think when the issues are made on team level team themselves know where the problem is and they should have the option to solve it. So measuring individuals wouldn't matter that much.

3.2. What effects do you think an improved method of measuring efficiency would have?

I'm not seeing it as a good thing, more KPIs. Because in my opinion, at least in how we are working now every sprint or every block let's say, can have itself context that will have a big influence on the charts but will not show the real illustration. Because yes, I think that when you have more KPIs, KPIs are not like something that you should take a look after. KPIs or something like that you should follow and make sure that everything is going well and is in good shape for that specific KPI. That adds more stress. Maybe it's a good thing to add more responsibility on decision making, basically for every developer, but again it adds too much granularity on one decision in all the charts. From my perspective and from my experience, more charts do not mean something that can help but it's more like something that it's a reference for managers to make sure that everything works fine without any context. Just was it done or not, when it was done, how many of stuff was done by one person or by specific team and so on. So, it has no essence.

3.3. What effects do you think an improved method of measuring effectiveness would have?

It depends on like what are the goals. So it should be related to goals, otherwise it's just to make sure that the graphs look okay.

4. Need-payoff questions

4.1. Do you think that both aspects, effectiveness and efficiency, should be measured to the same extent?

Yes, but maybe efficiency is more is more important, maybe.

4.1.1. Why do you think so?

Because again, if we are differencing to error reports or any follow-ups by history, the efficiency is easier to demonstrate and it's easier to show or to illustrate. Effectiveness, I'm not sure about that. It's more about how people interpret, like it's more subjective.

4.2. Specifically, in what way, do you think such tools help prevent problems, such as effectiveness dip?

In some cases it can help.

4.3. How do you think that an improved measurement of effectiveness affects the team's long-term growth?

I think it's also important how or what actions are taken after those measurements because sometimes those measurements could be taken silently with some overall reviews. Maybe the period of reviews should be bigger and not like on every sprint or on every small period.

4.4. What would you like to see in a tool that provides a more detailed picture of the performance of a team?

Not sure about this, I think the timelines and what was planned and when it was finished. I understand that we can achieve this by combining some of the charts, but I think sometimes we're missing the, I'm not sure how to explain it, when it was basically promised to be finished and when it was finished. But we already have this kind of measurements so not sure. We don't have a chart for it, we have more like pipeline.

Appendix 10: Protocol of SPIN Interview with Director of Human Resources (HR)

Before the interview, provide information on: information letter, consent form and recording of the interview.

Date: 22-10-2024

Duration: 36 min

Location: Google Meet

Participants: Director of HR & Laura-Liisa Lillemäe (Student)

1. Situationsfrågor

1.1. Hur arbetar organisationen för närvarande med att mäta prestation hos mjukvaruutvecklingsteam?

Vi gör vi samma performance process över hela bolaget. Vi jobbar ofta med och är ett bolag som hanterar alla samma och vi har ett system som heter bamboo HR som har en assessment modul där. Där man gör en övergripande assessment både individ eller anställd och dess manager. Så när jag ska göra ett assessment gör jag ett assessment på mig själv och sen görs en assesment på mig. Och när båda har gjort det, då öppnas det upp så att man kan se varandra. Så det gör vi två gånger per år som en grund för att säga hur känner jag mig värderad, har jag allt jag behöver för att göra ett bra jobb, finns det något som kan göras för att det ska bli bättre, vad är mina styrkor och vad är mina utvecklingsområden. Så påminner vi alltid att relatera till målen och personliga utvecklingsplaner som man också har, lite är tanken att man ska lägga upp i systemet. Så att vi har liksom en och samma. Vi har inte ett en kompetens performance sheet, utan vi anser att man relaterar sitt jobb och sitt jobb description och sina mål i hur man responderar vad man tycker att man gör det assessment. Sen så kan vi ha sådana här strukturer där vi tänker om, men man kanske har varit lägre i utvecklingskurvan. Är man nyanställd, inte känner bolaget än, vad är typiskt att en naturlig utveckling då? Så då kan man anses vara en climber så att säga, man håller på att växa in i bolaget och i rollen. Men kanske man byter roll hos oss och då kanske man blir en climber igen och har utvecklingsmål mot att komma upp i rollen. Och sen har man olika mål utifrån vilken nivå man är på, erfarenheten och kapaciteten man kan räkna med att man har i det stadier man är och bolaget där. Vi har liksom one fits all, men i det så räknar man ju in när X utvärderar mig så tänker man ju på vad är förväntningarna på vår Chief People Officer. Och när någon utvärderar en utvecklare så relaterar man till vad är dina utvecklingsområden och vad är dina styrkor. Men i våra assessment värvar vi också in mjuka värden med kultur värderingar, samarbete och sådana saker.

1.2. Hur ofta genomförs uppföljningar av teamens prestation?

Man gör det två gånger om året, som man gör det större och faktiskt dokumenterar i systemet.

1.2.1. Hur används dessa resultat?

Då kan man ju bolla med varandra one-to-one för att följa upp vad visar det här? Finns det något gap, att vi ser saker olika eller har vi samma planer? Hur? Hur kan man stötta varandra för att göra progress i det. Vad har man för intressen och vad behöver man för stimulans och så vidare. Det är ju sådant som kommer i one-to-one vid sidan om och vi vill ju ha en ganska agil process. Vi tycker ju att det är rätt bra att följa upp utvecklingsplan och mål några gånger per år för det kan ju röra sig. Både att man stänger ett projekt och går in i nästa eller man möter olika faser eller utmaningar i det man sitter med och så där. Det ska vara lite som en grund till att ha en löpande diskussion kan man säga.

1.3. Hur ofta genomförs uppföljningar av enskilda individers prestation?

Man gör det två gånger om året och one-on-ones vid sidan om.

1.3.1. Hur används dessa resultat?

1.4. Hur väl upplever du att de nuvarande metoderna fångar upp mjukvaruutvecklarnas prestation på ett rättvist sätt?

Först och främst skulle jag nog vilja säga, programutvecklarna och hur det följs upp, det skulle man nästan kunna fråga en manager inom våra product engineering organisation. Men jag kan säga som feedback har vi inte fått att vi måste ha en annan typ av assessment tool eller någonting annat. Utan

igen, vi har ju en ram som gör att man fyller det utifrån en jobb beskrivning. Så jag skulle säga att på det sättet fylls ju ut ganska väl men det är ju inte en egen kompetens assessment modul för massa olika kompetenser utan det är generella assessment.

1.4.1. Fångar dessa metoder upp prestation på ett rättvist sätt?

Om båda parter gör det här bra och man är engagerad i det så gör det det. Det här krävs att man är engagerad i sin utveckling och att chefen är engagerad i att se till att alla håller i processen. Jag kan ge en liten bakgrund på vad jag tycker har hänt med sådana performance utvärderingar för bolag totalt sätt. Då har man ju dels blivit mycket mer agila och vill ha verktyg som förenklar för det. Tidigare så var sådana assessment mer som utvecklingssamtal. Det kanske gjordes en, två gånger per år med ett dokument med en miljon frågor. Det jag upplevde var att det blir som att gå igenom en screening. Man lyfter aldrig blicken och pratar, har vi bra expectation management och så vidare. Det blev mer formalia än dialog.

2. **Problemfrågor**

2.1. Vilka problem har ni stött på när ni har mätt prestation hos team?

Att mäta på team nivå är ju sådant som vi också gör i våra assessment för att pragmatisera lite. Om man tittar på hur nivån är spridd i teamet i erfarenhetsnivå, performance nivå, är det ett team där alla är högpresterande, har stor erfarenhet för att bära rollen, då kanske vi har ett team som bär risken av att alla kommer vilja gå vidare. Där alla behöver mer stimulans. Har vi ett team som är spritt i strukturen, för att det ska finnas en naturlig succession ordning, sådant kan vi titta på team nivå. Sen kan man ju inte bara mäta så utan man vill ju också se individen. Man vill också se kulturen bakom för om man bara rate'ar kompetens och inte mäter kultur, relation och kommunikation, då kan det bli fel dynamik i teamet. Att någon som är otroligt skild ändå skapar mycket osämja. Man behöver se individen bakom team prestation också.

2.2. Vilka problem har ni stött på när ni har mätt prestation hos enskilda individer i team?

Nu härddrar jag det bara men säg att ett team bär upp för någon som av olika skäl har en prestations dipp. Vi som arbetsgivare kan inte se saker hos den individen som vi vill stötta. Det kanske är saker i livet som sätter lite hinder eller hälsan som inte är på plats. Det finns saker som vi vill hjälpa med och röra mot rätt riktning. Men det kan också vara att någon börjar demoralisera och tänker att "han/hon kommer alltid undan, varför ska vi göra?". Då skapar det en negativ team-kultur. Bara små exempel på varför det är viktigt att inte bara se helheten i teamet men också individerna bakom det. Det har absolut hänt hos oss, nu pratar jag inte bara om Extenda Retail, just det här att det finns en stark kollegial situation som täcker upp för andra för att man ställer upp för andra och så har det slutat lite olyckligt. Vi hade önskat att komma in tidigare och vara med i stöttningen och vara proaktiva. Men jag säger inte att det här är något som pågår hos oss just nu men det är sådant man jobbar med som arbetsgivare i dessa situationer. Det är bara ett exempel på varför vi tycker att det är viktigt att koppla ihop individ och helhet.

2.3. Vilka etiska risker finns det med att mäta prestation på individnivå hos team, enligt din mening?

Om du skapar systemet och säljer det till oss, då kommer du säkert med krav på vad vi ska följa för etiska delar. Men vi kommer ju också vilja väva in det så till exempel Bamboo HR är ett system i sig som ramar in vem har tillgång, vem ser, vem gör. Sen lägger vi också till access levels i systemet och går ut och kommunicerar det som en förpackad process. Bakom allt det här så har vi ju redan hantering av personuppgiftslag. Här kan man dels behöva säga vem äger systemet? Vart ligger det? Gällande informationssäkerhet i leverantörsavtalet mellan dig och mig. Är det här i cloud? Är vi enskilt ansvariga för datahantering? Sedan är en annan sak det etiska ut mot användarna. Till exempel om jag gör ett djupare assessment, vilket vi har vid rekrytering, promotion och så vidare, där vi har tillgång till att göra förmågetester av olika slag. Det systemet i sig har ju en del i sig i hur man godkänner dom reglerna innan man går in i systemet och gör en assessment. Men just etiskt att vi beskriver hur det kommer att användas, hur ser vi på det, vem äger informationen och ber dom om medgivande, som du gör, "kan jag dela det här med X och Y". Det är just dom här bitarna som är viktiga för oss att hantera, att berätta att vi gör ett nytt verktyg, förklara varför vi har tagit fram det,

vad är syftet, vad ska det användas till, hur kommer du bli sedd i det här, vilken data finns och hur kommer vi använda den. Det gäller att vi också tittar på just personuppgiftshantering och GDPR.

2.3.1. Hur skulle dessa kunna hanteras på ett hållbart sätt?

Dels skulle jag tänka på sådana saker som vi var inne på, som är viktiga compliance mässigt. Hur länge ligger data kvar, vad händer när någon slutar, vad händer med trend diagrammen, om individer syns i det här. Just det här med GDPR compliance. Vilken data anses som personlig, vilket data ska individen ha rätt att säga "nu ska det här tas bort, då jag slutar". Hur lång är det relevant att titta tillbaka på trender? Sedan skulle jag titta på det här hållbara, hur bygger man trygghet och trust i hur man använder systemet. Hur blir jag sedd, är det i en grupp eller i ett team, kan vi komma dit att vi är så transparenta att vi kan dra lärdomar av varandra och alla känner att det är tryggt och transparent. Ju mer transparent det är desto mer finns det nytta av verktyget. Kan man hitta det här verktyget och ha det som bas när man jobbar och har en trygghet med att ha det som diskussionsunderlag i team, när man springer på problem eller kvartalsvis för att reflektera och ge lärdomar eller liknande när man stänger saker med lessons learned. Då tror jag att det har en starkare effekt.

3. **Inverkansfrågor**

3.1. Vad anser du är konsekvenserna av att inte ha en tydlig metod för att mäta prestation i form av ändamålsenlighet?

Om man inte mäter det alls är det väldigt lätt att dra slutsatser och då kan det bli fel. Sen behöver inte det vara att man har ett mätsystem för allt, det kan vara det här att man snarare har KPI trender och kan faktiskt titta på resultat, det kan ju räcka. Men man behöver ett visst underlag med relevanta exempel. Om jag bara ger feedback "jag tycker att du är ..." då är det en tolkning. Men när du gör såhär, då tänker jag att "det här skulle kunna göras annorlunda för att uppnå det resultatet". Man måste förstå datan, man måste se situationen. Men man kan inte alltid hoppa på feedback utan man måste också lyssna in lite. Det ligger mycket i parterna oavsett vad man har. Jag tycker att kunna räkna sin verksamhet är viktigt för att kunna luta sig mot det och kunna ge exempel på och fråga individer hur vi kan bli bättre på det här. Om man inte har bra KPIer eller data att luta sig mot, då är det svårare att jobba med förbättringar och löningseffektivitet. Sen behöver man också skapa trust i vad det är som dom här sakerna säger och att man förstår. Vad mäter vi? Vad säger dom? Så att man kan vara med och resonera. Där kommer det här med transparens och trygghet in. Sen om man mäter något som antingen mäter rätt sak men man gör andra tolkningar av det som inte går att väga in eller om man faktiskt mäter saker som inte är relevanta för det vi pratar om. Då är ofta förtroendet borta. När förtroendet är borta då har man inte riktigt en motiverad och engagerad individ, team, vad det nu är. Så det handlar, tycker jag, om trust.

3.2. Vad anser du är konsekvenserna av att inte ha en tydlig metod för att mäta prestation i form av produktivitet?

Samma gäller för produktivitet och vad konsekvenserna blir för det. Vägen för att hitta det är att göra som du gör och inkludera alla parter, hur ser man på det här, vad mäter det här, vad säger det här.

3.3. Vilka effekter tror du att ett verktyg för att mäta prestation har på arbetsmiljön?

Jag skulle säga att vi går tillbaka till det här med att förankra alla och förklara vi använder det. Jag kan inte värdera hur just det här mätverktyget skulle kunna påverka men jag kan säga hur man kan jobba för att det ska få bästa effekt.

4. **Need-payoff questions**

4.1. Vilka är dina uppfattningar om att mäta prestation på individnivå?

Jag skulle säga, utan att gå in för många detaljer, att det är ett ganska bra komplement att titta på hur teamen, utvecklingen och individen progressar. Vi kan säkerligen hitta just verksamhetsutrymme för "här behöver vi hitta bättre processer" eller "här behöver vi jobba mer med kvalitét". Så jag tror definitivt att det är ett kompletterande. Men jag tror också att individen behöver det här med lite mer assessment som går lite mer övergripande, vad mer krävs i din roll för att du ska bli framgångsrik, vilka sidor kan du jobba på och så vidare. Sådant som inte syns i såna här utvärderingar.

~~4.1.1. Varför anser du så?~~

~~4.2. Vilka metoder skulle möjliggöra en bra balans mellan behovet av att mäta enskilda individers prestationer och de etiska överväganden, enligt din mening?~~

~~4.3. Vilka egenskaper skulle du vilja se i ett verktyg som hjälper till att mäta prestationer på ett sätt som tar hänsyn till enskilda individers bidrag?~~

Jag har för lite insyn i det för att ge ett svar och det är inget negativt, det är sådant som jag kan komma in i sen.

Appendix: 11: Protocol of SPIN Interview with Team lead A

Before the interview, provide information on: information letter, consent form and recording of the interview.

Date: 22-10-2024

Duration: 36 min

Location: Google Meet

Participants: Team lead A & Laura-Liisa Lillemäe (Student)

1. Situational questions

1.1. How are you currently working with measuring performance in terms of efficiency within development teams?

We don't have any measurements for efficiency at the moment in any of the teams I'm involved with. We use story points and then we compare sprints by seeing the the amount of story points completed during the sprint, but that's not specifically for measuring efficiency. It's more for planning purposes, so that we can see what the capacity is for upcoming sprints.

1.2. How are you currently working with measuring performance in terms of effectiveness within development teams?

We don't have any specific measurement for effectiveness. It's more of if we are missing deadlines, then we react on that. But we don't actively measure it and follow up on the measurements currently.

1.3. How are you currently measuring the performance of separate individuals in teams?

It's also the same answer there. We don't have any specific measurements for individual team members at the moment, and we don't do any measuring at all in that regard.

1.4. How often are follow-ups made regarding the teams' performance?

Performance is indirectly followed up through story points and informal observations, but there's no structured or systematic follow-up.

1.4.1. What methods do you use to follow up the work of a team?

Story points and capacity are used as rough indicators, but they are not direct measures of performance.

1.4.2. What methods do you use to follow up the work of separate individuals in teams?

Observational methods are used, but no formal system is in place.

1.5. How well do the current practices capture the performance of a team, in your opinion?

The current practices only partially capture the performance of a team. While we can see KPIs like story point velocity at all times, these do not provide insights into how efficient or effective the team is. They offer an idea of capacity, such as whether it is dipping, over the average, or stable, but they do not allow us to draw meaningful conclusions about the quality of performance. Decisions are primarily based on quantity rather than quality.

1.6. How well do the current practices capture the performance of separate individuals in teams, in your opinion?

The same applies to individuals. For individuals, other signals, such as observations from managers or team members, play a role. For instance, if a team member is performing below their known capability, it might signal the need for action. However, these observations are intuitive and subjective, lacking systematic or data-driven analysis.

2. Problem questions

2.1. What problems do you experience in getting an overall picture of how productive the teams in your department are?

The main issue is the lack of a systematic approach to measuring effectiveness. Without proper measuring capabilities, it is difficult to draw objective conclusions or set tangible goals that can be tracked and followed up on. This makes it challenging to work towards improvements that are measurable. As a result, the team often operates in a state where it is hard to identify areas for improvement or track progress constructively and productively. A problem that occurs frequently is achieving the goals we set, because it is hard to create positive change without the ability to measure progress effectively. While the team may know what it wants to achieve, there is no structured way to guide efforts towards those goals. Additionally, it is difficult to determine the current status of the team or compare it to previous periods, such as three or six months ago, due to the absence of active measurements. Instead, decisions are often based on intuition and personal observation rather than data.

2.1.1. What specific data are you missing today?

I would say that specifically, we lack data on an individual level for developers. Are they performing as expected? Are they exceeding expectations, or is their performance declining? Is there any reason for me as a team lead to take action, and if so, what kind of action should I take based on trends and behavioral patterns? Right now, we don't have the data to answer these questions. For example, can I objectively see if a specific person is underperforming? Can I compare their performance to others? Is the entire team underperforming, or is it just an isolated case? Different trends require different actions, but without the right data, it becomes more of a guessing game. Instead of having a systematic, well-informed platform to make accurate decisions, we rely on assumptions. We lack insights into whether developers are improving, delivering more story points, or if the features they release are stable or filled with bugs. Additionally, we don't have data on their day-to-day work—whether they follow good developer hygiene, commit code frequently, or break down their work into meaningful chunks rather than trying to complete everything in one go.

2.2. Have you experienced problems accurately identifying when a team is losing effectiveness?

Yes.

2.2.1. What causes the problem?

Because there's no structured system to identify or track effectiveness systematically.

2.3. What challenges have you encountered when measuring performance within teams?

I have used Excel, but it was very manual. I had to collect all the data myself, and while it gave me some new insights, it wasn't maintainable. It took too much time, and storage was limited. Any new KPI I wanted to track required a lot of manual input.

2.4. What challenges have you encountered when measuring the performance of separate individuals in a team?

The same challenges applied to both. It wasn't always clear what to measure, and I would have liked to have a well-researched, well-understood toolset for effective and efficient measurement. At the time, I didn't have that information.

2.5. What would you like to improve in the current process for measuring performance?

In addition to data intake and presentation, I would like to improve the models themselves. What should we measure, and how should we follow up on signals from these measurements? It's important to understand what trends we can identify, their possible causes, and what solutions we could explore. The key issue is gaining a better understanding of the models that form the foundation of these measurements and the actions that should be taken based on the signals they provide.

2.6. Do you see any problems with measuring performance at an individual level versus the team level?

Yes, it's sensitive because you don't want individuals to feel like they are under constant surveillance or strict measurement. It has to be clear that the goal is not to micromanage but to address challenges in a constructive way. The purpose is to support individuals, helping them become more productive in a way that benefits both them and the team. It's important to be careful not to create a feeling of being monitored too closely but instead to be very transparent. The idea is that this is a toolset designed to improve everyone's experience, especially the individuals themselves. It should provide useful insights, guidance, and support rather than just being a tracking mechanism.

3. Implication questions

3.1. What do you think are the consequences of not having a clear method for measuring performance?

It leads to a lot of stress because decisions are based on incomplete or inaccurate information. Without proper data, there's a higher risk of making poor decisions, which can create frustration for both leaders and team members. For leaders, the lack of data makes it difficult to take the right actions, leading to uncertainty and inefficiency. For individuals, it can cause stress because expectations and performance assessments become unclear or subjective. Ultimately, this impacts output, which is critical in an environment with tight deadlines and small margins. If performance measurement were clearer, it could help optimize workflows, reduce stress, and potentially even allow for more flexibility in deadlines.

3.2. What effects do you think an improved method of measuring efficiency would have?

Like I mentioned before, the primary impact would be reduced stress levels, both for individuals and at the leadership level. Happy customers would also be a result, as we would likely deliver on time and with fewer bugs. This would also improve team spirit and reduce friction within the team as well as between leadership and individuals. I see many improvements overall. Moving from no measurements, as we are now, to implementing some high-quality measurements with actionable tools to address observations would be a significant step forward. It doesn't need to involve a large number of KPIs, measurements, or monitoring. Just having some standardized tools in place, rather than none at all, is already a major improvement.

3.3. What effects do you think an improved method of measuring effectiveness would have?

Same goes for effectiveness and efficiency.

4. Need-payoff questions

4.1. Do you think that both aspects, effectiveness, and efficiency, should be measured to the same extent?

Effectiveness is the more important one.

4.1.1. Why do you think so?

The most important thing is to ensure that you are doing the right things. You don't want to waste time or mental energy on the wrong tasks. Once you focus on effectiveness making sure the right work is being done, efficiency naturally follows because there's no unnecessary effort being spent. Additionally, maintaining effectiveness helps keep individuals and stakeholders in a productive mindset, which also tends to improve efficiency. So, effectiveness is the primary driver, and efficiency is something that follows from that I think.

4.2. How would a tool that measures performance improve the way you manage projects?

It would make planning much easier and allow us to identify disruptions early on, making it possible to communicate these to stakeholders well in advance. This would prevent situations where, just two or three days before delivery, you have to inform stakeholders about delays. Such a tool would make it easier to foresee potential problems, plan for the future, and facilitate more informed discussions among all individuals involved. These discussions would be more focused and productive, avoiding blame games or silent frustrations where people might feel something is wrong but can't articulate it. Additionally, it could help reduce burnout by ensuring that people stay healthy, both physically and

mentally and feel happier about coming to work. Overall, I see many potential improvements from using a tool like this.

4.3. What types of information would help you make better decisions about team performance?

I would say insights into individual and team performance trends, quality of work, and behaviors of team members.

4.4. How do you think that an improved measurement of effectiveness affects the team's long-term growth?

I think one critical aspect is team building. When you have a team of strong individuals, say between four and twelve people, the idea of forming a team is that the collective effort should generate more value than individuals working separately. If you simply assigned tasks to each person individually, with no collaboration, you'd miss out on the synergy that comes from teamwork. By putting those individuals together in a team, you're aiming for added benefits, things like shared ideas, collaboration, and mutual support. A well-functioning team can be two, three, or even five times more productive than the same individuals working alone. With good tools to measure performance and provide actionable feedback, you enhance team building, team spirit, and collaboration. Over time, this improves the team's potential and results in better outputs. It also creates a more meaningful, enjoyable workplace for team members, positively benefiting both their productivity and their overall experience.

4.5. What would you like to see in a tool that provides a more detailed picture of the performance of a team?

As I have mentioned, I think, some actionable KPIs, those that we discussed during the brainstorm are really good and we did discuss why they could be good for our team also.

Appendix 12: Example of a Custom Calculated Measure in EazyBI

Change failure rate (CFR)

[Measures].[Bugs in Production] =
([Discovered in phase].[Production], [Issue Type].[Bug], [Measures].[Issues closed])

[Measures].[Total Tickets in Production] =
[Measures].[Issues closed]

[Measures].[Change failure rate] =
([Discovered in phase].[Production], [Issue Type].[Bug], [Measures].[Issues closed]) /
[Measures].[Issues closed]

Appendix 13: Protocol of MVP 1 Test with Senior Frontend Developer

Date: 27-12-2024

Duration: 23 min

Location: Google Meet

Participants: Senior frontend developer & Laura-Liisa Lillemäe (Student)

Student: This will be in the form of thinking aloud, I will be as quiet as possible and you will have the chance to say what you think about the dashboard that you have tested.

Senior Frontend Developer: That is regarding the velocity by assignee KPI. The data looks incorrect. For Sprint 10B, it shows I delivered only 3 SP. However, when I check the sprint report, there are two tickets with 5 SP and 3 SP.

Student: The story points for Sprint 10B didn't count correctly because one of the tickets was marked with the "In Delivery" status instead of "Done." easyBI considers only tickets in the "Done" status as completed when calculating delivered story points. The Team Lead mentioned that the "Delivery" status isn't typically used to close issues, and this will be corrected manually. Going forward, tickets will only be marked as "Done" before the sprint closes to ensure accurate story point counting.

Senior Frontend Developer: I already have selected the correct projects for the KPIs, but it seems that I have to do this every single time I enter the dashboard.

Student: Yeah, it should not be that way. I will definitely look into why it's resetting.

Senior Frontend Developer: Yeah, because for sure whenever I set this up, I selected a number of eight to 10 projects or something like that in my board and I selected all of them for every single widget.

Senior Frontend Developer: I don't have the CFR KPI by assignee which does mean that if I see a value of five bugs in here it's the number of bugs that the team has based on the filters, but I would like to see not always to compare myself with the team but if I was a manager. I think the return of investment for doing this is to have some KPIs for each and every one of your employees. So somehow I would like to know that my team delivered an average of let's say five bugs, the total number and four of them are created by the same person, then I can make a decision. Maybe I will talk to that person. Okay, let's see what improvements can be done and so. But if I only see the total number of bugs per team, I don't know if that is that great. I mean, I can only compare the number of bugs, let's say for the last quarter compared to the previous quarter and see that there is an improvement. I guess that if I were a manager, it would be nice to be able to filter it by employees here. The Change failure rates of the team, it doesn't mean so much for me. I don't know how my stats are compared to my team for example and I would like to know that to know what my performance is. I know I see that there is a bug in production, but I don't know, is it mine or who it is?

Student: When I talked to the Team Lead, I also asked "How do you work when a bug comes in, is it connected to the person who caused it?" but the Team Lead said that the bugs come in and everyone can solve them. So they're not connected to the one who caused them. But if it is so that it is connected to the developers who caused it then it would be possible to gather this data and to do that chart.

Senior Frontend Developer: So in theory at least you can map the bug to a previously done ticket so that you should know which one of your employees was the one that was assigned to the initial ticket and if that bug is mapped to that ticket then the bug was created by you basically that it's simple.

Senior Frontend Developer: There should be another discussion for the Cycle Time KPI. From my perspective, my work is done once a ticket is reviewed and moves to the QA lane. However, QA might find issues, causing back-and-forth between QA and the to-do list, but this usually doesn't take much time. If a bug is found, it might take another week to resolve. The problem is, tickets marked as 'done' only count after leaving the QA lane. For instance, if I complete 16 story points in a sprint but only 12 are tested by QA, the KPI suggests I delivered 12 story points, underreporting my actual work. This highlights a bottleneck in QA, especially if one tester is overwhelmed by tickets from multiple developers. Cycle Time can also be misleading. For example, the first ticket in QA might be processed quickly, but the last could wait days. Seeing an average of 35 minutes for QA seems wrong, likely due to a calculation error.

Senior Frontend Developer: As for the dashboard as a whole, I would like to have a filter outside of those widgets. Meaning if I would have the option to set a filter somewhere outside of the widget these are the 11 projects that I would like to filter the data based on and then apply the filter to all of the widgets. Again, it's a little bit too much to filter 11 projects. Till now that are my findings.

Student: I will just ask one question, the KPIs themselves, how do you think they help you in some way?

Senior Frontend Developer: I really like this one (Velocity by Assignee). For me I don't see the benefits for me of the Change failure rate on team level.. while this one can be very useful (Cycle time by status) but once again I have wrong data in here for it. This one (Cycle time by status - Current Assignee) would be also very very beneficial because somehow it will tell me where I have bottlenecks. I can see that a developer is delivering two story points, but maybe the issue is not that a developer is working only on two story points, but the fact that the developer finishes 16 story points, yet out of those 16 story points, the QA team can only test two story points. Then I will know that the bottleneck is not on this developer's side but on the QA team. And I should be able to see here that the average time spent of a ticket in the QA lane is more than, I don't know, one week. And that will provide me the information that okay I should do something. Maybe I need to hire someone else to help the QA team do better. So these together are the ones that for me are very very beneficial. Again having in mind an abstract manner not with those values. So that is my feeling.

Appendix 14: Protocol of MVP 1 Test with Team lead B

Date: 27-12-2024

Duration: 9 min

Location: Google Meet

Participants: Team lead B & Laura-Liisa Lillemäe (Student)

Student: This will be in the form of thinking aloud, I will be as quiet as possible and you will have the chance to say what you think about the dashboard that you have tested.

Team lead: I really like the ability to look at the Change failure rate. only as it is now it's purely on overall projects and then with so many teams and me having focus just on one team it becomes very very useless. Now it would be really nice if you could somehow pin that on a project. I also was thinking if it would be useful to have that on a person, but I think that's not useful because you're always doing these things as a team and refining things as a team and also having the reviews in there, etc. Also, it would be nice if you could change the time range because currently it's only possible to say it for the last six months. would be nice if you could spend it to a year or have a more detailed view per week or something like that.

Team lead: For the velocity by assignee I was looking at it and I think it's a nice indication, but the thing that I currently have and in that sense it's good that we're testing in the holiday period is that some people leave this sprint and you're looking at it and then it comes like okay yeah this person has only one or two points. But she only worked for one day. and then it would be nice if you could have a look at some past sprint or something like that. Or I don't know stuff like standard deviation for this person from the average. So you can have a look just this person maybe if someone has let's say, an average of 15 points but also have an average deviation of 15 points that's an indication that this person usually carries over the task. So then one sprint they've closed nothing and the next sprint they close double that kind of things. That means the person is processing the same amount but it's not that predictable yeah that kind of stuff would be useful and as I said to be able to look into a past sprint also to see okay what's happening in the holidays etc yeah that's pretty much it for me.

Student: Is anything missing in your opinion?

Team lead: For now I focus mainly on having a look at these KPIs. What is a big challenge in these is to kind of see people have different roles. if someone is more focused on for example aligning the other team members etc. sharing their knowledge, helping them out on pull request etc. that will somehow be lost in a chart by this kind of velocity. I don't know how to measure it. It's just something that it's important to be aware of, that kind of factors.

Appendix 15: Protocol of MVP 1 Test with Engineering Manager

Date: 27-12-2024

Duration: 20 min

Location: Google Meet

Participants: Engineering manager & Laura-Liisa Lillemäe (Student)

Student: This will be in the form of thinking aloud, I will be as quiet as possible and you will have the chance to say what you think about the dashboard that you have tested.

Engineering manager: The Team Velocity by Assignee KPI is useful and dangerous. It is useful in that sense that you get better insights and it can be used as one I would say of the KPI KPIs to measure on an individual level, which is good because that is something we missed, and in combination with GitHub contributions you can see the frequency from GitHub. I think in combination with also your own observation it gives you useful insights. It can also be very misleading because if you look at this there are basically two risks for me. The first one is if you look to moment hydrar you can say okay that guy is not productive but in fact he's the product owner so you need to understand his role and that means that his role is giving value for the team in a different way than a software engineer. So in that sense you always need to understand the context when you're looking into it. Some other guys as well, they are sometimes busy with helping others, that is not reflected in the statistics. So they have value, but within their context and you should not depend only on velocity. But I'm glad that they are now finally visible. The other risk is that if you share this KPI in the team, the team starts to become aware of what the velocity is for others. What you will get is a sprint of people that try to get the highest velocity and velocity necessarily does not equal value. So you need to be aware of how you look into this very useful KPI and you need to make sure that people are not going to use this as their main goal.

Engineering manager: So that is my observation on this one but again basically I think it is very good that we have this one in any questions for me regarding this one?

Student: No just that right now I have two developers also testing and the developers see only story points for themselves. They can't even get the option to see for others. So maybe it should be left that way that this kind of KPI where you see all the team members shouldn't be seen by all the team members, or what do you think?

Engineering manager: No, no, no. I think that is in our first call, I did call this one a hidden feature for the developer because you don't want to get people that are purely focused on velocity.

Student: But then we also have cycle time and Change failure rate.

Engineering manager: I understand the cycle time so you create an issue and then at a certain point it is resolved. but I'm a bit struggling because you see the status "in progress" and "development", but development basically is also in progress. So I do not exactly understand what the "in progress" means. Does that mean it is ready to refine and refined and it is in QA? I'm not so sure. So this probably needs some more finetuning than just seeing it's in development or blocked. I probably want to see the other states and I think that currently the other states are summarized in progress. I think then you can get more out of this because if you get the ready to refine and the refined it also gives you a bit of insights in how long is a user story actually refined before we are picking up and that gives insights in are we just refining before a sprint or do we have more time to refine our stuff, for example. So I suggest adding more details here if possible. If we understand what we have to do in Jira, for example to make sure that you get the right KPIs in, it would be beneficial. Because I have

three teams where I can see the right measures for individual velocity, but when I go into the fourth team, the velocity of all these guys is zero. They work with story points so that is not true. They are doing something different and that's crossing the fact that some KPIs are not working. So it might be useful to describe how and what they need to change in the process so that everyone is able to get the same KPIs.

Student: It can be how the issues are closed because I know if they're done it will count the story point but if it's in delivery at the end of the sprint or some other status it might not count the story point.

Engineering manager: The Change failure rate is also very useful. And if this is the truth, then the number is terrible. There are bugs in production and total issues in production. Yeah, I definitely need to understand this one better than I do at the moment. I had a couple of wishes myself which was the cycle time indeed. not sure if this one is close to the first time yield or not (the Change failure rate KPI).

Student: Yeah. it could be seen like that. they're not exactly the same though.

Engineering manager: If you can share the document again, then I can have a check and I really would like to understand when reading this. But I think I understand how to read it now. the red one is indeed the number of issues and this is the percentage and this line is connected to this percentage. But that is a useful KPI I think for every team and then you need to define what is deployed in production because different teams have different views there. But it is indeed pretty close to the first time yield that I had in mind.

Engineering manager: I think these are very useful dashboards. As soon as we have the proper data in and the proper understanding and the proper reading of the dashboard, that's definitely important, especially when you look into the individual ones. But it's good to see this.

Appendix 16: Protocol of MVP 1 Test with Software Engineer

Date: 30-12-2024

Duration: 10 min

Location: Google Meet

Participants: Software engineer & Laura-Liisa Lillemäe (Student)

Student: This will be in the form of thinking aloud, I will be as quiet as possible and you will have the chance to say what you think about the dashboard that you have tested.

Software engineer: I don't really understand the dashboard. Velocity per sprint does not say something that I would like to look at often. Maybe in the beginning of the sprint to see that it is stable. I haven't used the dashboard frequently everyday. Especially now when everyone is on vacation and I'm the only one here. I do other things outside of tasks assigned to me, so no data is gathered about me for me to look at.

Student: Is it that you don't understand how to read the dashboard or do you not find it useful?

Software engineer: I Don't find it useful. I think this is more useful for the management.

Student: We also have other KPIs like Cycle time by status , for the team and for yourself, and Change failure rate for example, what do you think of the other KPIs?

Software engineer: I don't find it useful, the correlation between story points and time maybe i understand yes, but everything else, not useful for me.

Student: What do you think about the graphics?

Software engineer: The graphics are ok. But I don't understand the Change failure rate, what is the percentage of? In correlation with what? Is it a dip/rise from the previous month or what?

Student: *explains how to read the graph and what can be seen through it*.

Student: Is anything missing from the dashboard, according to you?

Software engineer: Not sure if anything is missing.

Student: Is there anything that you would like to add?

Software engineer: No, not this time at least.

Appendix 17: Protocol of MVP 1 Test with Team lead A

Date: 30-12-2024

Duration: 20 min

Location: On site

Participants: Team lead A & Laura-Liisa Lillemäe (Student)

Student: So this will be in the form of thinking aloud, I will talk as little as possible and you will just tell me your thoughts on the dashboard and maybe I will ask a few questions if needed later.

Team lead: So first thing I thought was I want to make the data better like feeding into the dashboards because now we have the tools which we didn't have before but we need to improve the data coming in. The more interesting part for me is how do we make this individual velocity better. One approach is to have smaller tasks like less story points for each issue because that means that the developers can clear many tasks in one sprint instead of maybe having one eight pointer and then maybe it takes two sprints for something because something happened along the way then if you have three pointers instead then they will tick off more tasks which will give better data. So if you take a look at the average per sprint for the last period and then what happened last sprint if you just have one task eight points and they didn't close it then it's just zero points for that sprint but if you would have three tasks. At least they would have been able to close two and that would be more reasonable. The KPIs are good because these are driving us to improve our way of working as well not just the actual performance but the processes but because now we can see these graphs and figure out how do we make our way of work maximize kind of the graph output but because the graph output is also a quality insurance. It's kind of driving us to think about how can we improve our process. So that's one. And then second of all, it's going to take a little bit of time to learn how to use them. So don't be in a rush in the beginning just use them and try to kind of play around with them and along the way you will get actually better with them. I started thinking about for example let's take this individual velocity one it would be interesting to see okay so the last sprint he had three story points his average is 12 but how was it for the last three sprints? Is it a downward going trend for him or is it an upward going trend or what's going on? you need more data. And you can't put all of that in one graph because then the graph becomes too convoluted and too unspecific. Then maybe you can have one dashboard with the most important graphs, the daily drivers, and then another dashboard with more detailed ones. So if we see that one developers last sprint average is 14 but the total average of 6 sprints is 7, but we can see that for the last three sprints he's actually on 13. Then we don't have an issue.

Student: Do you have any examples for something bettering for next week if I would adjust or add or change anything?

Team lead: For example to see what is the trend? Is it going down for each sprint?

Student: Could you use the current assignee velocity chart but with the ability to choose a developer? Then you can dig deeper into specific developers and I can make it possible for you to choose.

Team lead: but I don't think this is something that you should do now because now you should focus on the ones that we have created and that's what I'm saying just by creating these now we have a bunch of questions like what's the next steps right, what can we do here and that's enough. I think just have this set because then it falls on the normal operation to just create the new ones, right? So that's if you want, but what I'm saying right now is more of a future thing, like every time you wish you could see more about this then you create a graph to complement these. Because you get ideas from all of this and it makes the set better right. Don't think about this as something just for the developer we're

going to look at this developer or this team for but think about it also in forms of how do we improve our way of working our processes based on insights we see. Then the cycle time one is also something that this is a snapshot of the current moment, but it always raises the question about what is the trend. I mean last sprint or last three sprints, during the summer time, maybe the tester was on vacation. it says something but it can be complemented with more information. So what's the trend? what's the slope? Is it going up? What's going on? That kind of data needs to also be there somewhere and those can be supporting graphs on other dashboards. They don't have to be on the main (This one). This is going from no analysis to some analysis and it reveals stuff that you didn't know about, for example the Change failure rate for the company 36% that's a catastrophe. 36% of everything you release to production has bugs in it that's completely unacceptable. So already from that graph just by formulating it, creating it, I can go to my CTO and say, look, we have a problem here and no one knew about, it at least not me. I think it will be interesting once we get the data for the individual teams as well because now we needed to activate some fields in the issue ticket, now they're there, now we need to be mindful of filling those. That's kind of what I'm saying about insights, that you start seeing these patterns which were impossible to see more or less if you couldn't visualize them like we are doing now. So it's a massive improvement I would say compared to when you don't have any analytics.

Team lead: It was a little bit silly everything is gut feeling and everyone can just talk away things. But here it's really precise. I mean you're telling me in this one-on-one that you want to move towards a senior developer role or something like that or you want more responsibility, then I think we can set up some goals with KPIs. You should aim for improving your velocity with five story points per sprint, let's say, together with maybe learnings some new skills, which will force you to kind of look over how you work like your way of working and maybe find optimizations. After one month, you have a follow-up one-on-one with the guy and then he has an upward going trend. That's why I'm saying the trend is important. Then you say, "Yeah, good job. You're doing right. Continue on that path or no change or even maybe a decline then what's going on? I'm a little bit stressed about this," you can set up meaningful goals which are accurate and then you can follow up effectively. That's the same for the team, you can have discussions with the team about what's going on and how to improve on that in a more precise manner scientifically and then follow up on that and the second one is that you can kind of spot I would say issues mostly. Maybe it's a problem when someone is doing more than they usually are, because he's stressing and you can maybe feel it in other places like in the dailies that he's stressed or he's getting frustrated or angry at other people, then maybe he thinks that he needs to compensate for someone else. And the number three is that you can communicate better with your stakeholders like the ones waiting for the work to be done. I mean you could see it just by taking a look at the amount of tasks that need to be completed and how many was actually but it was more of a gut feeling as well intuition and people could say yeah that's because these are the hard ones and now they're done so now it's only easy ones left and we will make it, but here you get more clearer picture the team is performing like this, so you feel that with these KPIs you get that support. Then you would otherwise and then you can go to the stakeholders the client delivery managers that are responsible with communicating with the customer directly and say look we're not going to make the deadline I'm telling you this one month in advance so now I'm making your life easier, because can go to the customer and you can talk with the customer and plan for this maybe they have a campaign in store like they want to push out some promotions or something and then now they know we won't be able to make it in time. It becomes a lot more smoother for the stakeholders as well if they get information from me earlier and this helps me get a more confident picture of the situation.

Team lead: The cycle time is a good one, if the development in progress time starts going up and then you take a look at the velocity and people have downward going trends there, then maybe there's some problem with that part of the code base or the project is too complicated, not well explained enough or not clear enough or you have some team spirit problem. You can kind of see it in the cycle time and then go back and even reinforce that in the velocity chart so I think these bring a good overview, they complement each other but that's not the main point here. It's not really obvious how they do that for me like some stuff does but that's the point now where do we now add graphs that complement these and expand our vision so to speak.

Student: And just if we think about the appearance is it something that the graphics that you feel should be different?

Team lead: I think it's good. I think I had in mind more complex graphing but maybe it was too complex maybe it's better to have these to the point graphs that show one or two things maximum because that makes them more effective. Yeah, no, I'm happy. and one of them, the cycle time again, we started out with just a table with text and I think it was a good investment to make it like this because a picture tells more than a thousand words. I'm looking forward to using it. You did a really good job, I have to say.

Appendix 18: Protocol of MVP 2 Test with Team lead B

Date: 03-01-2025

Duration: 15 min

Location: Google Meet

Participants: Team lead B & Laura-Liisa Lillemäe (Student)

Student: I will begin with asking if you have any new thoughts since last time?

Team lead: Yeah, I think we also already found out that we need some Jira administration to be able to set for our project the right fields values to get some data into the Change failure rate. I've also been looking a little bit at the cycle status and what kind of surprised. I expected it to be almost all in progress. Which is not surprising because our workflow is very much simplified. We have backlog selected for development, in progress and done. We have very little bit of time spent on things when they're done which may sound weird at first, but then I started thinking about it and then I thought, yeah, it makes sense. Somebody thinks it's done and they have to recheck something, so the ticket is opened again. But I expected some time also to be locked on backlog because that's when we analyze tickets and refine them. I have to look to the data for that also a little bit better. Maybe people are locking that on a generic task or it's somehow getting lost by the filters.

Student: Yes, I have other people too that experience that all the statuses are not correct in time and we are trying to figure out what the cause can be. One thing is how you move the tickets between the statuses but I don't know if that's the only cause, we need to look into it deeper.

Team lead: That's pretty much it. what I had so far in between. I don't know if I mentioned already last time for the velocity thing it would be nice if you can currently see the current and an average.

Student: Yes, I have added a new KPI for that so you can configure it to your dashboard. It's in the team success KPI map in EazyBI. It is called Velocity by Assignee - detailed. and there you can choose whatever team-member you want to look closer at.

Team lead: Okay, I will look into adding that one.

Student: Then you can let me know if it is what you expected or what you wished, if that helps the problem. But overall has anything been confusing or unexpected while using the dashboard?

Team lead: Not really actually. I think, and that's my own confusion, about the Jira issue terminology. When you look at the Change failure rate what is exactly issues in production and bugs in production, how do they relate? Because for us, all Jira issues go to production and are deployed when they are marked as done.

Student: Could the confusion be in how I have named the components in the report?

Team lead: I think it could help a little bit as in the naming but because I think in production basically means it's set to done right? Because we have bugs in production and total issues in production. And bugs in production are the bugs found in production, right? But the total issues, kind of confuses me in the sense that is it really the total issues or the total issues that are deployed or completed?

Student: *explains what the components mean*. Do you feel that you would get any use of this KPI, Change failure rate?

Team lead: I am really interested to see the relationship between how much issues we complete and how much bugs we have. so that would be really interesting to see. Overall I think the dashboard is pretty straightforward, and the layout, you can drag it a little bit around yourself, which is good. To be honest, for me it doesn't also matter that much because I'm just looking at something and I want to see one of the graphs or related to another and then I'm just picking that one or two. I'm just not smart enough to look at all of them at the same time.

Appendix 19: Protocol of MVP 2 Test with Software Engineer

Date: 03-01-2025

Duration: 2 min

Location: Google Meet

Participants: Software engineer & Laura-Liisa Lillemäe (Student)

Student: I will begin with asking what are the new thoughts you have since last time?

Software engineer: This time for sure I don't have any.

Student: Have you engaged with the dashboard at all or looked at it since last time?

Software engineer: No, because this week was not so productive. Firstly I'm the only one working since last time we have met. But in two weeks maybe I'll have some feedback. Because everybody's back and maybe we'll have some discussions about it. Maybe you have some guidance on what I should look into and what should I follow.

Student: If you don't find use in the dashboard, that is totally okay. So, you can maybe focus on the layout, or is the information and the names of things clear in the dashboard? Do you understand everything? Could the graphics be different? You can just focus on those stuff on the appearance and the understandability of the dashboard.

Software engineer: Mhm. Okay.

Student: Thank you for this time and see you next week.

Appendix 20: Protocol of MVP 2 Test with Senior Frontend Developer

Date: 03-01-2025

Duration: 10 min

Location: Google Meet

Participants: Senior frontend developer & Laura-Liisa Lillemäe (Student)

Student: I will begin with asking what are the new thoughts you have since last time?

Senior frontend developer: This time I managed to save the filters. The floppy disc button. Other than that, now I have no data in the, bug section, the Change failure rate for my products, but as mentioned I don't see the benefit for having this graph. At least we already discussed about it if we can map the bugs to the tickets then it might have a benefit. Coming back to the Cycle time by status, in here I don't think the data is accurate, and I can tell you why. We touched the topic in our last discussion but I will open it once again. Keep in mind that our sprints are two weeks, right? Which means less than 10 working days. A ticket is created, ready to refine and basically there is a queue of tickets and once a week or even twice a week, depending of how big the queue is, we have meetings where we refine those tickets so that everything is clear and we estimate them, so that whenever a ticket is refined it can be taken to one of the following sprints. So, what I'm trying to say is that a ticket spend an amount of time in the 'ready to refine' and 'refine' statuses and then the actual sprint has the statuses that are in here. Development, blockes, closed, review etc. All of those one, two, three, four, five, six are the statuses that are in the sprint. So basically what I'm trying to say is that a ticket, its life cycle should be at least 10 working days because a ticket should be in a sprint, and on top of those 10 working days, I guess every ticket will have another number of at least hours if not weeks where it's staying ready to refine and refined. So simple math is telling me that those numbers should be more than at least 10 working days. What I can see I have only three hours which does not make sense to me. So this is one of the questions. I mean I'm not able to read some charts in such a way that it makes sense to me without proper data. That's one thing. I have another observation about the velocity. maybe I am mistaken but velocity means the availability for the next sprint somehow. If let's say that my velocity is 10 for a sprint and i have worked 10 working days. But if I will take for the next sprint a week off, then my velocity will be impacted. What I'm trying to say is that if I look here in this chart, I would like to see an average of story points compared to the actual velocity because this is what velocity means at least for me but maybe I'm wrong again but what I'm trying to say is that it's very counterintuitive to compare this value when I was off for more than one week let's say.

Student: Just to clarify, so average of what specific time do you mean that you want to see?

Senior frontend developer: working days divided by the delivered story points.

Student: That's a really good idea, I'll look into that. Has anything been confusing or unclear to you regarding the dashboard?

Senior frontend developer: I think that we should try to change a little bit the things like Cycle time by status so that we can properly use the data in here. But I still think that this can be very helpful to us.

Student: So it's useful KPIs?

Senior frontend developer: Yeah. It's a matter of details in my perspective, finetunings.

Appendix 21: Protocol of MVP 2 Test with Engineering Manager

Date: 06-01-2025

Duration: 10 min

Location: Google Meet

Participants: Engineering manager & Laura-Liisa Lillemäe (Student)

Engineering manager: Maybe you can begin with presenting the changes.

Student: I have added a chart and it should be looked at in comparison to the Team Velocity by Assignee. I got a request that the team leads want to see more sprints, not just the last sprint. So if you see an average and last sprint, and you want to look deeper, you have this chart. So here is all the last sprints and you can choose who you want to look into. I also made it possible to choose timeframe in the Change failure rate. Before it was just six months and now you can choose by year or by month. So you can click by quarter and then by month and then by days.

Engineering manager: And you can almost zoom into the last sprint.

Student: Yes.

Engineering manager: The grouping of different states in "In Progress" is probably too broad to provide real value. If you look at the development lifecycle in Jira, a feature progresses through various stages, such as "Ready to Refine," "Refining," "Development," "QA," and so on. It would be useful to extract and display these individual states instead of lumping everything under "In Progress." This granularity can show how much time is spent preparing versus developing, helping identify where to focus efforts to improve effectiveness. That's my main feedback for the dashboard.

Student: It depends on how teams use statuses and columns, but there are cases where teams use them correctly, capturing the stages you mentioned. However, the dashboard doesn't always reflect this properly. That's an issue I need to investigate further.

Engineering Manager: *Starts looking at the workflows of teams in Jira*. I notice some workflows don't differentiate "Development" and just use "In Progress". Consistency across projects using the same schema is essential for the dashboard to make sense. Otherwise, its usefulness is limited. That's all the feedback I have—it's already a good start.

Appendix 22: Protocol of MVP 2 Test with Team lead A

Date: 08-01-2025

Duration: 15 min

Location: Google Meet

Participants: Team lead A & Laura-Liisa Lillemäe (Student)

Student: Since last time, I have added the chart Team Velocity by Assignee - detailed, so you can see more sprints and choose who you want to focus on. It's right below this one (Team Velocity by Assignee). This was based on feedback to show more sprints instead of just the last one. I also made it possible to customize the time period for the Change failure rate. Before, it was fixed at six months, but now you can choose the time frame by year, quarter, month, or even days. Those are the changes I've made.

Team Lead: Okey.

Student: Have you had any thoughts since last time or anything you'd like to share?

Team Lead: Not much, but I think the changes you made are good. These small updates allow us to dig deeper and find more information. I've been starting my day by looking at this dashboard. Even though I haven't used it much yet, it's already giving me more information than I had before. It helps me organize my thoughts and provides more material to discuss during morning meetings with the teams. That's an improvement. I've also added something new to my workflow because of these graphs. I've scheduled one-on-ones to discuss personal growth, and I'll base many of the goals we set on this dashboard. We'll establish quantifiable milestones along the way. That's a new addition, made possible by these tools. On the downside, the data we have is a bit messy. I need to ensure that everyone using Jira and involved in projects pays attention to the new fields and moves tasks on the boards correctly. Improving data accuracy will give us better insights. It's a negative because the starting point isn't great, but it's also a positive because it shows areas where we can improve our Jira usage. That's my initial impression. I haven't used it much since I've been off, but there's more to come.

Appendix 23: Protocol of MVP 3 Test with Senior Frontend Developer

Date: 10-01-2025

Duration: 5 min

Location: Google Meet

Participants: Senior frontend developer & Laura-Liisa Lillemäe (Student)

Student: Do you have any further thoughts or feedback regarding the dashboard since we last spoke? The changes that I have made since last time is the calculation of Change failure rate KPI. Now, every ticket in the workflow that goes to "Done" is treated as deployed and in production. You don't need to use the field 'Discovered in production' for anything other than bugs in production anymore. I also have renamed the components in the chart for clarity. I also added the option to be able to choose timeframe for the Cycle time by status KPIs.

Senior frontend developer: No, I don't have any new observations or feedback to give. My feedback remains the same as what we discussed previously.

Student: Okay, I understand. Thank you for your time!

Appendix 24: Protocol of MVP 3 Test with Software Engineer

Date: 10-01-2025

Duration: 2 min

Location: Google Meet

Participants: Software engineer & Laura-Liisa Lillemäe (Student)

Student: I will present you with the changes that i have made since last time, and will kindly ask for feedback on these. I have made changes in the calculation of Change failure rate KPI. Now, every ticket in the workflow that goes to "Done" is treated as deployed and in production. You don't need to use the field 'Discovered in production' for anything other than bugs in production anymore. I also have renamed the components in the chart for clarity. Do you think that this naming (Total tickets in production) more clearly states what it stands for than 'issues'?

Software engineer: Mhm. Yes.

Student: And for Cycle time by status , I have just made it possible to choose a time frame. Before it was set to 3 months. What do you think about this change?

Software engineer: Yep. I think it's beneficial.

Student: So this is the last session and therefore I want to ask you what would be your last feedback that you wish to give?

Software engineer: Not sure what I can else say that was already said because yeah we have been in a holiday and again I'm the only one who was working and I have not yet start starting looking into live data or making something of them. Unfortunately.

Appendix 25: Protocol of MVP 3 Test with Team lead B

Date: 10-01-2025

Duration: 11 min

Location: Google Meet

Participants: Team lead B & Laura-Liisa Lillemäe (Student)

Student: I've made changes to the Change failure rate KPI. Now, every ticket in the workflow that goes to "Done" is treated as deployed and in production. You don't need to use the field 'Discovered in production' for anything other than bugs in production anymore. Does this make more sense now?

Team Lead: Yes, it does.

Student: I also renamed it to 'All tickets in production' instead of 'Total issues in production'. Do you think that's clearer, or would you suggest something else?

Team Lead: No, it's very clear.

Student: For Cycle time by status, I added the option to customize time frame by year, quarter, month, or even days.

Team Lead: Yes, I like that you can select a range. It's useful to drill down or look at a broader picture.

Student: That's it for the changes since last time. Have you found the dashboard helpful since last time?

Team Lead: Yes, the historical view for velocity is really helpful. Seeing trends over time allows me to coach individuals better. The Change failure rate KPI is also very useful. It took some effort to set everything up, but that's more about me getting familiar with the system. I like how the bugs and total issues are displayed now, it's very clear and helpful. One thing, I noticed a bug I added yesterday isn't showing up yet. Could there be a delay?

Student: If it's in EazyBI, sometimes filters need to be reapplied after adjustments to the charts. The last data import was eight hours ago.

Team Lead: Thanks. I also noticed the KPIs are more realistic now. They're still a bit high but much better than before. Previously, one out of three deployments seemed off. Now it's more accurate. That's pretty much it from my side. You've done great work. Good luck with the final steps of your reporting.

Student: Thank you so much for your input throughout this project. Your feedback has been invaluable.

Appendix 26: Protocol of MVP 3 Test with Engineering Manager

Date: 10-01-2025

Duration: 13 min

Location: Google Meet

Participants: Engineering manager & Laura-Liisa Lillemäe (Student)

Student: I've made changes to the Change failure rate KPI. Now, every ticket in the workflow that goes to "Done" is treated as deployed and in production. You don't need to use the field 'Discovered in production' for anything other than bugs in production anymore. Does this make more sense now?

Engineering Manager: I see. Yes, indeed. If I scroll down... Okay, this is all tickets in production. The percentage is the proportion of bugs, and the red bar shows the total number of bugs, correct?

Student: Exactly. The red bar is the bug count, and the yellow shows the percentage.

Engineering Manager: Got it. The highest percentage here is 7.14%. It's adjusting dynamically, right?

Student: Yes. The scale adjusts based on the values in the diagram.

Engineering Manager: Alright. So, for September, there were 175 issues in production, with a 7.14% failure rate. That's clearer now.

Student: Yes, though data is still limited as the field has only just started being used. Over time, this will improve.

Engineering Manager: I understand. Let me filter by specific projects. Okay, this looks busier in Q3 and Q4, with a peak.

Student: I've also made updates to the cycle time KPI. Before, it was fixed to three months, but now you can choose different time frames, such as yearly or monthly.

Engineering Manager: That's helpful.

Engineering Manager: This dashboard is really useful. We can now look back at projects, see what's coming back from production, and have discussions with teams on how to improve. Great work!

Student: Thank you! It's been fun working on this.

Appendix 27: Protocol of MVP 3 Test with Team lead A

Date: 10-01-2025

Duration: 5 min

Location: On site

Participants: Team lead A & Laura-Liisa Lillemäe (Student)

Student: I will start by presenting a few changes since last time. So I made some changes in the calculation. Now you don't have to use the discovered in production for all ticket types, only the bugs. So the data is instantly more correct, I think. And I also renamed it because I got feedback that the word "issues" was confusing. I meant Jira issues, but some understood it as "issues" like a problem, and then bugs and issues were very similar. So I named it 'Total tickets in production.' What do you think about that naming?

Team Lead: Yeah, I think it makes it a little bit more clear.

Student: And then for cycle time, I also made it possible to choose a time frame. Before, it was set to three months, but now you can choose a quarter, which is kind of three months' worth of days.

Team Lead: That's also really good. I think that covers what we talked about last time. With drilling into and maybe needing different graphs, but maybe you don't need them anymore because you can just use this one and change the time frames. I like it. I think the changes were good. They were needed as well. Now we at least have data here (Change failure rate). And also the first bug in production can be seen. So we have the first Change failure rate percentage for our team. That's cool to see. Yeah, I've used it, but not much. I checked it yesterday. We had a sprint end yesterday and a new sprint start. So now I can go in myself and just create a quick dashboard for sprint planning purposes. Add it. So that's a result of this work. But besides that, I don't have that much feedback today. And as I said, just going from nothing to something is good. And I think this is even a step further. This is actual useful stuff. And I like the idea that you've based it also on an existing tool set. So that way we know kind of that there is a good point in using it as well. We're not just guessing, it's based on some actual data and research.

Team Lead: I am very happy. Are you happy?

Student: Yes, I'm happy. I've had fun.

Team Lead: Have you learned a lot?

Student: Yes, really much.