



Examensarbete inom datateknik

Grundnivå, 15hp

AI's effect on Agile Environments using automated insights

**Carlos Moza
William Ramberg**

AI's effect on Agile Environments using automated insights

AI:s påverkan på agila miljöer genom automatiserade insikter

Carlos Moza
William Ramberg

Examensarbete inom datateknik
Grundnivå, 15 hp
Handledare på KTH: Lucy McCarren
Examinator: Fransico J. Peña
TRITA-CBH-GRU-2025:316

KTH
Skolan för kemi, bioteknologi och hälsa
141 52 Huddinge, Sweden

Abstract

This thesis investigates the impact of AI-driven automation on Agile project management, focusing on automating performance insights in Jira for Scrum teams at the charter travel company TUI. The goal was to reduce manual labour of the Technical team leads (TTLs) and Scrum masters by developing a tool that automatically delivers Agile metrics through AI driven insights.

The tool integrates Jira's API, TUI's internal AI and Mail API to collect sprint data, calculate key metrics and deliver structured insights via email. These insights provide objective, data-driven feedback on team performance, supporting Technical team leads and Scrum masters in monitoring trends, identifying anomalies and making educated decisions.

The effectiveness of the tool was evaluated through surveys which were answered by 14 Technical team leads. The evaluation showed clear benefits, including time savings and increased consistency in reports, as repetitive tasks were automated. While AI-generated insights proved to be a helpful tool, it was not intended to replace human judgment. Challenges include AI's limitations in understanding of team context and privacy concerns regarding the handling of personal and work data.

Keywords

Agile Project Management, Jira, Artificial Intelligence (AI), Automation, Agile Metrics, AI-Driven Reporting, Team Performance

Sammanfattning

Denna uppsats undersöker hur AI-driven automatisering kan användas inom agil projektledning, med särskilt fokus på att automatisera insikter om teamets prestationer i Jira för Scrum-teams hos TUI. I dagsläggen lägger Tekniska team ledare (TTLs) och Scrum masters mycket tid på att manuellt samla in, analysera och tolka sprintdata. En process som är både tidskrävande och svår att standardisera.

För att lösa detta utvecklades ett verktyg som integrerar Jiras API, TUIs AI-tjänst samt deras Mail API. Verktöget samlar automatiskt in sprintdata, beräknar centrala agila nyckeltal och skickar AI-genererade insikter via mejl. Dessa insikter ger objektiv, data-driven återkoppling kring teamets prestation och hjälper TTLs och Scrum masters att identifiera trender och avvikelser i arbetsflödet.

Verktygets effektivitet utvärderades genom enkäter som besvarades av 14 TTLs. Utvärderingen visar tydliga fördelar i form av tidsbesparing och ökad konsistens i rapporteringen. Även om AI-insikterna är ett värdefullt stöd, ersätter de inte mänsklig bedömning. Utmaningar som lyfts är AI:ns begränsade kontextförståelse samt integritetsaspekter kopplade till hantering av känslig data.

Nyckelord

Agil

Projektledning, Jira, Artificiell Intelligens (AI), Automatisering, Agila mätvärden, AI-baserad rapportering, Teamprestation

Preface

We want to send a special thanks to Sandra Atkins and Susanne Franke from TUI for helping us through this journey and offering their support whenever needed. Also Lucy McCarren as well as the Technical team leads at TUI who took their time to provide feedback for us to further improve our tool and thesis.

Contents

1	Introduction	1
1.1	Background	1
1.2	Problem Statement	2
1.3	Research Questions	2
1.4	Delimitations	2
1.5	Ethical Approach	3
2	Theory	5
2.1	Agile Metrics and Insight Generation	5
2.2	Limitations of Traditional Dashboards and Tools	6
2.2.1	Inconsistent Measurement Validity and Context Sensitivity	6
2.2.2	Data problems and Difficult Metrics	7
2.2.3	Cognitive Overload and Poor Usability	7
2.3	The Role of AI and Automation in Agile Insight Delivery	7
2.4	Key Metrics for Insight Generation in Agile Teams	8
3	Method	9
3.1	Methodology	9
3.2	Development Approach	10
3.2.1	Project Foundation and Tools Selection	10
3.2.2	Issue Collection via Jira API	12
3.2.3	Metrics Processing and Summarization	12
3.2.4	AI-Generated Insights Using TUI's Internal AI	15
3.3	Data Collection	16
3.4	Data Analysis	17
4	Results	19
4.1	Showcase of AI-Generated Insights	19
4.2	Survey Results	20
4.3	Perceived Usefulness	21
4.4	Efficiency and Time Savings	21
4.5	Quality of AI-Generated Insights	22
4.6	Trust and Adoption	22
4.7	Open Feedback	22
5	Discussion	25
5.1	Increased Efficiency in Agile Reporting	25
5.2	The Role of AI in Supporting, Not Replacing, Humans	25
5.3	Standardization and Knowledge Sharing	26

5.4	Limitations and Future Work	26
6	Conclusion	29
7	References	32

1 Introduction

1.1 Background

Agile project management is a crucial aspect of a company's development, as it fosters both velocity and adaptability through iterative approaches, and it is commonly utilized in software development. This methodology operates by breaking down major tasks into several smaller, manageable tasks, allowing for adjustments and refinements throughout the process rather than adhering to a rigid, linear path. This flexibility is one of the key advantages of Agile, enabling teams to respond to changes and new information swiftly and effectively [1].

There are three primary frameworks within Agile project management: Scrum, Kanban, and Lean [1]. While all three are valuable, this project will focus on Scrum, which is characterized by fixed-length project iterations known as sprints. Sprints typically last between one to four weeks, during which specific tasks and goals are pursued. At the end of each sprint, teams review their progress, reflect on what was accomplished, and plan for the next sprint. This cyclical process promotes continuous improvement and ensures that projects remain aligned with evolving requirements and stakeholder needs.

To effectively manage and track all the sprints and tasks within a sprint, it is essential to utilize a robust application [2]. This project will be conducted in collaboration with the charter travel company TUI who use Jira an Agile Project Management tool. Jira is widely adopted across various industries due to its comprehensive features that support efficient bug tracking, issue tracking, and Agile project management. It provides teams with the ability to create detailed workflows, prioritize tasks, and monitor progress in real-time, thereby enhancing overall productivity and collaboration [2].

Large Language Models (LLMs) are a type of Artificial Intelligence trained on vast amount of text data, enabling them to generate human like responses and perform complex language tasks such as summarization, pattern recognition and contextual analysis [3]. Recent industry research highlights that LLMs are being adopted in project management to streamline repetitive tasks and improve analytical precision [4]. Applying these capabilities to Agile workflows could offer substantial gains in efficiency and insight quality, particularly for roles such as TTLs and Scrum masters, whose time is often split between technical work and team coordination.

1.2 Problem Statement

In Agile teams, it is essential to regularly monitor and evaluate team performance to ensure continuous improvement. At TUI, this responsibility falls on Technical team leads (TTLs) and Scrum masters, who currently collect sprint data manually and analyse it individually based on their own interpretation and understanding.

This workflow is both time-consuming and repetitive, often requiring several hours of effort depending on the number of metrics and teams involved. As Agile projects scale, this manual process becomes increasingly inefficient and unsustainable. It also introduces the risk of inconsistencies in how data is interpreted and communicated.

To address these challenges, there is a growing need for an automated solution that can extract, analyse, and distribute sprint insights. AI has been proposed for this due to its ability to efficiently process large volumes of structured and unstructured data from tools like Jira. AI has the ability to identify patterns in sprint performance, detect anomalies, and generate summaries or actionable recommendations with minimal human intervention. This allows for more objective, scalable, and timely feedback across agile teams. The aim of this thesis is therefore to explore the capabilities of integrating AI in agile environments and its impact on workflow. By integrating such an AI-driven system with Jira, it becomes possible to enhance agile practices and support continuous improvement in a more automated and intelligent manner.

1.3 Research Questions

The scope of this thesis has been refined into the following specific research questions

- How can LLMs be used to automate Agile projects insights in Agile environments?
- How does automation impact workflow efficiency of TTLs compared to manual methods?

1.4 Delimitations

While Agile methodologies share common principles, the proposed solution is specifically designed to support Scrum-based agile teams since this is the Agile framework adopted by TUI. Although several of the researched metrics are relevant

to other Agile methods, that primary focus on tailoring the solution to fit Scrum practices. Consequently, this solution may not be fully applicable or effective for teams using other Agile methodologies, such as Kanban or Lean, which have different structures and workflows.

The developed tool is tailored for integration with Jira, a widely used project management application. Its functionalities are optimized for users within TUI who rely on Jira for their project management needs. This means that teams using other project management tools may not be able to leverage the benefits of this project's solution, as it is specifically designed to work within the Jira environment.

The lack of detailed information regarding TUI's existing AI systems, their architecture, and their data processing methods acts as another limitation of this study. Since the internal documentation and technical details were limited or not allowed to be shared, the analysis and design of the proposed solution had to be based on assumptions and general knowledge of AI-driven analytics rather than direct insight into the company's current implementation.

1.5 Ethical Approach

Implementing AI-driven automation involves handling sensitive performance data, which raises important data privacy concerns. Ensuring the security and confidentiality of this data is paramount to protect the privacy of team members and the integrity of the project which was done using TUI's internal services, meaning no third-party handled the confidential data.

The authors of the report hereby certifies that the report was produced without the aid of generative AI for purposes other than language checking.

2 Theory

This section explores existing academic and industry research on Agile metrics, the application of AI in Agile environments, and the challenges associated with the current tools. The goal is to identify what already has been researched and to establish a conceptual foundation for understanding the conditions under which AI can effectively contribute to Agile practices and decision-making processes.

2.1 Agile Metrics and Insight Generation

Agile software development emphasizes short feedback cycles, team autonomy, and iterative delivery. To support these principles, Agile teams often rely on lightweight metrics such as *velocity*, *lead time*, *cycle time*, and *throughput* to monitor progress and guide planning decisions. Visualization tools like burn-down charts, cumulative flow diagrams, and work-in-progress (WIP) trackers are commonly embedded within platforms such as Jira or Azure DevOps to help teams detect workflow issues and support sprint retrospectives [5].

These metrics are designed to improve transparency and enable teams to adjust their practices sprint by sprint. For instance, an unexpected increase in cycle time may signal emerging blockers, while a drop in throughput might point to reduced availability or scope creep. Yet, as Agile teams grow and operate in increasingly complex environments, conventional metrics show limitations. They often lack granularity, overlook contextual factors such as task complexity or cross-team dependencies, and fail to capture organizational or interpersonal dynamics [6].

Moreover, certain metrics can be misleading if taken at face value. For example, a sprint with high velocity does not necessarily indicate high productivity; it may conceal issues such as burnout, declining quality, or incomplete deliverables [5]. As Pham and Neumann argue, Agile metrics are prone to misinterpretation when shared definitions are missing or when metrics are decoupled from team goals. This has led to a growing interest in analyzing how work is performed, incorporating factors like communication patterns, review cycles, and bottlenecks, rather than only tracking outputs. Such process-aware and collaboration-sensitive insights lay the groundwork for AI-driven approaches that can uncover deeper meaning in Agile data [5].

Furthermore, these metrics are often manually tracked or interpreted subjectively, introducing more noise and potential bias into decision-making for the teams progression. Feng and Asad highlight this disconnect, emphasizing that traditional

productivity indicators such as the number of commits or lines of code written are insufficient proxies for actual developer effectiveness [7].

Feng and Asad's approach leverages a combination of machine learning, natural language processing (NLP), and predictive analytics to analyze a broad spectrum of development artifacts; including code repositories like Git, issue tracking systems like Jira, and collaboration platforms like Microsoft Teams. By correlating structured data, like ticket status or sprint cycles, with unstructured communication, such as comments or pull request reviews, their method enables the identification of emergent patterns, bottlenecks, and contextual workload estimations that are far more insightful than traditional dashboards [7].

Instead of simply measuring the number of tasks completed in a sprint, Feng and Asad concludes that AI can assess which tasks were most impactful, how long they were blocked, who was involved, and why delays occurred. These insights are crucial for team leads, who often need to distinguish between velocity as a quantitative metric and actual productivity or team health as qualitative indicators. Moreover, the ability to forecast potential issues based on historical trends enables more proactive sprint planning, as teams can anticipate risks, balance workloads, and adjust their backlog strategies accordingly [7].

2.2 Limitations of Traditional Dashboards and Tools

Commercial tools such as LinearB¹, ActionableAgile² and SmartBear Zephyr³ are designed to automate Agile metrics tracking and support planning and review. However, empirical research within academic and industry literature, which is covered in this subsection, reveals key limitations affecting their interpretability, relevance, and usability, especially for Agile leadership roles like TTLs, but also for the rest of the Agile Teams. Exploring previous tools to find their limitations, builds a solid foundation for how to properly integrate AI in future projects.

2.2.1 Inconsistent Measurement Validity and Context Sensitivity

Chronis and Gren conducted a validation study across three different Agile assessment tools and found that different tools gave different results; similar constructs within the tools yielded inconsistent results (e.g., teams gave different responses to equivalently phrased items) [8]. This inconsistency raises significant

¹<https://linearb.io/>

²<https://actionableagile.com/>

³<https://smartbear.com/test-management/zephyr/>

concerns about the meaningfulness of metrics when tools treat different team contexts the same. TTLs may struggle to trust outputs that vary across tools or do not accurately reflect their team's practices.

2.2.2 Data problems and Difficult Metrics

Ram et.al showed that many teams struggle to gather the right data or interpret it in a useful way. In their case study, teams reported that they often couldn't access the needed information or didn't agree on what metrics like "lead time" really meant in their context. Because of this, teams either avoided using the tools or created their own informal ways of tracking progress [9].

2.2.3 Cognitive Overload and Poor Usability

The dashboards used in Agile teams can in some cases be cognitively demanding. While they provide a large volume of data, the relevance of this information is not always clear. The sheer number of charts, alerts, and Key Performance Indexes can overwhelm users, particularly during time-constrained meetings like sprint reviews or daily stand-ups. In such situations, teams often fall back on intuition or anecdotal information, rather than relying on analytics that require deep interpretation [10].

2.3 The Role of AI and Automation in Agile Insight Delivery

To overcome the limitations of traditional dashboards and tools, researchers and tool developers are increasingly utilizing AI to enhance the way Agile Insights are delivered. Rather than simply presenting raw data, AI systems can identify patterns, detect anomalies, and generate higher-level interpretations of team behavior [11].

One notable example is SonarQube⁴, a commercial platform that uses automated tracking to monitor engineering metrics like cycle time, deployment frequency, and pull requests. It provides real-time alerts and recommendations to teams when delivery risks are detected. While SonarQube improves visibility, it remains limited in its interpretability and cannot adapt insights based on team-specific dynamics [5].

Research by Ferrao, Miranda and Soler introduces a more sophisticated approach, their LLM-powered system processes GitHub contribution data and produces

⁴<https://www.sonarsource.com/products/sonarqube/>

natural language summaries of team activity [12]. The tool was found to be especially helpful for managers and instructors overseeing multiple teams, as it abstracted away noisy, detailed logs in favor of concise overviews. This showcases the intended function of AI-generated sprint insights covered in this thesis; providing quick, context-aware summaries of team performance.

Another dimension of AI integration in Agile environments is the use of LLMs to automate and maintain technical documentation. Birru introduces a system called Autodoc, which employs Retrieval-Augmented Generation (RAG) to update documentation in response to source code changes [13]. This solution reduces the manual burden on developers and technical writers by aligning documentation with evolving software artifacts, particularly within Docs-as-Code pipelines. Although focused on documentation, the approach, combining prompt engineering, vector embeddings, and contextual retrieval, offers a solid foundation for automating Agile sprint insights. By retrieving relevant data (e.g., Jira issues or sprint history) and generating concise, human-readable outputs, LLMs can support Agile roles traditionally reliant on manual information processing [13].

2.4 Key Metrics for Insight Generation in Agile Teams

To support AI-generated sprint digests, it is important to identify which performance metrics provide actionable and relevant insights for Agile teams. While traditional indicators such as velocity and throughput remain useful for overall tracking, they often fail to capture deeper process dynamics [15]. Studies has highlighted a set of metrics that offer more informative and context-sensitive insights. Among these, cycle time stands out as a crucial indicator of workflow efficiency, allowing teams to detect bottlenecks or delays in specific stages like development or review [15].

Other valuable metrics include velocity, which when interpreted carefully can reveal workload imbalances or risks of overcommitment, and change failure rate, which reflects the stability and quality of deployed work. Lillemäe emphasizes that combining a larger set of metrics provides a more comprehensive view of effectiveness and efficiency [15].

Other sources identifies work-in-progress (WIP), queue time, and throughput as common Agile metrics, used to track flow and diagnose inefficiencies. Similarly, DevOps-aligned metrics such as deployment frequency and mean time to recovery (MTTR) can be adapted to be included in sprint-level insights [17]. The metrics used in this report are specified in Figure 3.3.

3 Method

3.1 Methodology

This study follows a practical and applied research approach, focused on designing, building, and evaluating a tool that helps TTLs and Scrum masters gain insights into team performance. The research is structured around the development of a functional proof of concept, which will be tested in a real-world Agile environment and evaluated based on both qualitative and quantitative feedback.

The approach is divided into three main phases:

Development Phase – The tool will be developed using Visual Studio Code, Jira’s API, and TUI’s internal services, including their AI and Mail APIs. This setup enables the automated collection, analysis, and delivery of Agile team metrics.

Testing Phase - Once the tool is operational, it will be introduced to a few Agile teams within the organization. TTLs and Scrum masters will use it as part of their regular monthly review process, allowing us to observe its performance and usefulness in a real work setting.

Evaluation Phase – Feedback will be collected through structured surveys designed to measure time saved, usefulness of insights, and areas for improvement. This feedback will be compared with traditional methods of gathering and analyzing metrics.

In addition to these practical steps, a review of relevant literature, including previous theses, blogs, and technical documentation, has been conducted to help guide the development process and support the research with existing knowledge. This combined approach allows for a well-rounded evaluation of the tool’s value and potential impact in Agile team environments as well as how useful AI is in an agile environment.

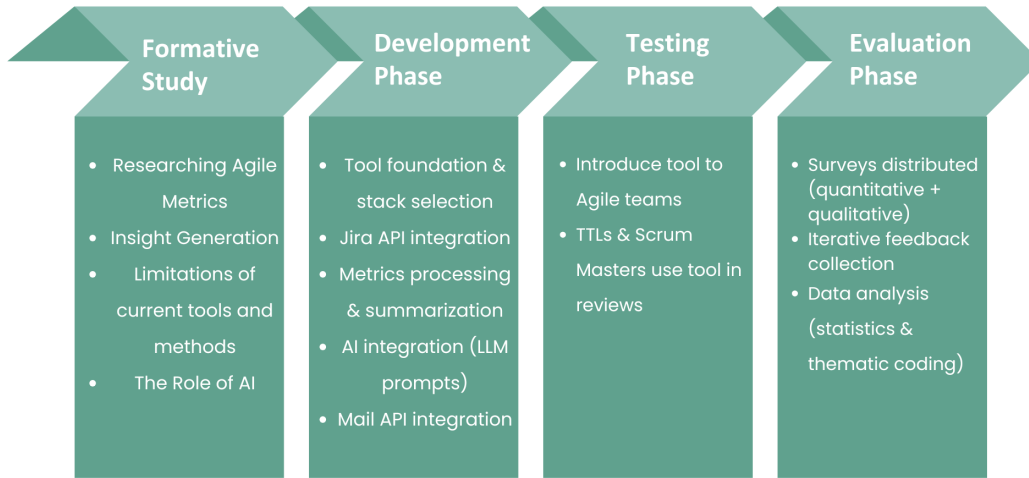


Figure 3.1: Methodology workflow showing the phases of development, testing, and evaluation in this study.

Figure 3.1 provides an overview of the aforementioned methodology applied in this thesis. It illustrates how the study was structured into three phases that together form a complete research workflow. The Development Phase shows the technical construction of the tool, where different services were integrated step by step to form a functional proof of concept. The Testing Phase highlights how the tool was applied in a real-world Agile setting, enabling observations of its practical use by TTLs and Scrum masters. Finally, the Evaluation Phase emphasizes how both quantitative and qualitative feedback were collected and analyzed, providing the basis for answering the research questions.

3.2 Development Approach

3.2.1 Project Foundation and Tools Selection

Developing an automated Agile Insight tool required a robust technology stack capable of integrating multiple APIs while maintaining efficiency, scalability and security. Visual Studio Code was chosen as the development environment, allowing us to build, test, and debug the solution within a structured workspace. The implementation was centered around a Node.js-based application that would serve as the core processing unit for collecting, analyzing, and delivering insights. To

achieve the desired automation, three different services were used within TUI's infrastructure:

- **Jira API:** Used for retrieving real-time Agile sprint data which are used to process agile metrics.
- **TUI's AI:** A large language model responsible for analyzing the metrics and generating structured insights.
- **TUI's Mail API:** Used for delivering finalized insights to TTLs and Scrum masters in HTML-formatted emails.

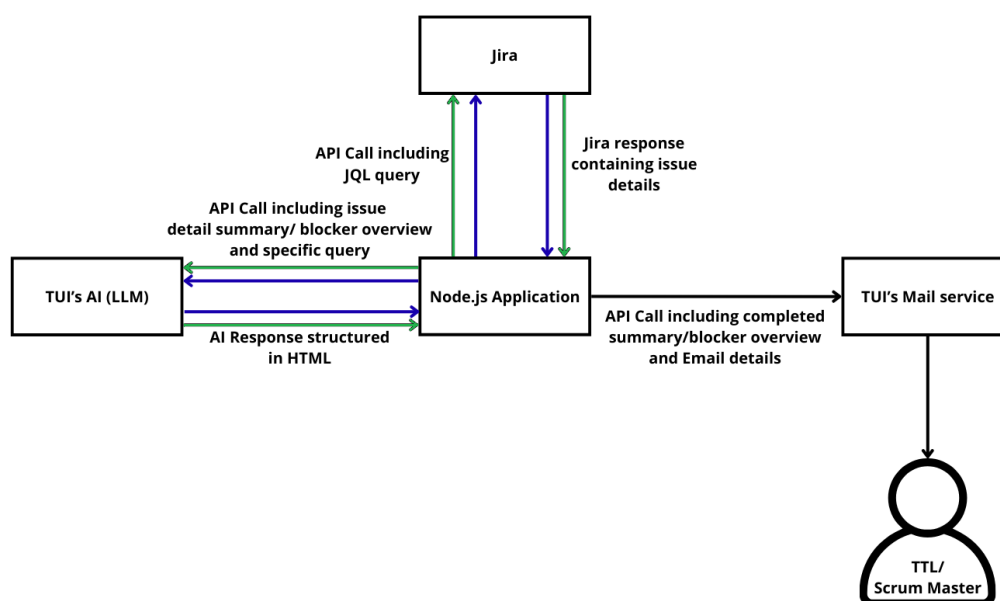


Figure 3.2: System architecture of the developed tool, showing integration between Node.js, Jira API, internal AI API (LLM), Mail API and the recipient TTL/Scrum master.

Each service communicates with the others to complete the automated processes, ensuring that Agile data is collected, interpreted, and delivered efficiently without manual effort from TTLs. Figure 3.1 illustrates the complete flow of these services. The tool itself was divided into two parts: one summarizing issue completion within a given period, and another identifying dependencies to detect blockers linked to a team's Jira issues. Despite these different focus areas, both parts followed the same process shown in Figure 3.1. The flow was simply executed twice with different queries, once for the detail summary (blue arrow) and once for the blocker overview (green arrow). In contrast, the mail service required only a single API call, which is why it is represented by a single black arrow in the diagram.

3.2.2 Issue Collection via Jira API

As illustrated in Figure 3.1, the first step in the workflow was retrieving Agile project data from Jira. Jira's REST API provides a structured method for extracting relevant issue details using Jira Query Language (JQL). This enabled precise filtering of Agile data based on various parameters, such as:

- Project ID and team specifications.
- Issue creation and resolution timestamps
- Labels associated with Agile tracking (e.g., blockers, priorities, dependencies)

Data collection was executed using batch processing, ensuring that API requests were efficiently handled, without causing system strain. The results from Jira were returned as JSON objects, containing detailed attributes for each issue. These structured objects provided raw data that could then be parsed and transformed into meaningful insights. To prevent unnecessary processing delays, queries were designed to extract only the most relevant sprint details, minimizing overhead while ensuring comprehensive analysis. For example, there are two different API calls for the issue details used for the summary and the issue details used for the blocker overview.

3.2.3 Metrics Processing and Summarization

Once the Agile issues were retrieved from Jira, the next step was the structured processing and summarization of those issues into agile metrics. This was done using a dedicated function which was designed to convert raw JSON responses from the Jira API into metrics grouped by issue type (Bug or Story), resolution status, time in workflow stages, and temporal performance windows.

The summarization logic operated in two modes: long-term analysis and a monthly analysis, which was made specifically for calendar months. In both modes, the tool evaluated the total number of issues, completion counts, and assignment ratios, alongside temporal metrics such as resolution time and time spent in specific workflow statuses.

To ensure precise calculations, each issue's timestamps were processed using the `moment.js` library. The script determined time-in-status metrics by parsing the issue changelogs to reconstruct each issue's journey through Jira's workflow states. These raw states were then normalized using a canonical mapping to handle inconsistencies across teams and projects. For example, statuses like "In Dev" and "In Development" were aggregated under the same logical label to ensure cross-team

comparability.

The following were the main components of the summarized insights:

1. The insights focused on several key metrics to provide a comprehensive view of sprint performance. First, completion metrics were used to track the total number of issues during a given period, how many of those were resolved, and the overall resolution rate. To better understand efficiency, time metrics captured the duration (in hours) from issue creation to resolution. These durations were analyzed both globally and for each sprint, with minimum, maximum, and average times calculated.
2. To identify potential bottlenecks in the workflow, status metrics were introduced. These measured the average time issues spent in each normalized status, such as “In Development” or “Ready for Testing”, enabling the system to flag phases that might be unnecessarily prolonged. Assignment metrics provided insight into task distribution, with a particular emphasis on distinguishing assigned versus unassigned issues. Special attention was given to unassigned bugs that remained unresolved, as these could indicate gaps in ownership or triage processes.
3. Lastly, age analysis was performed on unresolved issues. By calculating the average, minimum, and maximum age (in hours) of these tasks, and breaking them down by status category, the system was able to highlight potential backlog staleness or neglected work items that might require prioritization.

Summary

```

|--- totalIssues (Work In Progress)
|--- completedIssues (Bug and Story Completion)
|--- statusDistribution
|--- assignment (Assigned vs Unassigned)
|     |--- unassignedIssues
|     |--- assignedIssues
|--- timeSpent (Velocity)
|     |--- totalIssuesWithTime
|     |--- totalTimeSpent
|     |--- minTimeSpent
|     |--- maxTimeSpent
|     |--- averageTimeSpent
|--- timeInStatuses (Cycle Time)
|--- timeInStatusesIssueCount
|--- periodMetrics
|     |--- createdInPeriod
|     |--- resolvedInPeriod
|     |--- resolutionRate
|--- periodResolutionMetrics (Lead Time)
|     |--- totalIssuesWithTime
|     |--- totalTimeSpent
|     |--- minTimeSpent
|     |--- maxTimeSpent
|     |--- averageTimeSpent
|     |--- timeInStatuses

```

Figure 3.3: Textual representation of the summary data structure

All results were grouped into the summary structure which is illustrated in Figure 3.3. The output of this summarization process was then used as input for both the AI analysis and the graphical reports. It ensured that all further insights and suggestions were based on a structured set of Agile metrics. Which metrics were used are specified in Figure 3.3. Notably, by including not just summary statistics but also distributions and status-specific breakdowns, the tool allowed TTLs and Scrum masters to focus on specific points of worry or inefficiencies within the team's delivery pipeline.

3.2.4 AI-Generated Insights Using TUI's Internal AI

The AI integration was a key component of the automation process. Once Agile data had been transformed into structured metrics, a request was sent to TUI's internal AI, an advanced LLM designed to analyze Agile workflows. The AI request included the full detailed list of past sprint performance. Including both the historical and specific period data, which allowed the AI to analyze the new data based on previous values.

The request also included a query which specified what the LLM should analyze and key insights which it could focus on to draw its conclusions. To optimize these queries, a structured approach was applied to define both the scope of analysis and the interpretative framework for AI-generated responses. During the development process, it was necessary to continuously refine the LLM prompts to accommodate new metrics and evolving analytical requirements. The prompt configuration aimed to encapsulate all desired analytical dimensions while preserving the open-ended nature of AI processing. An open-ended query format was essential to allow the LLM to independently detect patterns, identify inconsistencies, and generate recommendations beyond predefined parameters. To improve analytical precision and usability, two distinct LLM configurations were implemented, each designed to serve a specific function within the automated insights system; one which focused on the summary analysis and the other focused on the blocker overview.

To generate a structured summary of Agile sprint data, a query was designed to examine multiple performance indicators across both short-term and long-term metrics. The query for the summary analysis was structured as follows:

“Based on the following project summary, including the time spent in each status for bugs and stories, as well as the minimum, maximum, and average resolution times, and the age analysis of issues, what actions should we take to improve our development process, reduce the number of bugs, and optimize time spent in each stage? Please provide specific recommendations based on the time distribution across statuses, the range of resolution times, and the age of issues in different statuses.”

This query ensured that AI-generated insights remained comprehensive while preserving adaptability in identifying performance trends and optimization strategies.

For the AI to properly structure an overview of blockers and dependencies, an additional query was designed used for categorizing and creating a structured overview of analyzed data. The query was formulated as follows:

“Please categorize the following Jira dependencies into common categories (e.g.,

'Waiting for external dependency', 'Technical debt', 'Resource constraint', etc.). For each category, provide a percentage of how many dependencies fall into that category. Also make an analysis of the dependencies and give key insights, for example if there is any common dependencies that blocks multiple issues, Circular issues etc. Draw conclusions based on major inconsistencies compared to the historical data, and give recommendations to what steps can be used to improve the teams output"

This query enabled the LLM to systematically assess dependencies, ensuring objective classification while allowing for anomaly detection beyond predefined categories.

The LLMs also needed to follow specific rules and instructions to generate responses in a structured and useful format. These rules were embedded within the LLM to act as behavioral constraints, guiding the LLM's tone, analytical scope, and formatting style. The goal was to impersonate an experienced Scrum master so that the output could be directly included in the automated insights.

The AI was told to be wary of suspicious metrics, such as unrealistically short resolution times, and to flag them as potentially inaccurate without jumping to conclusions. Another rule instructed the LLM to focus primarily on monthly performance data, using historical trends only as a comparative baseline. This ensured that the insights remained timely and actionable, rather than rooted in outdated behaviors. Formatting consistency was also enforced through the rules. The AI was instructed to structure its output using HTML and follow a clear analytical flow: starting with insights, then what went well, and finally outlining specific actions the team could take. This made the content ready for immediate use in emails or dashboards, without requiring manual editing.

3.3 Data Collection

To objectively assess the tool's potential efficiency and usefulness, structured surveys were distributed to TTLs. Appendix A showcases the survey which primarily consisted of rating scales and close-ended questions to ensure quantifiable data collection, enabling an organized evaluation of the tool's impact in Agile environments. Additionally, open-ended questions were included to gather qualitative insights that complement the numerical assessments, allowing for a more nuanced understanding of user experiences.

The survey design combined both quantitative and qualitative elements. Initially, 14 participants were asked to rate the tool's overall usefulness on a 1–5 scale where 5 is best and 1 is worst, enabling standardized comparison of user perceptions. Following

this, respondents were invited to provide open-ended feedback on specific aspects they found valuable, as well as areas where improvements might be necessary. This format encouraged diverse perspectives, capturing insights that might not emerge from numerical ratings alone.

To evaluate the tool's perceived impact on workflow efficiency, participants were also asked whether they believed the tool contributed to time savings. If they answered affirmatively, they were prompted to estimate the time saved. Conversely, those who did not perceive such benefits were asked to elaborate on their reasoning, providing contextual insights for future refinement.

A key component of the tool is its AI-driven functionality. Participants were therefore asked to rate the usefulness of the AI's analytical outputs, recommendations, and overall responses using a 1–5 scale. To supplement these ratings, they were encouraged to describe specific strengths or areas for improvement regarding the AI-generated insights.

In addition to the formal survey, an iterative feedback loop was implemented throughout the project's development process. Early versions of the tool were shared with TTLs, allowing them to test its functionalities and provide continuous feedback. This iterative engagement enabled incremental improvements based on real-world use, ensuring that the tool evolved to better meet user needs before the formal evaluation process.

3.4 Data Analysis

Quantitative survey data were analyzed using descriptive statistics. For each evaluation dimension, the mean score was calculated to capture central tendency, while the distribution of responses (the count of participants selecting each rating from 1–5) illustrated the spread of opinions. Additionally, the sample standard deviation (SD) was calculated to quantify the degree of variation around the mean. A low SD indicates that participants responded consistently, whereas a higher SD signals more divergent opinions.

The sample SD was calculated as:

$$s = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n - 1}} \quad (1)$$

Where:

x_i = each participant's score

\bar{x} = the mean score

n = number of participants

Open-ended responses were examined thematically to identify recurring patterns related to functionality, usability, and suggested improvements. This combined analysis offered both a quantitative overview and qualitative depth, ensuring that the findings reflected not only overall trends but also the contextual experiences of participants.

The continuous feedback collected during the development phase served as a valuable supplement to the formal evaluation. Many initial concerns raised by early users were addressed incrementally, as reflected in the survey data. Together, the structured survey responses and iterative feedback provided a comprehensive understanding of the tool's impact and highlighted directions for future development.

4 Results

This section presents the findings from the evaluation of the AI-driven Agile Insights tool. Data were collected through a survey of 14 participants, primarily TTIs and Scrum masters, who tested the tool in their respective Agile environments. The results are presented according to the survey themes: usefulness, efficiency, quality of insights, trust and adoption, and open feedback. An example of a generated insight is also included for context.

4.1 Showcase of AI-Generated Insights

The AI-driven system produced structured sprint digests containing key performance metrics such as sprint velocity, cycle time, blocked issues, and bug counts. These were delivered in an automated HTML email format monthly, or biweekly. Alongside raw metrics, the system generated narrative recommendations designed to highlight anomalies or trends. Figure 4.1 showcases an example of how the insight might have looked.

Key Insights from January 2025

- **Bug Resolution Rate:** 58% (14 out of 24 new bugs resolved) - showing moderate efficiency in handling new issues
- **Story Completion Rate:** 86% (42 out of 49 new stories completed) - excellent progress on feature development
- **Bug Resolution Time:** Average of 20 days 10 hours - slightly higher than the long-term average of 15 days 18 hours
- **Story Completion Time:** Average of 43 days 19 hours - slightly longer than the long-term average of 39 days 15 hours
- **Bottlenecks:** UAT (6d 19h) and Backlog (5d 22h) for bugs; Ready for Development (9d 2h) and Backlog (6d 3h) for stories
- **Unassigned Bugs:** 3 out of 30 bugs (10%) currently unassigned

What's Going Well

- **Excellent Story Completion Rate:** 86% completion rate for stories shows the team is delivering features efficiently
- **Improved Testing Efficiency:** "In Test" time for bugs has decreased to just over 1 day compared to the long-term average of 18.5 hours
- **Consistent Analysis Time:** "In Analysis" time for bugs has remained relatively stable at around 10-12 hours
- **Active Development:** The team created 49 new stories in January, showing strong product development momentum
- **Reduced Backlog Time:** Stories spent less time in backlog in January (6d 3h) compared to the long-term average (8d 11h)

Areas for Improvement & Recommended Actions

1. Reduce UAT Time for Bugs

Bugs are spending nearly 7 days in UAT, which is significantly higher than the long-term average of 3 days 20 hours.

- Schedule more frequent UAT sessions with stakeholders
- Consider implementing a "bug bash" day once every two weeks
- Improve pre-UAT quality checks to reduce rejection rates

2. Address Ready for Development Bottleneck

Stories are spending over 9 days in "Ready for Development" status, indicating potential capacity or prioritization issues.

- Review team capacity and workload distribution
- Implement a WIP (Work In Progress) limit for each development phase
- Consider pair programming for complex stories to reduce wait times

Figure 4.1: Example of AI-generated performance summary and recommendations sent via email.

Figure 4.1 illustrates the type of output participants evaluated, although the results are dynamic and covers different areas for improvement based on each teams data. Several respondents reported that these recommendations aligned well with how they normally structured retrospectives, and thus could be incorporated with little additional effort.

4.2 Survey Results

Table 4.1 summarizes both the central tendency (mean scores) and variation (standard deviations and distributions of responses). The distribution column shows how many participants selected each point on the 1–5 scale, while the SD reflects how tightly clustered these responses were around the mean. In this study, all SD values were relatively low (0.36–0.52), indicating strong agreement among participants.

Table 4.1: Mean value of survey responses and standard deviation across evaluation dimensions (N = 14)

Dimension	Mean	SD
Perceived usefulness	4.86	0.36
Workflow efficiency improvement	4.79	0.43
Clarity of AI-generated insights	5.00	0.00
Actionability of recommendations	4.50	0.52
Trust in the tool	4.57	0.51

4.3 Perceived Usefulness

All participants rated the tool positively in terms of usefulness, with an overall average score of 4.86 out of 5. A majority (12 participants) rated it as “extremely useful,” while 2 participants gave a rating of 4.

Respondents emphasized the benefit of having a consolidated overview of sprint performance available in a single report. Many participants described that previously they needed to gather metrics from multiple Jira dashboards or rely on manual exports, whereas the automated digest reduced this overhead. Several respondents further noted that the insights were “immediately actionable,” particularly when used to set the agenda for retrospective discussions.

4.4 Efficiency and Time Savings

Efficiency gains were one of the clearest outcomes reported. Twelve out of fourteen participants (86%) stated that the tool saved them measurable time compared to manual reporting practices. The estimated time savings varied between 30 minutes and two hours per sprint, while two participants reported saving over two hours.

The survey item “The tool improved my workflow efficiency” received an average rating of 4.79 out of 5, with no respondent scoring it lower than 4. Respondents frequently linked these savings to the automation of tasks such as compiling Jira data, exporting metrics, or manually prompting AI tools.

A Technical team lead explained:

“Instead of spending over an hour preparing reports, I now spend 10 minutes validating the automated insights. The time saved is used for actual problem-solving.”

4.5 Quality of AI-Generated Insights

The perceived quality of the AI-generated insights was evaluated through clarity and actionability. The average rating for clarity was 5/5, while actionability received 4.5/5.

Respondents consistently agreed that the digest format mirrored retrospective structures they were already accustomed to (e.g., “what went well,” “what to improve,” “suggested actions”), which reduced friction in adopting the tool. The action-oriented phrasing of the insights was particularly appreciated, as it allowed teams to transition directly from reading the digest into planning improvement actions.

No major inaccuracies were reported. However, three participants pointed out that the AI occasionally flagged predictable changes (such as reduced activity during holiday periods) as anomalies. While these did not undermine trust in the tool, they highlighted the importance of human interpretation alongside automated analysis. Despite the mismanaged predictable changes there was an overall positive review of the AI-Generated insights. One TTL noted that the metrics showcased great information, but the AIs input made it easier to catch anomalies which could be overlooked otherwise.

4.6 Trust and Adoption

Trust in the tool was reported as generally high. The survey results showed an average trust rating of roughly 4.57 out of 5. All participants stressed that they regarded the tool as a support system rather than a replacement for human judgment, which aligned with the intended design of the system.

When asked whether they would recommend the tool to other Agile teams, 13 participants answered “Yes,” and one answered “Maybe.” No participant responded negatively.

4.7 Open Feedback

The qualitative responses contained several recurring themes:

- Standardization across teams – Respondents valued that all teams received sprint digests in the same structure, which enabled easier cross-team comparison.

- Archival and long-term tracking – Several participants highlighted the value of automatically storing digests for future reference. This archival function was described as useful for quarterly reviews and knowledge transfer.
- Participants proposed several potential improvements, including:
 - Interactive elements (e.g., the ability to request insights on demand).
 - Integration with visualization tools such as Power BI or Jira dashboards.
 - Improved contextual awareness to prevent false anomaly detection during predictable low-activity periods.

5 Discussion

This section interprets the results in relation to the existing literature in order to assess the impact of AI on Agile environments. By comparing the findings with insights from the literature review, it becomes possible to evaluate how well the study aligns with, extends, or challenges previous research, and to draw broader conclusions about the role of AI in Agile practices.

5.1 Increased Efficiency in Agile Reporting

A key outcome from the survey was the reported time savings of 30 minutes to more than 2 hours per sprint. This supports earlier findings that Agile teams often face challenges related to the manual collection and interpretation of metrics [17]. Previous studies have found that manual handling of Agile project data can introduce errors and create cognitive challenges for users [10]. Performance measurement in Agile requires both efficiency and reliability, yet traditional reporting practices can consume valuable time [5]. The automation of reporting observed in this study directly addresses this issue, streamlining the process by automating repetitive tasks while ensuring more consistent results. The relatively low SD values (0.36–0.52) across survey dimensions further reinforce this finding, as they indicate that respondents consistently perceived the tool as useful and efficient, reducing the risk that results were skewed by outlier opinions

5.2 The Role of AI in Supporting, Not Replacing, Humans

Respondents consistently viewed the tool as a support mechanism rather than a replacement for human judgment. This finding aligns with previous research, which highlights that AI in Agile project management is most effective when complementing rather than substituting human expertise [16]. Similarly, practitioners often trust AI to assist in decision-making, but full replacement is hindered by context-specific knowledge that automation alone cannot provide [4]. The high trust scores in this study reflect this balance: participants valued AI-driven insights but recognized the need for contextual interpretation. For example months with a lot of vacations or holidays which slows development. TTLs mentioned the AI also has the possibility to catch anomalies and red flags which could be overlooked otherwise, which strengthens the argument that the AI insights is a great complementary tool, as long as the data is verified. Here too, the low SD values suggest that participants shared a consistent view of AI's role as a supportive

tool, strengthening confidence in the conclusion that AI complements rather than replaces human judgment

5.3 Standardization and Knowledge Sharing

Participants also emphasized the benefit of standardized reporting, which provided consistent insights across teams and sprints. Previous research has argued that inconsistency in metrics usage limits comparability between teams, reducing the value of performance measurements at an organizational level [6]. By delivering uniform digests, the tool addressed this challenge and enabled improved cross-team knowledge sharing. Other studies have suggested that real-time and standardized metrics can improve both decision-making and long-term process optimization [14]. However, over-standardization may risk overlooking unique team contexts, suggesting that future iterations of such tools should balance consistency with adaptability.

5.4 Limitations and Future Work

TTLs were asked to estimate how long it would take to construct similar insights manually. However, they concluded that replicating the AI-generated insights was not feasible, as some of the underlying calculations were too advanced to reproduce with Jira dashboards or manual methods. As a result, reported time savings should be regarded as estimations rather than precise metrics. Relying on self-reported measures may also introduce bias, since these reflect individual experiences rather than objective performance data.

Another limitation is that the AI occasionally highlighted predictable fluctuations (e.g., holidays) as anomalies. Similar challenges are discussed in previous research, who caution that AI-driven productivity tools are at risk of misinterpretation without sufficient contextual awareness [7].

In addition, the relatively small number of participants ($N = 14$) limits the representativeness of the findings. With a larger participant pool, greater variation in perspectives might have been captured, leading to a more comprehensive evaluation of the tool. The short testing period further constrained the evaluation, as TTLs had limited opportunities to assess the tool across multiple insights. Extending the evaluation period in future studies could provide a broader understanding of the tool's efficiency and practical value.

Future development should focus on:

- Increasing interactivity and enabling on-demand insights.
- Integrating with visualization platforms for richer analysis.
- Improving contextual awareness to reduce false positives.
- Scaling the solution for distributed or large-scale Agile environments.

6 Conclusion

This thesis set out to investigate two key research questions, first how LLMs can be used to automate Agile project insight and second how automation affects the workflow efficiency of TTLs/Scrum masters.

First of the results show that LLMs automation within Agile environments can significantly enhance Agile project management. The developed tool successfully automated the retrieval, processing, and analysis of Agile metrics, such as velocity, cycle time, commit vs deliver ratios and blocked time, and delivered tailored insights via email using TUI's AI and Mail API services. By shifting this work from a manual to an automated process, the tool enabled notable time savings and introduced more data-driven, objective assessments of team performance.

Secondly, feedback from TTLs confirmed that the tool not only provided efficiency but also served as a valuable archive of team performance over time, promoting reflection and continuous improvement while also showcasing metrics which are close to impossible to gather within Agile project management tools alone. By studying the results from the surveys we are able to see the large number of positive results and also a low SD, indicating strong agreement among the respondents and strengthening the reliability of the findings. While the AI-generated insights were not intended to replace human judgment, they served as a strong foundation for discussions during reviews.

Beyond that the tool also contributed to standardization and knowledge sharing, ensuring that all teams received insights in the same format. This not only improved comparability but also provided an archival resource for long term reflection and improvement. At the same time, the study has highlighted important challenges and limitations. Full contextual understanding remains a limitation of AI-generated content, while the tool can detect anomalies or trends, it cannot always explain the underlying human or organizational causes. There are also concerns related to data privacy, flexibility to other workflows, and the current lack of interactivity in the system, which relies on fixed, scheduled reports.

Despite these limitations, the project illustrates the transformative potential of AI in Agile environments. By moving from static dashboards and manual reporting to dynamic, AI-assisted insights, organizations can empower their teams with better, faster, and more scalable decision support.

In conclusion, this demonstrates that AI-driven automated insights can enhance Agile projects management by reducing repetitive tasks, providing objective metrics and support decision making processes. The thesis covers the tool made specifically

to address this possibility, but also tools where similar advantages have been recorded. As AI capabilities continue to evolve and Agile teams generate more data, the opportunity to build smarter, more adaptive workflow is greater than ever. Future developments should focus on interactivity, personalization, and broader integration to maximize the value of such tools across diverse agile contexts.

7 References

1. Beck K, Cunningham W, et al. Manifesto for Agile Software Development. 2001. Available from: <https://agilemanifesto.org>
2. Atlassian. Get started with Jira – comprehensive beginner’s guide. Available from: <https://www.atlassian.com/software/jira/guides/getting-started/introduction> (Accessed: 04 Mar 2025).
3. Naveed H, Khan AU, Qiu S, Saqib M, Anwar S, Usman M, Akhtar N, Barnes N, Mian A. A Comprehensive Overview of Large Language Models. arXiv preprint arXiv:2307.06435. 2024. Available from: <https://arxiv.org/abs/2307.06435> (Accessed: 01 Aug 2025).
4. Nilsson M, et al. Artificial Intelligence and Project Management: A Global Chapter-Led Survey. PMI. 2024. Available from: <https://www.pmi.org/-/media/pmi/documents/public/pdf/artificial-intelligence/community-led-ai-and-project-management-report.pdf> (Accessed: 11 Apr 2025).
5. Pham KP, Neumann M. How to measure performance in agile software development? A mixed-method study. In: 2024 50th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). 2024. Available from: <https://ieeexplore.ieee.org/document/10803386>
6. Almeida F, Carneiro P. Perceived Importance of Metrics for Agile Scrum Environments. ResearchGate. 2023. Available from: https://www.researchgate.net/publication/371523874_Perceived_Importance_of_Metrics_for_Agile_Scrum_Environments (Accessed: 07 Apr 2025).
7. Asad S, Feng Z. AI in Agile Workflows: Measuring Developer Productivity with Data-Driven Insights. ResearchGate. 2023. Available from: https://www.researchgate.net/publication/388833846_AI_in_Agile_Workflows_Measuring_Developer_Productivity_with_Data-Driven_Insights (Accessed: 20 Apr 2025).
8. Chronis K, Gren L. Agility Measurements Mismatch: A Validation Study on Three Agile Team Assessments in Software Engineering. 2019. Available from: <https://arxiv.org/abs/1904.02460> (Accessed: 22 Jul 2025)
9. Ram P, Rodriguez P, Oivo M. Software Process Measurement and Related Challenges in Agile Software Development: A Multiple Case Study. 2018. Available from: <https://arxiv.org/abs/1809.00860> (Accessed: 22 Jul 2025)

10. Fawzy, A., Tahir, A., Galster, M. et al. Exploring data management challenges and solutions in agile software development: a literature review and practitioner survey. 2025. Available from <https://doi.org/10.1007/s10664-025-10630-4> (Accessed: 22 Jul 2025)
11. Project Management Institute (PMI). Shaping the future of Project Management with AI. PMI. 2023. Available from: <https://www.pmi.org/learning/thought-leadership/ai-impact/shaping-the-future-of-project-management-with-ai> (Accessed: 20 Apr 2025).
12. Ferrao R, Miranda F, Soler D. LLM Contribution Summarization in Software Projects. 2025. Available from: <https://arxiv.org/abs/2505.17710>. (Accessed 22 Jul 2025)
13. Birru H. Exploring the use of LLMs in agile technical documentation. Mälardalens Universitet. 2024. Available from: <https://mdu.diva-portal.org/smash/get/diva2:1901287/FULLTEXT01.pdf> (Accessed: 20 Apr 2025).
14. Budacu E, Pocatilu P. Real Time Agile Metrics for Measuring Team Performance. ResearchGate. 2018. Available from: https://www.researchgate.net/publication/331157722_Real_Time_Agile_Metrics_for_Measuring_Team_Performance (Accessed: 22 Jul 2025)
15. Lillemäe LL. Key performance indicators for measuring efficiency and effectiveness in an agile software development team. 2025. Available from: <https://urn.kb.se/resolve?urn=urn:nbn:se:kau:diva-103339> (Accessed: 03 Apr 2025).
16. Umer M, Aljahdali S, Hussain T, Jeon M. Integrating AI for Agile Project Management: Innovations, Challenges, and Benefits. Available from: https://www.researchgate.net/publication/382241441_Integrating_AI_for_Agile_Project_Management_Innovations_Challenges_and_Benefits(Accessed: 20 Jul 2025)
17. Kupiainen E, Mäntylä M, Itkonen J. Using metrics in Agile and Lean Software Development – A systematic literature review of industrial studies. 2015. Available from: <https://doi.org/10.1016/j.infsof.2015.02.005> (Accessed 22 Jul 2025)

Apendix

A Survey Questionnaire

35

A Survey Questionnaire

The following survey was distributed to Technical team leads, Scrum masters, and other Agile roles at TUI to evaluate the AI-driven Agile Insights tool. The survey combines quantitative rating scales with qualitative feedback to capture both measurable outcomes and user perspectives.

Section 1: Background Information

1. What is your current role?
 - Technical team lead
 - Scrum master
 - Developer
 - Product Owner
 - Other: _____
2. How many Agile teams do you currently oversee or work in?
 - 1
 - 2–3
 - 4+

Section 2: Perceived Usefulness

3. On a scale of 1–5, how useful did you find the automated insights?
 - 1 = Not useful at all
 - 5 = Extremely useful
4. Which parts of the insights did you find most valuable?
 - Metrics overview (velocity, cycle time, blocked time, etc.)
 - Historical comparisons (trends over time)
 - AI-generated recommendations
 - Issue/blocker analysis

- Other: _____

5. Please explain briefly what you found most valuable about the tool.

Section 3: Efficiency and Time Savings

6. Did the tool save you time compared to your previous manual process?

- Yes
- No

7. If yes, approximately how much time do you save per sprint?

- Less than 30 minutes
- 30 minutes – 1 hour
- 1–2 hours
- More than 2 hours

8. On a scale of 1–5, how much do you agree with the statement:
“The tool improved my workflow efficiency.”

- 1 = Strongly disagree
- 5 = Strongly agree

Section 4: Quality of AI-Generated Insights

9. On a scale of 1–5, how clear and understandable were the AI-generated insights?

10. On a scale of 1–5, how actionable were the AI’s recommendations?

11. Did you notice any limitations or inaccuracies in the AI-generated insights?

- Yes
- No

If yes, please describe: _____

Section 5: Trust and Adoption

12. To what extent do you trust the automated insights when making decisions?
- 1 = Not at all
 - 5 = Fully trust
13. Do you see this tool as a replacement for manual reporting, or more as a support tool?
- Replacement
 - Support tool
 - Not useful in either way
14. Would you recommend this tool to other Agile teams?
- Yes
 - No
 - Maybe

Section 6: Open Feedback

15. What improvements would make the tool more valuable to you?
-
16. Any additional comments or concerns (e.g., about privacy, accuracy, or workflow integration)?
-

TRITA-CBH-GRU-2025:316
Stockholm, Sweden 2025

www.kth.se