Metrics in Large-Scale Agile Software Development: A Multivocal Literature Review

Renato Menezes[1], **Marcelo Marinho**[1], **Suzana Sampaio**[1]

[1]Computer Department – Universidade Federal Rural de Pernambuco (UFRPE)
Recife – PE – Brazil

{renato.vmenezes, marcelo.marinho,suzana.sampaio}@ufrpe.br

Abstract. *Software development is widespread across various sectors. As large-scale projects increasingly adopt agile development practices, there arises a need for metrics to enhance team coordination, promote continuous improvement and monitor progress. This discussion focuses on the current state of metrics for large-scale agile software development, outlining the reasons for their adoption and showcasing the achieved results. The analysis involves a comprehensive literature review, exploring grey literature. A catalog of metrics applicable to scalable agile projects is presented, featuring examples such as 'Velocity', 'Business value per effort', and 'Defect rate'.*

1. Introduction

Coordinating multiple teams across different projects using agile methodologies represents a complex and crucial challenge in the software development environment [Marinho et al. 2021, Camara et al. 2021]. In this dynamic context, maintaining an adequate level of communication becomes more challenging [Philipp et al. 2023]. Large organizations struggle to use metrics effectively due to many influencing factors, such as lack of skill demanded, infrastructure and a lack of guidance, making efficient measurement complex [Korpivaara et al. 2021]. Therefore, for monitoring processes and projects, more than relying solely on qualitative feedback by regularly communicating with each agile team is required for program management decision-making [Philipp et al. 2023, Camara et al. 2021]. In such circumstances, metrics can enhance or replace qualitative feedback [Ertaban et al. 2018, Philipp et al. 2023].

Despite the fundamental importance of metrics in this context and repeated calls from researchers for documented knowledge, there is still a shortage in the scientific literature of dedicated studies to fill this gap [Dingsøyr and Moe 2014, Edison et al. 2021, Korpivaara et al. 2021, Philipp et al. 2023]. Therefore, it is essential to turn to grey literature to add more evidence to the academic literature on metrics used scaling agile, as discussed and presented by industry professionals and experienced academics.

This paper aims to contribute to the body of knowledge in large-scale agile methods, proposing a comprehensive approach that includes the compilation of metrics, an in-depth exploration of the justifications for their application, and an analysis of the results obtained. To do so, we systematically categorize metrics to provide continuous support to organizations in agile scaling and those exploring new work paradigms. We highlight how the judicious application of metrics can drive agility at scale, promoting substantial improvements in organizational practices and development team performance.

In addition to this introduction, Section 2 presents a background and related work. Sections 3 and 4 present the research methodology and results, respectively. Section 5 discusses the findings of this study. Section 6 exposes some threats to validity, and finally, Section 7 provides some conclusions and contributions of the work.

2. Background and Related Works

2.1. Software Metrics

In traditional software development, metrics assisted in project guidance and decision-making [Chloros et al. 2022]. With the popularization of agile values, different software development processes emerged, where metrics needed to adapt to changes in all development phases [Hossain et al. 2021, Leal et al. 2022]. Agile teams use metrics to improve agile processes, comply with agility protocols, enhance software quality during development, improve estimation and planning, and increase productivity [Chloros et al. 2022, Leal et al. 2022].

Metrics are measurements, variables to which a quantity is assigned due to measuring one or several entities [Fenton and Bieman 2014, Leal et al. 2022]. Due to their advantages, many agile metrics have been proposed or adapted to help measure various aspects of software development processes and products [Leal et al. 2022].

2.2. Large-Scale Agile Development

Most agile software development organisations are based on self-managed teams responsible for developing a subset of features [Camara et al. 2020, Marinho et al. 2021, Marinho et al. 2019]. Agile at scale is a way to manage processes beyond team-level software development using specific agile tools [Laanti 2014], also encompassing the implementation of agile methods and principles throughout the organisation [Dingsøyr and Moe 2014].

Dingsoyr *et al.* [Dingsøyr et al. 2014] categorize the scale of agile software development projects by team count. Small-scale projects consist of a single team, coordinating through standard agile practices. Large-scale projects involve 2 to 9 teams, requiring scaling methodologies for coordination. Projects with 10 or more teams are considered very large-scale, necessitating a scaling framework. This study follows Dingsoyr's definition of large-scale agile.

The organization's size, geographic distribution, team size, culture, system complexity, governance and business objectives are some aspects considered when scaling agile methods [Razzak et al. 2017, Shameem et al. 2017]. The main known agile scaling frameworks are: Scrum-at-Scale, Large Scale Scrum (LeSS), Scaled Agile Framework (SAFe), Disciplined Agile (DA), Spotify and Nexus [Shameem et al. 2017, Edison et al. 2021].Compared to SAFe, other agile frameworks (e.g., Scrum-at-Scale, LeSS, Nexus, Spotify) offer fewer native metrics, artifacts, roles and events beyond Scrum [Razzak et al. 2017].

2.3. Related Work

Among the related works, Edison et al. (2021) investigated the various known methods of large-scale agile development by comparing them through an SLR. The review did not present which metrics are used in some of the major existing frameworks [Edison et al. 2021]. Kišš and Rossi (2018) sought to understand the "Agile to Lean" transformation processes, covering benefits, challenges, and metrics in transformations at larger scales. Lead Time emerged as the most used metric to measure transformation progress among the metrics. However, metrics were presented only in that context [Kišš and Rossi 2018].

Britto et al. (2014) reported the evidence obtained from SLRs regarding effort estimation in software development in the global and agile context. Size metrics such as Function Points, Lines of Code, Use Case Points, and Story Points were presented. However, the work is focused only on effort estimation aspects [Britto et al. 2014].

3. Research Method

We conducted a Multivocal Literature Review (MLR) to gather and synthesize the limited information found in academic literature with other findings from grey literature, aiming to extract information about the metrics used within the context of large-scale agile development. This MLR followed the guidelines provided by [Garousi et al. 2019].

To specify the objective of this article, two research questions were formulated: **(RQ1)** What metrics are used in large-scale agile software development? And **(RQ2)** What are the reasons and results of using metrics in large-scale agile development?

We used the following databases to find relevant literature: ACM, IEEExplore, Springer-Link, Scopus and ScienceDirect to specifically locate academic literature. And Google search (http://www.google.com/) to find grey literature. To find evidence according to the research questions, the string used was as follows: **(Metric) AND (SAFe OR "Scaled agile framework" OR LeSS OR "Large scale scrum" OR "Scrum-at-Scale" OR "Scrum@Scale" OR DA OR "Disciplined agile" OR Spotify OR Scrum OR "Scaling agile" OR Nexus) AND ("Large-scale" OR Large OR Scaled)**.

We defined inclusion and exclusion criteria to ensure the inclusion of all relevant sources and the exclusion of sources outside the scope. We included: (i) literature that explicitly discusses the use of metrics related to large-scale agile projects, solutions, or adaptations of metrics in this scenario; (ii) conference papers, journals and pages about metrics of the related frameworks; (iii) literature published after 2001; (iv) first 10 pages of the Google search. We excluded: (i) inaccessible literature; (ii) results that Google search considers similar to other results; (iii) Google ads; (iv) articles and pages in a language other than English.

For academic literature, the period from January 2001 to December 2022 considers the entire period since the publication of the Agile Manifesto. A total of 3,479 articles were returned in the automatic selection. After removing 393 duplicate articles, 3,086 remained to be classified, of which no articles belonged to ScienceDirect since all duplicates came from this database.

The process of selecting studies began with analyzing titles and abstracts (phase 1), in which 3,027 articles were discarded for not meeting the inclusion criteria. Of these works, 59 were sent to phase 2, in which the introduction and conclusion of the articles were read, and then 34 articles were excluded. After reading the full text of the remaining 25, only 14 papers that met the inclusion criteria proceeded to extraction.

Before selecting results from grey literature, links from official pages available on metrics of recognized frameworks present in the string were separated in advance to ensure their inclusion in the research. Subsequently, the first ten links from search with string were considered, excluding Google ads. From the valid results, up to 2 cycles of forward snowballing were applied in the selection process. The search for grey literature was conducted in January 2023. After reading the titles and metatexts, 13 initial valid results were selected, considering the top ten links plus snowballing and three official framework pages. No links were academic articles. During the full-text reading, one page was added after snowballing. Applying the inclusion criteria to the 14 sources, five pages were eligible for the study. Table 1 summarizes and quantifies the results.

Table 1. Summary of research findings and selection for primary studies

Search mechanism	Initial results	Titles, metatext, abstracts, introductions, conclusions	Full texts
IEEEXplore	974	5	3
ACM	285	1	0
SpringerLink	184	14	7
Scopus	1643	5	4
ScienceDirect	0	0	0
Google	13	13	5

To enable peer review and study replicability, all methodology and research data is available as supplementary material (See in [Menezes et al. 2024b]).

3.1. Analysis

We consider the Software Development Life Cycle (SDLC) stages to address RQ1. The mapped metrics applied in the agile scaling environment will be linked to some SDLC stages. To do so, we will consider both information provided in the studies about the stages of the cycle in which they were applied and the descriptions and characteristics of the metrics themselves.

In addition, we will adopt the Goal Question Metric (GQM) approach to address RQ2, aiming to identify patterns in the transformations caused by the discovered metrics. The GQM approach provides tools for defining and interpreting software measurements guided by organizational goals, specifying a measurable objective refined into a set of quantifiable questions related to the objectives [Basili 1992]. As presented by the author, these questions, in turn, help define a set of metrics and data for collection.

Regarding the goal within the GQM approach, inspired by [Scrum@Scale 2022], it is observed that large agile organization seeks to improve and measure at least one of the following factors: customer responsiveness, value delivery, productivity, sustainability, and quality [Scrum@Scale 2022]. Therefore, the target metric model have the following objectives: to improve customer responsiveness, commercial value delivery, end-to-end productivity, sustainability of practices from both the organizational and customer perspectives and quality of the software development process. Based on these objectives, five questions were proposed: **Q1:** Are we more responsive to customers in the agile way of working? **Q2:** Do we deliver more value to customers in the agile way of working? **Q3:** Are we more productive working agile? **Q4:** Do we have better sustainability practices in the organizational environment? **Q5:** Do we have better product quality?

4. Results

4.1. Overview of academic studies

The included academic studies were classified according to research type aspects based on [Wieringa et al. 2006]. Regarding the contributions of the works, they were classified according to [Petersen et al. 2008], which include theory, model, framework, guideline, lessons learned, addendum, and tool. Table 2 overviews academic studies, research type, contribution aspects, publication year, and methods used.

Table 2. Overview of academic studies

Year	Research Type	Contribution	Study Method	Reference
2010	Experience	Lessons learned	Experience report	[Greening 2010]
2011	Experience	Lessons learned	Case study	[Brown 2011]
2012	Solution	Tool	Action research/literature review	[Staron et al. 2012]
2013	Experience	Tool	Experience report	[Tabib 2013]
2013	Solution	Guideline	Grounded theory/quasi-experiment	[Heidenberg et al. 2013]
2013	Solution	Tool	Action research	[Staron et al. 2013]
2014	Philosophical	Addendum	Grounded theory	[Laanti 2014]
2015	Experience	Lessons learned	Experience report	[Greening 2015]
2015	Experience	Lessons learned	Case study	[Tripathi et al. 2015]
2016	Validation	Framework	Case study	[Grimaldi et al. 2016]
2017	Experience	Framework	Observations	[Razzak et al. 2017]
2018	Experience	Lessons learned	Multiple case study/exploratory	[Stettina and Schoemaker 2018]
2020	Solution	Tool	Case study	[Thawaba et al. 2020]
2022	Validation	Framework	Experience report/experiment	[Tessarolo et al. 2022]

Table 3 presents information about the number of organizations (No. Org) referenced in the included literature. The symbol "+" in the column for organization size next to the term "large scale" represents a quantity of teams greater than or possibly greater than 9, and the term "large scale" represents a quantity between 2 and 9 teams [Dingsøyr et al. 2014]. In the column distribution, "Locally" is used when people are working remotely from the same city, "Nationally" working from cities within a single country, "Continentally" for working from different countries within the same continent and "Globally" for working from countries on different continents.

Table 3. Organizational context of the studies

No. Org	Size	Framework/ Method	Distribution	Industry	Reference
1	+ Large scale	Scrum+Kanban	Globally	IT	[Greening 2010]
1	+ Large scale	Scrum+XP	Continentally	Banking	[Brown 2011]
1	Large scale	Scrum	Globally	IT	[Tabib 2013]
-	Undefined	Scrum+Kanban	Undefined	IT	[Greening 2015]
1	Large scale	Scrum+Lean	Continentally	Healthcare	[Tessarolo et al. 2022]
1	+ Large scale	Agile+Lean	Nationally	Telecom	[Staron et al. 2012]
1	+ Large scale	Agile+Lean	Nationally	Telecom	[Staron et al. 2013]
1	Undefined	Agile	Nationally	IT	[Heidenberg et al. 2013]
-	Undefined	SAFe	Undefined	Undefined	[Laanti 2014]
1	Undefined	SAFe	Undefined	Telecom	[Grimaldi et al. 2016]
1	Large scale	SAFe	Globally	IT	[Razzak et al. 2017]
1	Undefined	SAFe	Undefined	Healthcare	[Thawaba et al. 2020]
5	Undefined	Scrum-of-scrums /SAFe/Spotify	Undefined	Telecom /Government	[Stettina and Schoemaker 2018]
2	Large scale	Kanban	Globally	Telecom	[Tripathi et al. 2015]
1	Large scale	LeSS	Continentally	Financial	[Korson 2015][1]

[1] *Grey literature sources.*

4.2. RQ1 - What metrics are used in large-scale agile software development?

Table 4 presents all the metrics found in this review. They are grouped by the primary studies in which they were mentioned or by the reference of the internet pages where they were mentioned and classified according to the agile framework/method in which they were applied or related. Metrics with identical names that appear repeatedly were identified in both types of literature and are equivalent.

Table 4. Metrics for large-scale agile software development (MLR)

Metrics	Framework/Method	Reference
Time to delivery first increment; Time to project closure; Velocity; Blocked tasks; Defect rate with severity; Process maturity level; Agile practices adoption rate; Function points per man-year; Burndown chart; Burnup chart; Agile Team Pulse (Adoption of iterative development; Two-level planning; Shared vision; Continuous integration and Team-based approach)	Scrum/XP	[Brown 2011]
Velocity; Velocity Deviation; Forecast Horizon; True Sprint Length; Dependency Count; Lead Time	Scrum/Kanban	[Greening 2015]
Net Present Value per effort	Scrum/Kanban	[Greening 2010]
Estimation accuracy delta; Effectiveness; Efficiency; Planned virtual hour cost; Real virtual hour cost; Waste; Impediments; Delta cost	SAFe	[Grimaldi et al. 2016]
Customer service response time; Functionality per work effort; Business value per effort; Commit Pulse; Flow distribution; Number of external problem reports; Open days and external problem reports; Lead Time; Cycle time per resource	Agile	[Heidenberg et al. 2013]
Net Promoter Score (NPS)	SAFe	[Laanti 2014]
SAFe practices adoption rate (Product Ownership Health, PI/ Release Health, Sprint Health, Team Health and Technical Health)	SAFe	[Razzak et al. 2017]
Measurement systems completeness indicator	Agile/Lean	[Staron et al. 2013]
Release readiness indicator	Agile/Lean	[Staron et al. 2012]
Planned velocity; Velocity	Scrum/Scrum of Scrums	[Stettina and Schoemaker 2018]
Team happiness	Scrum/Spotify	[Stettina and Schoemaker 2018]
Cycle time per resource; Velocity	Scrum/SAFe	[Stettina and Schoemaker 2018]
Code lines per user story; Number of files per user story; Code lines for refactoring; Number of developers per resource; Unit test coverage per user story; Unit test success per user story; Number of defects per user story	Scrum	[Tabib 2013]
Perceived effectiveness (Teamwork assessment; Requirement elicitation; Planning; Methodology quality; Culture; Knowledge sharing; General process perception; Team morale; Participant engagement and satisfaction)	Scrum/Lean	[Tessarolo et al. 2022]
Cost per function; Time per function; Remaining time; Time spent (all tasks); Remaining cost; Cost spent (all tasks); Remaining functions; Completed functions (all tasks); Remaining patterns; Achieved patterns (all tasks); Time Spent (main tasks using subtask completion); Available time; Available cost	SAFe	[Thawaba et al. 2020]
Work In Progress; Lead Time	Kanban	[Tripathi et al. 2015]
Employee engagement; Flow distribution; Velocity; Flow time; Flow Load; Flow efficiency; Flow predictability; Deployment frequency; Wait time for changes; Time to restore service; Change failure rate; Business agility (Lean-Agile Leadership – Team and Technical Agility – Agile Product Delivery – Enterprise Solution Delivery – Lean Portfolio Management – Organizational Agility – Continuous Learning Culture); Core competencies (Team and Technical – Execution – Value Delivery – Organizational and Value Stream – Lean Portfolio and Strategy – Customer and Solutions – Quality Practices)	SAFe	[SAFe 2022][1]
Test pass rate; In-flow defect rate to out-flow defect rate; Release readiness	SAFe	[Copado 2022][1]
Value of business by effort; Team happiness; Function points per man-year; Defect rate; Service downtime	Scrum-at-scale	[Scrum@Scale 2022][1]
Predictability index; Technical debt; Acceleration	Agile/Lean	[Liyanage 2014][1]
Burnup chart; Work In Progress	LeSS	[Korson 2015][1]

[1] *Grey literature sources.*

Table 5 presents an overview of the metrics grouped by software development stage on the life cycle.

Table 5. Metrics linked to the Software Development Life Cycle

Stage	Metric	Reference
Product feasibility		
	Planned virtual hour cost; Estimation accuracy delta; Effectiveness; Real virtual hour cost; Waste; Efficiency; Impediments; Delta cost	[Grimaldi et al. 2016]
	Cost per function; Time per function	[Thawaba et al. 2020]
	Planned velocity	[Stettina and Schoemaker 2018]
	Number of developers per resource	[Tabib 2013]
	Flow predictability	[SAFe 2022]
Development		
	Velocity; Blocked tasks; Burndown chart; Burnup chart	[Brown 2011]
	Code lines per user story; Number of files per user story; Code lines for refactoring; Unit test coverage per user story; Unit test success per user story; Number of defects per user story	[Tabib 2013]
	Velocity; Dependency Count	[Greening 2015]
	Functionality per Work Effort; Commit Pulse	[Heidenberg et al. 2013]
	Velocity	[Stettina and Schoemaker 2018]
	Work In Progress	[Tripathi et al. 2015]
	Velocity; Flow Load	[SAFe 2022]
	Burnup chart; Work In Progress	[Korson 2015]
Testing		
	Defect rate with severity	[Brown 2011]
	Number of external problem reports; Open days and external problem reports	[Heidenberg et al. 2013]
	Number of defects per user story	[Tabib 2013]
	Release readiness indicator	[Staron et al. 2012]
	Remaining time; Time spent (all tasks); Remaining cost; Cost spent (all tasks); Remaining functions; Completed functions (all tasks); Remaining patterns; Achieved patterns (all tasks); Time spent (main tasks using subtask completion); Available time; Available cost	[Thawaba et al. 2020]
	Defect rate	[Scrum@Scale 2022]
	Test pass rate; In-flow defect rate to out-flow defect rate; Release readiness	[Copado 2022]
Deployment		
	Customer service response time	[Heidenberg et al. 2013]
	Net Promoter Score (NPS)	[Laanti 2014]
	Deployment frequency; Wait Time for Changes; Time to restore service	[SAFe 2022]
	Service downtime	[Scrum@Scale 2022]
Maintenance		
	Net Present Value per effort	[Greening 2010]
	Time to delivery first increment; Time to project closure; Process maturity level; Agile practices adoption rate; Agile team pulse; Function points per man-year	[Brown 2011]
	True Sprint Length; Lead Time; Velocity Deviation; Forecast Horizon	[Greening 2015]
	Perceived effectiveness	[Tessarolo et al. 2022]
	Measurement systems completeness indicator	[Staron et al. 2013]
	Lead Time; Cycle time per resource; Business value per effort; Flow distribution	[Heidenberg et al. 2013]
	SAFe practices adoption rate	[Razzak et al. 2017]
	Cycle time per resource; Team happiness	[Stettina and Schoemaker 2018]
	Employee engagement; Flow distribution; Flow time; Flow efficiency; Change failure rate; Business agility; Core competencies	[SAFe 2022]
	Business value per effort; Team happiness; Function points per man-year	[Scrum@Scale 2022]
	Predictability index; Technical debt; Acceleration	[Liyanage 2014]

4.3. RQ2 - What are the reasons and outcomes of using metrics in large-scale agile development?

The final step in GQM modeling was determining the 80 metrics in the multivocal literature (Table 4) to be used according to the raised questions. Some metrics related to each questions presented Q1, Q2, Q3, Q4 and Q5 are presented in Table 6, 7, 8, 9 and 10 respectively. Each table lists the metric name, descriptions (when necessary) and the formulas or data that quantify them (See all metrics in [Menezes et al. 2024a]).

Table 6. Are we more responsive to customers in the agile way of working?

Name	Description	Calculation Formula
Time to delivery first increment	Time spent from project initiation to delivery of the first increment	*Delivery date of the first increment - Project initiation date*
Time to project closure	Time spent from project initiation to project closure	*Project closure date - Project initiation date*
True Sprint Length		*Actual delivery date of the increment when greater than the previously defined sprint end date - Sprint initiation date*
Forecast Horizon	Sum of estimation points, h, from the top of the product backlog to a given point	If we know the estimated team velocity $\mu(V)$ and the standard deviation $\sigma(V)$, we can express the Forecast Horizon in sprints as $h/\mu(V) \pm \sigma(V) * h/\mu(V)$ for 68% confidence or $h/\mu(V) \pm 2\sigma(V) * h/\mu(V)$ for 95% confidence. If we know the sprint length l, we can express the Forecast Horizon in time units as $hl/\mu(V) \pm l\sigma(V) * h/\mu(V)$
Lead Time		*Completion date of a process - Start date of a process*
Customer Service Response Time	Measures the return of customer service requests	*Resolved customer service request date - Created customer service request date*
Cycle Time per Resource	Measures the cycle time for resources selected for development.	*Date of completion of resource readiness - Date when the resource was added to the backlog*
Flow Distribution	Measures the quantity of each type of work in the system over time	A simple measurement is counting the number of each type of work item at any given time. A more accurate measure may consider the size of each work item
Release Readiness Indicator	Predicts in which week the product release would be possible given the number of known defects up to that point, how many defects were removed on average in the last 4 weeks and how many defects were expected to be discovered	*#Defects / Defect_removal_rate - (Test_execution_rate - Test_pass_rate).* Where *#defects* is the number of open defects for the product, *defect_removal_rate* is the average amount of defects removed in the last 4 weeks, *test_execution_rate* is the average number of test cases executed in the last 4 weak points and *test_pass_rate* is the average number of test cases passed in the last 4 weak points
Time per Function		*Task duration time / Task function quantity*
Remaining Time	Remaining time to complete the task	*Task duration time / (Task test date - Task start date)*
Time Spent (All Tasks)	Time spent during task implementation	*((Task test date - Task start date) / Task duration time)* $*$ *100*
Available Time	Available time for sub-tasks or main tasks	*Time spent in sub-tasks - (Task test date - Task start date)*
Flow Time	Total time elapsed for all stages of a workflow	Measured by the average period of time it takes to complete a certain type of work item
Wait Time for Changes		Amount of waiting time to make a change

Table 7. Do we deliver more value to customers in the agile way of working?

Name	Description	Calculation Formula
Net Present Value per Effort	Profitability value by effort	*Net Present Value / Effort*
Functionality per Work Effort	How much functionality can be delivered relative to a certain work effort	*Test points / Person hours*
Business Value per Effort	Here it is indicated as more frequent major releases in relation to the work effort	*Number of major releases in a year / Hours per person*
Net Promoter Score	User/customer feedback	Calculated by asking customers if they would recommend the product to colleagues on a scale from 0 to 10. Responses are categorized as detractors (0-6), passives (7-8) and promoters (9-10). Finally, the total percentage of detractors is subtracted from the percentage of promoters to determine the NPS
Flow Efficiency	How much of the total flow time is spent on value-added work activities versus waiting between steps	*Total active time / Flow time*
Flow Predictability	Measures how well teams, Agile Release Trains (ARTs) and Solution Trains can plan and meet their Program Increment (PI) objectives	Ratio between planned business value achieved and actual business value delivered in a PI

Table 8. Are we more productive working agile?

Name	Description	Calculation Formula
Velocity		Quantity of story points completed for work items of a type over a period of time
Blocked Tasks		Number of tasks blocked during a specific time period
Function Points per Man-Year		Amount of work done by an individual throughout the year
Burndown Chart	Measures sprint progress and provides indicators of the team's work process	Marks the sprint days on the horizontal axis and the points planned to compose the sprint on the vertical axis, starting from the maximum points of the sprint (team velocity) to zero
Burnup Chart	Measures progress based on remaining hours or points from the top down. Measures release progress and provide indicators of the team's work process	Marks the sprint days on the horizontal axis and the points planned to compose the sprint on the vertical axis, starting from the maximum points of the sprint (team velocity) to zero
Velocity Deviation	Measures velocity stability	$\sigma(V)/\mu(V)$. Where $\mu(V)$ is the expected velocity (average velocity based on a certain number of previous sprints) and $\sigma(V)$ is the velocity standard deviation
Dependency Count	Dependency between teams where bugs and delays affect immediate dependents	Number of immediate dependents
Efficiency	Indicates how well we use the team compared to maximum capacity	*AH / C*. Where *AH* represents the actual hours spent producing tangible results and *C* is the ideal number of hours a team can deliver, depending on team size, number of teams, non-working days and days spent on ceremonies. $C = S{*}(DS-(NWD+(DS{*}KO/20))){*}TS{*}DH)$, where *DS* is the sprint duration, *NWD* represents days off, *KO* stands for knowledge transfer/planning/estimation days, *TS* is the team size and *DH* is daily development capacity in hours
Impediments	Any 'time loss' due to a defect or obstacle that hampers productivity	Number of hours that do not produce tangible results
Commit Pulse	Measures how continuous integration is within sprints	Number of days between commits
Planned Velocity		Amount of work (story points) that a team expects to complete during a sprint
Cost per function		*Task cost / Number of task functions*
Remaining Cost	Measures the remaining cost for task completion	*Task cost - Expense cost*

In this MLR, among the academic studies, there are reports from large organizations that scaled Scrum to scale beyond software development teams, promoting agile thinking throughout the organization [Greening 2010], as well as to development teams working together [Brown 2011, Greening 2015], or situated in different geographical locations [Tabib 2013, Tessarolo et al. 2022], using metrics to address challenges and promote improvements. Purely agile management was referenced in the research in studies that addressed measurements more focused on aspects of organizational transformations [Heidenberg et al. 2013], improvement of workflow control [Staron et al. 2013], and organization performance goals [Staron et al. 2012]. The SAFe framework appears in research with proposals to enrich its native metrics to meet demands for speed in deliveries by measuring resources and costs in the process [Grimaldi et al. 2016] and improving confidence in developing critical security systems [Thawaba et al. 2020]. An efficient mechanism for measuring the adoption rate of SAFe practices in Small and Medium Enterprises (SMEs) wishing to adapt to scalable frameworks was also found [Razzak et al. 2017]. In [Stettina and Schoemaker 2018], performance, quality, progress and status metrics were mentioned in methodologies such as Scrum-of-scrums, SAFe and Spotify. The importance of Kanban in controlling and scaling workflows among teams using metrics such as work limits and wait times was highlighted [Greening 2010, Greening 2015, Tripathi et al. 2015].

In the grey literature, most findings are related to SAFe, which has a robust system

Table 9. Do we have better sustainability practices in the organizational environment?

Name	Description	Calculation Formula
Agile Team Pulse	Involves team's regular casual assessments to understand their views on adopting iterative development practices, two-level planning, shared vision, continuous integration and a team-based approach	Collected through informal surveys
SAFe Practices Adoption Rate	Self-assessment survey sent to participants. Each question includes a numeric rating (Likert scale) and an optional comment section	The Likert scale has six response options (from 'never' to 'always') to gauge the frequency of practice usage in areas like product ownership health, PI/release health, sprint health, team health and technical health
Measurement Systems Completeness Indicator	Assessment of measurement system completeness for workflow monitoring, as temporal and process dependencies are used between process activities	*(#Activities with measures or indicators / #Activities in total)* $*$ *100%*. The measurement system that provides measures or indicators for all activities of the monitored process is 100
Employee Engagement	Measures how motivated individuals feel and how actively engaged they are in supporting the organization's goals and values	Various methods exist to gauge employee engagement, and each organization should choose what suits them best. Some use an annual survey, while others rely on an Employee Net Promoter Score (eNPS), asking employees how likely they are to recommend their employer on a 10-point scale
Business Agility	High-level assessment summarizing how agile the business is at any given time	Two assessment methods are available: (i) Participants fill out assessments independently, followed by group discussion and analysis; (ii) All participants collectively discuss and agree on scores (1 to 5) for each statement. The assessment report includes visualizations tracking progress in SAFe's seven core competencies

of suggested metrics [SAFe 2022]. In addition to SAFe's native metrics, we found a proposal to add quality measurement to its flow monitoring in a DevOps environment [Copado 2022], a perspective currently not covered in [SAFe 2022]. Scrum-at-scale does not officially commit to a predefined metrics system but clearly emphasizes the need to monitor productivity, quality, value delivery and sustainability, directly influencing decision-making and promoting transparency [Scrum@Scale 2022].

The synergy between Agile and Lean is vital in improving agility in large-scale agile systems [Liyanage 2014]. In [Liyanage 2014], metrics to measure business agility based on predictability of development risks/costs, maintaining product reliability and adaptability to conditions impacting value delivery were presented and directly associated with Lean and Agile management principles. LeSS emerged in the research through a case study summary in which using the burnup chart as an indicator of scope increase and establishing a WIP limit for making corrections were essential for project progress and completion [Korson 2015]. No evidence was found regarding the metrics used to implement the Disciplined Agile scalable agile framework.

5. Discussion

Considering the small number of academic papers (14) it can be inferred that while the interest in agile software development is growing, the community likely still lacks comprehensive knowledge and understanding of this context, given the limited number of studies. Regarding the variety of research types, experience reports predominate [Brown 2011, Greening 2010, Greening 2015, Razzak et al. 2017, Stettina and Schoemaker 2018, Tabib 2013, Tripathi et al. 2015], followed by solution proposals [Heidenberg et al. 2013, Staron et al. 2012, Staron et al. 2013, Thawaba et al. 2020], with 7 and 4 occurrences, respectively. Only 2 validation studies

Table 10. Do we have better product quality?

Name	Description	Calculation Formula
Defect Rate with Severity	Measures the software defect rate	*Number of defects (severity 1 and 2) in production / 100*
Number of External Problem Reports	Measures the total number of external problem reports during a specific time period	Number of external problem reports originating from a particular version
Open Days and External Problem Reports	Measures the average number of days external problem reports have remained unresolved from creation to resolution	*Date of resolved problem reports - Date of problem reports created*
Code Lines per User Story		*Code Lines / User Story*
Number of Files per User Story		*Number of Files / User Story*
Code Lines for Refactoring		Quantity of code lines for refactoring
Number of Developers per Resource		*Number of Developers / Feature*
Unit Test Coverage per User Story		*Unit Test Coverage / User Story*
Unit Test Success per User Story		*Unit Test Pass Rate / User Story*
Number of Defects per User Story		*Number of Defects / User Story*
Change Failure Rate	Measures flow efficiency	Percentage of changes requiring remediation after going into production
Time to Restore Service	Flow time metric	Amount of wait time for service restoration
Test Pass Rate	Measures trends in approval rate for automated test suites	Percentage of approval rate for automated test suite
In-Flow Defect Rate to Out-Flow Defect Rate	Determines if the team can fix more bugs than just those discovered during testing	*In-Flow Defects / Out-Flow Defects*
Defect Rate		*Number of Defects / 100*
Service Downtime		Amount of time a particular service has been down

[Grimaldi et al. 2016, Tessarolo et al. 2022] and 1 philosophical study [Laanti 2014] were recorded. This reinforces the need for more relevant research addressing new metric proposals not yet implemented in practice and extensive evaluations of metric usage in daily operations using large-scale agility.

When separating metrics by lifecycle stages, it is concluded that 29 metrics were or could be used during the maintenance stage, defined as the last stage of the life cycle. 23 metrics were or could be used within the testing stage. The quantities of 15, 12, and 6 metrics were or could be applied for the development, product feasibility, and deployment stages. No metrics were found to be applied or specifically defined for the requirements and design stages.

For each question in the resulting metric model, there is a set of associated metrics out of the total 80 identified in this MLR. For Q1, related to monitoring customer responsiveness, 18 metrics were identified. Value delivered to customers metrics (Q2) has the smallest set, with only 6 metrics. This number may indicate the need to investigate the feasibility of additional value delivery measurement alternatives. The larger set (Q3), with 21 metrics, is related to team performance. There are 18 metrics evaluating sustainability in practices (Q4), and 17 measuring product quality (Q5).

Most works address metrics to support large organizations that strategically scaled Scrum to improve results or mitigate challenges in adverse situations [Brown 2011, Greening 2010, Greening 2015, Tabib 2013, Tessarolo et al. 2022]. Pure agile management was referenced in the research, along with metrics that supported flow control [Staron et al. 2013], organizational transformations [Heidenberg et al. 2013] and the pursuit of business agility [Staron et al. 2012, Liyanage 2014]. We found that SAFe has native metrics cited in articles as proposals to enrich its metrics to monitor resources and costs [Grimaldi et al. 2016], meet evaluating reliability [Thawaba et al. 2020] and quality demands [Copado 2022] and apply organizational model transformations [Razzak et al. 2017]. The relevance of Kanban in controlling and scaling workflows among teams using metrics such as limits and wait times was also noted [Greening 2010, Greening 2015, Tripathi et al. 2015]. The Scrum-at-scale framework suggests measurements to facilitate decision-making and promote transparency [Scrum@Scale 2022].

6. Threats to Validity

The main limitation of this study is that, apart from selecting academic articles, all other phases of the MLR were conducted solely by the first author and validated by other authors. To mitigate this challenge, we followed guidelines from Garousi et al. (2019) for incorporating grey literature and conducting a multivocal review in software engineering, which were adopted and validated.

Another limitation arises from our search string containing generic terms like "Metric", "DA" and "Scrum", resulting in a diverse and irrelevant initial dataset, making the initial literature selection phases challenging.

7. Conclusions

This study aimed to consolidate key findings on large-scale agility metrics, enhancing visibility and accessibility to this knowledge for the scientific community and agile teams. We identified 19 sources through a multivocal literature review, providing insights into metrics, motivations, and outcomes in scalable, agile processes. We considered the stages of the software development life cycle to map the identified metrics. Also, a curated set of 80 metrics was categorized using the GQM model. These metrics are linked to organizational goals, including delivering customer value, improving team productivity, ensuring product quality, and fostering sustainable practices. Notably, most metrics focus on monitoring overall process functionality and boosting productivity, with only a minority addressing value delivery measurement.

Using metrics in large-scale agile development is essential for assessing effectiveness, identifying challenges, promoting transparency, and guiding decision-making. Their analysis significantly impacts the improvement and success of enterprise agile initiatives, providing a clear view of progress, facilitating collaboration among teams, and ensuring consistency in deliveries. By providing valuable context for data-driven decision-making, these metrics represent a fundamental tool for the success and effectiveness of large-scale agile development. Considering the diverse outcomes, a company's selection and customization of metrics often hinge on factors such as organizational size and alignment with its business goals, operational framework, and specific requirements. The insights from this study will equip researchers and practitioners with the knowledge to delve deeper into metrics-related challenges within large-scale agile environments, fostering the development of more targeted solutions.

For future work, we consider conducting case studies in organizations implementing Disciplined Agile as a scalable agile framework to collect data on used metrics. Additionally, we highlight the relevance of surveying to solidify findings regarding metrics implemented in a scalable environment.

Referências

Basili, V. R. (1992). Software modeling and measurement: the goal/question/metric paradigm. Technical report.

Britto, R., Usman, M., and Mendes, E. (2014). Effort estimation in agile global software development context. In *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation: XP 2014 International Workshops, Rome, Italy, May 26-30, 2014, Revised Selected Papers 15*, pages 182–192. Springer.

Brown, A. W. (2011). A case study in agile-at-scale delivery. In *International Conference on Agile Software Development*, pages 266–281. Springer.

Camara, R., Alves, A., Monte, I., and Marinho, M. (2020). Agile global software development: A systematic literature review. In *Proceedings of the XXXIV Brazilian Symposium on Software Engineering*, pages 31–40.

Camara, R., Marinho, M., and Moura, H. (2021). Agile tailoring with scaling agile frameworks in distributed large-scale settings. In *Anais Estendidos do XX Simpósio Brasileiro de Qualidade de Software*, pages 12–17. SBC.

Chloros, D., Gerogiannis, V. C., and Kakarontzas, G. (2022). Use of software and project management metrics in agile software development methodologies: A systematic mapping study. In *Proceedings of the 2022 European Symposium on Software Engineering*, pages 25–32.

Copado, T. (2022). How to optimize flow metrics quality in scaled agile framework (safe). `https://www.copado.com/devops-hub/blog/how-to-optimize-flow-metrics-quality`. (accessed 4 January 2023).

Dingsøyr, T., Fægri, T. E., and Itkonen, J. (2014). What is large in large-scale? a taxonomy of scale for agile software development. In *Product-Focused Software Process Improvement: 15th International Conference, PROFES 2014, Helsinki, Finland, December 10-12, 2014. Proceedings 15*, pages 273–276. Springer.

Dingsøyr, T. and Moe, N. B. (2014). Towards principles of large-scale agile development. In *International Conference on Agile Software Development*, pages 1–8. Springer.

Edison, H., Wang, X., and Conboy, K. (2021). Comparing methods for large-scale agile software development: A systematic literature review. *IEEE Transactions on Software Engineering*.

Ertaban, C., Sarikaya, E., and 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*, pages 1–3.

Fenton, N. and Bieman, J. (2014). *Software metrics: a rigorous and practical approach*. CRC press.

Garousi, V., Felderer, M., and Mäntylä, M. V. (2019). Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. *Information and software technology*, 106:101–121.

Greening, D. R. (2010). Enterprise scrum: Scaling scrum to the executive level. In *2010 43rd Hawaii International Conference on System Sciences*, pages 1–10. IEEE.

Greening, D. R. (2015). Agile enterprise metrics. In *2015 48th Hawaii International Conference on System Sciences*, pages 5038–5044. IEEE.

Grimaldi, P., Perrotta, L., Corvello, V., and Verteramo, S. (2016). An agile, measurable and scalable approach to deliver software applications in a large enterprise. *International Journal of Agile Systems and Management*, 9(4):326–339.

Heidenberg, J., Weijola, M., Mikkonen, K., and Porres, I. (2013). A metrics model to measure the impact of an agile transformation in large software development organizations. In *Int. Conference on Agile Software Development*, pages 165–179. Springer.

Hossain, S. S., Ahmed, P., and Arafat, Y. (2021). Software process metrics in agile software development: A systematic mapping study. In *Computational Science and Its Applications–ICCSA 2021: 21st International Conference, Cagliari, Italy, September 13–16, 2021, Proceedings, Part IX 21*, pages 15–26. Springer.

Kišš, F. and Rossi, B. (2018). Agile to lean software development transformation: A systematic literature review. In *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 969–973. IEEE.

Korpivaara, I., Tuunanen, T., and Seppänen, V. (2021). Performance measurement in scaled agile organizations. In *Proceedings of the Annual Hawaii International Conference on System Sciences*. University of Hawai'i at Manoa.

Korson, T. (2015). Cash-in comfort. `https://less.works/case-studies/cash-in-comfort`. (accessed 4 January 2023].

Laanti, M. (2014). Characteristics and principles of scaled agile. In *International Conference on Agile Software Development*, pages 9–20. Springer.

Leal, S., Hauck, J., Bertan, M., and Vieira, G. (2022). How agile organizations use metrics: A systematic literature mapping. In *Proceedings of the XXI Brazilian Symposium on Software Quality*, pages 1–11.

Liyanage, J. R. (2014). Lean agile metrics for scaled agile systems. `https://www.methodsandtools.com/archive/leanagilemetrics.php`. (accessed 4 January 2023).

Marinho, M., Camara, R., and Sampaio, S. (2021). Toward unveiling how safe framework supports agile in global software development. *IEEE Access*, 9:109671–109692.

Marinho, M., Noll, J., Richardson, I., and Beecham, S. (2019). Plan-driven approaches are alive and kicking in agile global software development. In *2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–11. IEEE.

Menezes, R., Marinho, M., and Sampaio, S. (2024a). Complete metrics tables for "goal question metrics" modeling. `https://doi.org/10.7910/DVN/HTQTEE`.

Menezes, R., Marinho, M., and Sampaio, S. (2024b). Replication data for: Metrics in large-scale agile software development: A multivocal literature review. `https://doi.org/10.7910/DVN/8MJG8K`.

Petersen, K., Feldt, R., Mujtaba, S., and Mattsson, M. (2008). Systematic mapping studies in software engineering. In *12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12*, pages 1–10.

Philipp, P., Tobisch, F., Menzel, L., and Matthes, F. (2023). A method for metric management at a large-scale agile software development organization.

Razzak, M. A., Noll, J., Richardson, I., Canna, C. N., and Beecham, S. (2017). Transition from plan driven to safe®: periodic team self-assessment. In *International Conference on Product-Focused Software Process Improvement*, pages 573–585. Springer.

SAFe (2022). Measure and grow. `https://scaledagileframework.com/measure-and-grow/`. (accessed 4 January 2023).

Scrum@Scale (2022). The scrum at scale® guide. the definitive guide to the scrum@scale framework. `https://www.scrumatscale.com/scrum-at-scale-guide-online/`. (accessed 4 January 2023).

Shameem, M., Kumar, C., Chandra, B., and Khan, A. A. (2017). Systematic review of success factors for scaling agile methods in global software development environment: A client-vendor perspective. In *2017 24th Asia-Pacific Software Engineering Conference Workshops (APSECW)*, pages 17–24. IEEE.

Staron, M., Meding, W., and Caiman, M. (2013). Improving completeness of measurement systems for monitoring software development workflows. In *International Conference on Software Quality*, pages 230–243. Springer.

Staron, M., Meding, W., and Palm, K. (2012). Release readiness indicator for mature agile and lean software development projects. In *International Conference on Agile Software Development*, pages 93–107. Springer.

Stettina, C. J. and Schoemaker, L. (2018). Reporting in agile portfolio management: Routines, metrics and artefacts to maintain an effective oversight. In *International Conference on Agile Software Development*, pages 199–215. Springer, Cham.

Tabib, R. (2013). Need 4 speed: leverage new metrics to boost your velocity without compromising on quality. In *2013 Agile Conference*, pages 117–120. IEEE.

Tessarolo, F., Petsani, D., Conotter, V., Nollo, G., Conti, G., Nikolaidou, M., Onorati, G., Bamidis, P. D., and Konstantinidis, E. I. (2022). Developing ambient assisted living technologies exploiting potential of user-centred co-creation and agile methodology: the captain project experience. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–16.

Thawaba, A. A., Ramli, A. A., Fudzee, M., Md, F., and Wadata, J. (2020). A mechanism to support agile frameworks enhancing reliability assessment for scs development: a case study of medical surgery departments. In *International Conference on Soft Computing and Data Mining*, pages 66–76. Springer.

Tripathi, N., Rodríguez, P., Ahmad, M. O., and Oivo, M. (2015). Scaling kanban for software development in a multisite organization: challenges and potential solutions. In *International Conference on Agile Software Development*, pages 178–190. Springer.

Wieringa, R., Maiden, N., Mead, N., and Rolland, C. (2006). Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requirements engineering*, 11:102–107.