

The Power of Scrum Mastery: An Analysis of Agile Team Performance and Scrum Master Influence

Hannes Salin^{1,2}^a, Felix Albrecht³ and John Skov³

¹Swedish Transport Administration, Department of Information and Communication Technology, Borlänge, Sweden

²School of Information and Engineering, Dalarna University, Borlänge, Sweden

³Independent Researcher, Copenhagen, Denmark

Keywords: Team Metrics, Agile Software Development, Decision-Making, Scrum Master, Agile Team Performance.

Abstract: We investigate the level of influence the scrum master role has on a team's success, in terms of four different leading indicators: feature lead time, defect leakage, predictability and velocity. We use statistical analysis on a large data set of agile metrics, collected by the target company, a large corporation in the Nordics. Our study showed that primarily the presence of a scrum master correlates well with team success for feature lead time. The remaining three indicators did not show strong statistical correlation with the inclusion/exclusion of a scrum master. However, these indicators should not be excluded for data-driven decision making, and need further research to identify obstructive or external factors that may influence either the scrum master's presence or the leading indicators' reliability.


1 INTRODUCTION

Adopting a scaled agile software development approach is a well used path for larger organizations today; although there are some challenges such as change resistance, many organizations seem to drive towards implementations of the Scaled Agile Framework (SAFe) and similar (Putta et al., 2018). Moreover, there are some indications that the adoption of SAFe can be demanding and expensive, from both a resource- and management perspective (Ciancarini et al., 2022). Therefore, the investment and understanding of different success - or leading - indicators are crucial. Agile ways of working, and in particular Scrum, have formed an industry de facto standard to face the challenges of working efficiently on software development projects (Ereiz and Mušić, 2019). One of the more important quests for the industry could then be described as finding ways to efficiently scale agile scrum teams, and understand what leading indicators drive the agile team towards success. A key role in the agile setup, using Scrum as an example of implementation, is the *scrum master*, who facilitates and support the developer team to success. In traditional settings, the team are part of a project, that is typically run by a project manager. Although there are differences between the scrum master's and project manager's roles, similarities exist

as well (Noll et al., 2017). Also, using agile project management may lead to several challenges (Marnada et al., 2022; Sadeh et al., 2022), thus having a role for the team that facilitates, lead and can provide support seems to be important. Moreover, using a data-driven decision making approach in the agile software engineering context, still have areas to research further; measuring traditional project key performance indicators may not be satisfactory but instead decision makers and stakeholders need more agile data to consider (Svensson et al., 2019).

1.1 The Scrum Master and Agile Software Development Teams

The scrum master (SM) role is typically defined to facilitate the team, the development process, removing impediments and ensure the adoption and utilization of agile principles and values (Noll et al., 2017; Ereiz and Mušić, 2019). Using scrum as an agile method for software development has been shown successful, and team performance is related to project success factors such as customer satisfaction (Kadenic et al., 2023). Although the adoption of an agile methodology seems to be one success factor in many cases, it also relates to many different factors such as roles, team composition, team maturity and more. Hoda and Nobel conclude several dimensions to consider to better understand why agile teams perform differ-

^a <https://orcid.org/0000-0001-6327-3565>

ently, when adopting the same agile ways of working (Hoda and Noble, 2017). However, the inclusion of a clearly defined scrum master role is not explicitly stated, although the drive towards autonomy and transitioning from a manager-driven to team-driven practice is highlighted.

1.2 Team Success Criteria and Metrics

Team success criteria in the software development context has been studied extensively; several case studies can be found, e.g. by Bogopa and Marnewick (Bogopa and Marnewick, 2022) which showed that commitment and well-defined goals are among some of the identified success criteria. Another comprehensive survey study was done by Lindsjörn et al. (Lindsjörn et al., 2016) which conclude that learning and work satisfaction had strong positive effects on team performance when perceived by the team members. However, compared to traditional (non-agile) teams, there was a lower level of agreement of what the evaluation of team performance should be defined. Depending on perspective, team success can be defined very differently. Therefore, for a better validity we need to clearly define what success criteria a team should aim for, in order to measure and correlate leading indicators for the success, in particular adjusted to what is relevant for the organization the team works in. Thus, using quality metrics for a software development project is useful when comparing team performance and quality; examples of such considered important metrics are customer satisfaction, lead time and test coverage (Salin et al., 2022; Maddox and Walker, 2021). These type of metrics, also referred to as leading indicators, may thus be useful for indicating output quality, and will be used in our study - in particular defect fixing measurements. Moreover, team performance is also considered to analytically correlate scrum master presence and influence with success. Agile performance metrics are usually associated to temporal factors such as lead time to provide a value (as in programming code) from idea to production, also known as lead time (Weflen et al., 2022).

1.3 Problem Statement

In order to answer the question of *how* a high performing agile software development team can be defined, in terms of agile characteristics, we have identified a research gap in understanding how the scrum master role can influence different leading indicators in order to reach team success. With the strict notion of success, defined as the temporal metrics *feature lead time* (FLT), *velocity* and *predictability*, together with the

quality notion of *defect leakage*, we investigate what primary leading indicators a team is sensitive for and what (if any) correlations there are between such indicators and the inclusion or exclusion of a SM in the team.

1.4 Contributions

Our contribution consists of the following:

- A statistical analysis on team success and leading indicators, using historical agile data from the target company.
- Without generalization, we provide insights of what leading indicators can be useful for team success evaluation in the (scaled) agile context.

1.5 Organization of the Paper

The paper is outlined as follows: in section 2 we describe related work, and in section 3 we outline the research method and describe the target company and data collection methods. In section 4 we provide our results, including the statistical analysis. Finally, in section 5 we summarize and conclude our analysis aligned with our problem statement.

2 RELATED WORK

Ereiz and Mušić investigated the potential risk of not having a dedicated SM working in an agile team (Ereiz and Mušić, 2019). By using interviews and online surveys with several different companies they concluded that agile projects in the form a Scrum, most likely lead to a higher level of project success if a SM is part of the team. However, no strong indication of the underlying reasons why the SM is of that level of importance was concluded.

The multifaceted role of the SM role was studied using grounded theory (Shastri et al., 2021). From a comprehensive set of interviews and analysis, it was concluded that it appears there is a positive association between several agile practices (e.g. definition of done, sprint planning, velocity measurements etc.) and the presence of a SM in the team. Furthermore, it appears that as a SM, a significant amount of time is devoted to facilitating and mentoring. While these dimensions may impact the team's overall performance, additional investigation is required to evaluate their effectiveness and compare them to established success criteria. Another angle on the topic was elaborated by Strode et al., where agile leadership seemed to not to be necessarily in one role but shared (Strode

et al., 2022). Their study proposed a team effectiveness model for agile teams, ATEM, which includes different coordinating mechanisms and components, where shared leadership is one such component.

A previous study conducted in 2017, Kristensen and Paasivaara used a quantitative research method, investigating the value received by assigning a SM to the developer team (Kristensen and Paasivaara, 2021). They concluded that the real value of a SM was based on the ability to understand other people. Moreover, assigning an experienced SM to two teams may lead to improved knowledge-sharing and cooperation, however, assigning even more teams to a single SM may increase the risk of not being able to return the expected value from the SM. Our work therefore build upon the inspiration of Kristensen's and Passivaara's initial work, whereas our paper address a slightly different problem statement, namely the investigation of correlating the presence of a SM and team success in terms of pre-defined success criteria.

3 RESEARCH METHOD

In this section we outline the used research method, including a target company description and the defined leading indicators (the success criteria).

We use a quantitative research method on historical data collected over a span of 18 months. The numerical data was continuously collected in the organization using Jira (Atlassian, 2022), a well-known (agile) project tracking software. The overall research design consisted of three primary phases: a data collection phase, a data filtering phase followed by the analysis phase. The data set consisted of Jira items with meta data such as type and time information. These items on an aggregated level and filtered into team-levels, provide several agile performance-related leading indicators, shown in Tab. 1. The analysis phase consisted of statistical testing, using *t*-testing in particular, for investigating differences between the team performance data of SM-driven teams and non-SM-driven teams.

3.1 Success Criteria

The chosen success criteria for our study included good levels of feature lead time, defect leakage, velocity and predictability. These are the indicators measured and analyzed in the data set. For each indicator the target company defined thresholds which would be considered success if the indicator reached that level or above. These indicators are formulated as follows:

Feature Lead Time: the time for an idea from being shared, analysed, developed, tested, deployed and released. FLT thus shows how long it takes from a request to production. The shorter the lead time the less the customer or market need to wait, and the faster adaptation the organisation have, the lower the risk and cost of missed market opportunities; lead time is a well established standard metric (Petersen, 2010).

Predictability: the accurateness of a delivery forecast. In our context, this refers to a team's work items forecasting, i.e. completed features within a given time frame. The target company defined that good predictability should improve over time and stabilize in the 80 -100% area. However, projects may change and need to be adjusted over time, hence there are reasons for not seeking to completely fix the level of predictability.

Defect Leakage: is defined as the defect findings in a staging environment set up as this context uses it. In particular the avoidance of defects in production are of interest, and this metric shows the quality enhancement and possibly the learning work from staging deployments. With a steady amount of releases, a decrease in defect findings over time in each staging environment would indicate increased quality engineering and higher uptime of the deployed product.

Velocity: is defined as the aggregated amount of time between already planned and described features, from their starting state to complete state within a fixed time period. The target company uses a time span of 12 weeks. The main difference between velocity and FLT is that velocity only consider the number of completed features within the fixed time, hence not considering if the features has been in the backlog for a long time or not before started.

3.2 Research Question

Our focus is on the influence a SM may have on a team's success, and what relation may exist between a set of the target company's chosen leading indicators and that influence. We formulate our research into the following research question:

- **RQ:** *What relation is there between having a scrum master role in a team and feature lead time, predictability, velocity and defect leakage?*

Our research concerns only the context of the target company and should not be considered to be a generalization.

Table 1: Description of the defined indicators including the pre-defined criteria for success, set by the target company.

Data	Description	Quantity	Criteria
Feature lead time	The time between a feature is in the <i>funnel</i> state and <i>closed</i> state.	Number of days	Less than 90 days.
Defect leakage	The number of defects identified in a pre-defined time span.	Number of defects	No absolute value set; when leakage is trending downwards.
Predictability	The amount of features completed from a pre-defined set of committed features per program increment, i.e. 3 months.	Number of features	80 – 100%
Velocity	The amount of features completed within a program increment, i.e. 3 months.	Number of features	No absolute value set.

3.3 Target Company Description

The target company is primarily located in the Nordics with software development teams both co-located and distributed; Sweden, Norway, Denmark, Finland (3 sites), Poland (2 sites) and India (4 sites). With the current implementation of SAFe and portfolio structure, the teams are divided into E2E teams, platform teams, enabler teams and complicated sub-system teams. In total 45 teams were included in the study where 21 teams, denoted G_{SM} , had an associated SM, and 24 teams, denoted G_{NONE} without a SM. The set of all teams is denoted G_{ALL} . Team sizes vary between 3-14 members, with the majority being on the range 5-9 members. In total G_{SM} included 197 individuals, and G_{NONE} included 207 individuals respectively. The backlog items of the team vary from independent deliveries to the majority being highly interdependent. The environmental setup for the teams, regardless of location, are the following:

- There is a *product manager* (PM) of product owners (PO) available.
- If the team have a SM, he/she attend to regular Scrum of Scrums meetings.
- Each team works in the same work management tool (Jira) on the same level of abstraction; epics, capabilities, features and user stories.
- Each team synchronize 2 week long sprints for 5 iterations before having an aligned innovation, inspect and adapt, demo and planning sprint of 2 weeks.

Due to the SAFe structure, all teams are organized into different *agile release trains* (ART), sharing overall backlogs and work towards the planned portfolios. The main purpose of an ART is to provide a contin-

uous flow of value to the organization via the virtual organization of agile teams. In the target company, a train is not bounded geographically and teams located in different countries can be part of the same ART. The company’s tool for working is Jira and the teams are using user stories and defect items which are children items to *feature* items who are owned by trains and capture the problem statement and its benefit when solved. These items are handled in Jira by the teams, and are synchronised in sprints of 2 weeks with exception of the summer and Christmas break. The year is divided into 4 program increments (PI) of 12 weeks cadence.

3.4 Data Collection

Jira is used for work and updated accordingly for information and status for enabling best collaboration on the prioritised problems. The updates are done manually by the team. Our data collection consisted of querying Jira for all registered meta data in the portfolios, including user story, feature and epic items. Jira also provide temporal data such as time stamps in the item workflows, i.e. when items change state, e.g. from state *in progress* to *done*.

The four metrics we collected was handled as follows by the teams:

Feature Lead Time: is updated with every update of state in the feature flow. Some features may have a slower flow than expected due to updating lag, however feature update from status *implement* to *validating* are automated if all children stories are moved to status *done*. The FLT is taken only when the feature is finalised. The FLT data is taken as a median or mean per PI. The data points are counted in the same PI the feature was put to status *done*, i.e. where it was finalised.

Predictability: is taken every PI closure as in number of features done vs committed to be done in the PI planning 12 weeks earlier. The data point is taken at the end of each PI as the percentage of the quotient of features done divided by features committed to.

Defect Leakage: Defects are created and categorised based on environment (e.g. pre-production or production) and other details on the findings related to how it can be resolved. A found defect may be communicated by the team, external users or via the PO, and usually it is the team that insert the defect into Jira. Defect data points are added per finding in the PI and later aggregated.

Velocity: is updated with every feature update into state *done*, released during a the PI of their closing, i.e. the difference in time stamps between the starting point of the feature's implementation status and complete closure.

3.5 Data Filtering

After collecting all meta data from Jira and exported into data sets, we applied data filtering according to the following criteria:

1. If there is a dedicated person as a SM more than 50% of the time it is categorized as G_{SM} , otherwise G_{NONE} .
2. If the team was not delivering features of working code it was excluded from the data set.
3. If the a team had less than 3 members over a whole year it was excluded from the data set.

The final data filtering was exported into a single data set for the analysis phase. Moreover, the absolute values of the data was converted into percentages for privacy reasons.

3.6 Statistical Analysis

In order to ensure statistical significance we used the t -test on the data sets. The value for significance set at $\alpha = 0.05$ (statistically significant) and $\alpha = 0.01$ (statistically highly significant). The t -test is used to compare means between two data sets to test whether two groups are different from one another. The two tailed t -test is used to indicate if one group is deviating from the other group in any direction, while the one tailed t -test is used to indicate if there is a deviation in one direction. This study uses the two tailed t -test for testing the significance of difference between a subset of the team data set with the whole data set. We then use

the one tailed t -test for testing the significance of difference between two subsets (partitions of the whole set G_{ALL}), namely G_{SM} and G_{NONE} . If the p -value is below 0.05, significant difference can be claimed.

4 RESULTS

In all results, the analyzed time period was 18 months (the x -axis), which corresponds to 6 PI:s (PI 14 to PI 20), each being 12 weeks long. Results in Fig. 1 show the level of predictability for the amount of features done from a pre-set of committed features per PI, i.e. 3 months, over the total length of 6 PI:s. The blue line G_{ALL} is the whole data set with all teams, while the orange line is the subset G_{SM} and the grey line is the subset G_{NONE} . The result is presented in percentage on the y -axis. The predictability results of the whole set shows that 5 out of 6 PI occurrences lies within the aimed 80 – 100% accuracy, with a slight downward trend. Overall, this results clearly shows a solid performance on an aggregated level. Both teams with and without SM, reach this target 4 out of 6 times with slight variations in time.

Fig. 2 shows median lead time for the number of features done per PI, i.e. 3 months, over 6 PI:s. The median is standardized against the company's FLT target value. The unit on the y -axis is shown in percentage of projected days to complete the features divided by the target value. The FLT ranges from slightly below 100 to 160 percentage of the company's target value, while being trending downwards in the total data set as well as the G_{SM} and G_{NONE} subsets. With the exception of PI 16 and 17 these subsets are moving in lockstep. PI 16 has a raise for G_{NONE} , which seems to precede the G_{SM} data set raise in PI 17. The results in Fig. 3 show defect leakage for production defects divided by test environment defects per PI, i.e. 3 months, over 6 PI:s. The result is shown in percentage on the y -axis. The results indicate that

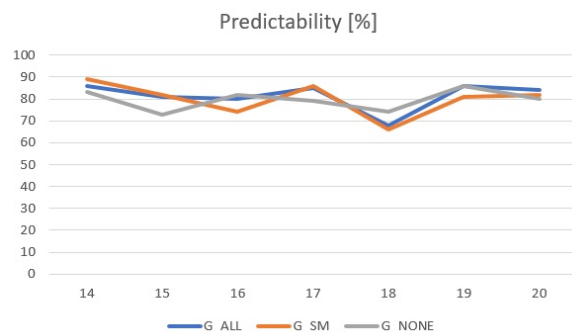


Figure 1: Feature predictability filtered on G_{SM} and G_{NONE} , measured in achieved percentage.

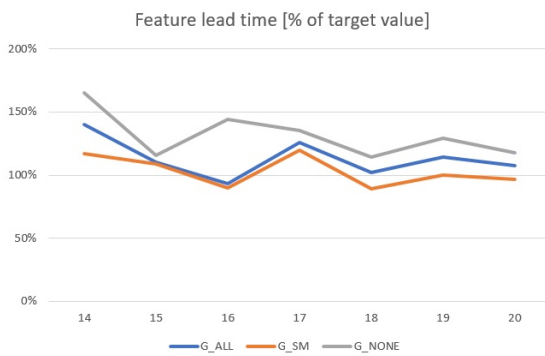


Figure 2: Feature lead time filtered on G_{SM} and G_{NONE} .

there may be a quality progress in the staging work by reducing the amount of defects found over time. One interpretation would be that both team subsets shows an improving trend of reducing the amount of defect in their staging over time.

In Fig. 4 we show the result as in the number of defects, as a percentage compared to the defects of the whole set in PI14 G_{ALL} (PI14), on the y-axis. The number of defects are roughly stable with an exception of PI 14 and 19, being somewhat lower and a slight overall downward trend.

Results in Fig. 5 show delivered features per PI, i.e. 3 months, over 6 PI:s. The result is shown in the number of features compared to the G_{ALL} (PI14) velocity. The velocity shows an upward trend with the exception of a drop in PI 18 which was related to a down-scaling organisational change which affected only teams with SM.

4.1 Comparative Analysis

In this section we provide a comparative analysis of the results presented in Sec. 4. If we compare the variations of predictability shown in Fig. 1 there seems to be no relation. Comparing the de-facto value per timestamp, the two subsets are not showing a clear difference or related pattern. Both G_{SM} and G_{NONE}

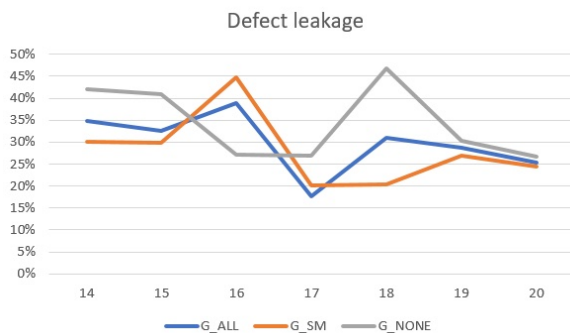


Figure 3: Defect leakage ratio from test to production environments filtered on G_{SM} and G_{NONE} .

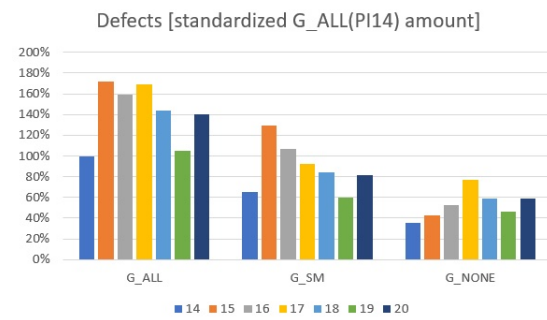


Figure 4: Absolute number of defects as percentage compared to the total number of defects found, filtered on G_{SM} and G_{NONE} over PI14 to PI20.

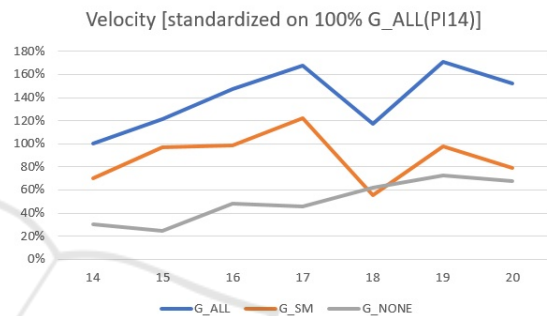


Figure 5: Feature velocity per PI filtered on G_{SM} and G_{NONE} .

have better and worse values in comparison to each other and to the total set. Moreover, the total FLT seems to be more related to the G_{SM} data set which makes sense as the SM-teams deliver more features per PI (seen in Fig. 5).

Comparing FLT between G_{SM} and G_{NONE} , the graphs suggest that team with SMs have consistently a faster FLT than non-SM teams. We computed a p value of 0.002301 in the statistical analysis, which indicates high significance. Even comparing the SM teams with the total set of teams' FLT has a p -value of 0.011899, i.e. a significant difference. Considering that the target company expectation is a FLT represented in the graph as 100% and below, only the teams with a SM achieve that intent in 4 out of 6 PI:s.

When comparing the defect leakage between G_{SM} and G_{NONE} , the teams with a SM are with the exception of one PI, ahead two times rather close distance, and the other PI:s up to 25% less leakage. However, the statistical analysis with the computed p -value of 0.125383 shows the difference as non-significant. Next, comparing the two subsets, G_{SM} have a higher amount of defects than the teams without a SM, which could make sense considering that teams with a SM produce more features; see Fig. 5. The noteworthy difference is the downward trend for teams with a SM compared to stable or even a slight

upward trend for teams without SM. This data could suggest a higher ability to learn and improve in teams with SM compared to non SM, but would require a more qualitative analysis for validation.

Finally, when comparing the velocity between the two groups, we note that it can be difficult to make clear interpretations. It is more difficult compared to the other metrics, since velocity is related to the number of teams and people, and therefore can potentially be faulty expressed in "more people - more features". However, in the time period for the collected data, approximately the same amount of individuals (around 200) for both groups were included. We chose to take a more cautious stand towards this metric since it is fairly easy to "game" the data by teams that want to please a stakeholder by closing the desired number of features that is expected, although all tasks are not completed (Levison, 2022).

4.2 Discussion

We have collected and analyzed data which can be used to resolve the RQ. Our concluding analysis is the following: considering the pre-defined team success criteria and comparing how teams with a SM have performed, versus teams without a SM, the major significant difference is in the FLT indicator, where teams with a SM deliver significantly faster than teams without a SM. Therefore, it shows a higher degree of team success when measured against FLT; teams with a SM thus succeed in the company defined ambition of delivering enough amount of features within one PI. Nonetheless, the data does not offer insights into the effectiveness of an SM's role execution. Consequently, a more detailed examination of SM performance could potentially yield a finer and more precise analysis. Moreover, the threshold of defining a team in G_{SM} as a team with a SM at least 50% of the time would also affect the analysis if considering the SM performance as well, hence we leave that type of analysis as proposed future work.

Predictability seem to be related between the two groups and both groups also succeed most of the time with the ambition of delivering in the range of 80 – 100% of committed features. However, team success in terms of predictability seems to not be related to the SM role. For the defect leakage criteria, there seemed to be a stronger relation for teams with a SM to perform more successfully, but the difference was statistically non-significant. Again, team success seems to not be related to the SM role. However, looking at the total amount of defects, the data may suggest that the SM role enhance the teams ability to learn (by fixing defects) in terms of decreasing

defects, which leads to better results in defect leakage particularly. We hypothesize that this metric also imply a higher level of product quality and other positive downstream effects.

To summarize, G_{SM} have clearly a benefit in terms of FLT as a success criteria and should therefore be considered when building and developing agile software development teams. Moreover, we have identified further areas to be investigated using qualitative research, such as the ability to learn from previous defect fixing and what factors that hinders or incubates the learning. From the perspective of data-driven decision making, our findings offer valuable information on how decision makers in the delivery organization, such as project- and management levels, can gather and utilize the analyzed indicators.

5 CONCLUSION

In our study the role of a SM had a significant impact on the team success in terms of delivering features, at the target company. We also see indications of what could be the increased ability of learning for developing software with higher quality in teams who were facilitated by the role of a dedicated SM, however, to validate this further, a qualitative study is necessary. Despite no strong correlation was found for predictability and defect leakage, with having a SM or not in the team, these metrics are still valid and important metrics to consider since they provide insights to progress and development of the team. We thereby conclude that a SM is still a key component of a team, and that all investigated success factors give valuable insights of teams' performance, but FLT is the most important to consider in terms of success criteria.

REFERENCES

- Atlassian (2022). Atlassian jira software.
- Bogopa, M. E. and Marnewick, C. (2022). Critical success factors in software development projects. *South African Computer Journal*, 34(1):1–34.
- Ciancarini, P., Kruglov, A., Pedrycz, W., Salikhov, D., and Succi, G. (2022). Issues in the adoption of the scaled agile framework. In *2022 IEEE/ACM 44th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pages 175–184.
- Ereiz, Z. and Mušić, D. (2019). Scrum without a scrum master. In *2019 IEEE International Conference on Computer Science and Educational Informatization (CSEI)*, pages 325–328. IEEE.
- Hoda, R. and Noble, J. (2017). Becoming agile: A grounded theory of agile transitions in practice. In

- 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE), pages 141–151.
- Kadenic, M. D., Koumaditis, K., and Junker-Jensen, L. (2023). Mastering scrum with a focus on team maturity and key components of scrum. *Information and Software Technology*, 153:107079.
- Kristensen, S. H. and Paasivaara, M. (2021). What added value does a scrum master bring to the organisation? — a case study at nordea. In *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 270–278.
- Levison, M. (2022). The trouble with sprint burndowns - are sprint burndown charts really helpful?
- Lindsjörn, Y., Sjöberg, D., Dingsøy, T., Bergersen, G., and Dybbå, T. (2016). Teamwork quality and project success in software development: A survey of agile development teams. *Journal of Systems and Software*, 122.
- Maddox, M. and Walker, S. (2021). Agile software quality metrics. In *2021 IEEE MetroCon*, pages 1–3.
- Marnada, P., Raharjo, T., Hardian, B., and Prasetyo, A. (2022). Agile project management challenge in handling scope and change: A systematic literature review. *Procedia Computer Science*, 197:290–300. Sixth Information Systems International Conference (ISICO 2021).
- Noll, J., Razzak, M. A., Bass, J. M., and Beecham, S. (2017). A study of the scrum master’s role. In Felderer, M., Méndez Fernández, D., Turhan, B., Kalinowski, M., Sarro, F., and Winkler, D., editors, *Product-Focused Software Process Improvement*, pages 307–323, Cham. Springer International Publishing.
- Petersen, K. (2010). An empirical study of lead-times in incremental and agile software development. volume 6195.
- Putta, A., Paasivaara, M., and Lassenius, C. (2018). Adopting scaled agile framework (safe): A multivocal literature review. In *Proceedings of the 19th International Conference on Agile Software Development: Companion, XP ’18*, New York, NY, USA. Association for Computing Machinery.
- Sadeh, A., Rogachevsky, K., and Dvir, D. (2022). The role of the project manager in the agile methodology. In *2022 Portland International Conference on Management of Engineering and Technology (PICMET)*, pages 1–5.
- Salin, H., Rybarczyk, Y., Han, M., and Nyberg, R. G. (2022). Quality metrics for software development management and decision making: An analysis of attitudes and decisions. In Taibi, D., Kuhrmann, M., Mikkonen, T., Klünder, J., and Abrahamsson, P., editors, *Product-Focused Software Process Improvement*, pages 525–530, Cham. Springer International Publishing.
- Shastri, Y., Hoda, R., and Amor, R. (2021). Spearheading agile: the role of the scrum master in agile projects. *Empirical Software Engineering*, 26(1):3.
- Strode, D., Dingsøy, T., and Lindsjörn, Y. (2022). A teamwork effectiveness model for agile software development. *Empirical Software Engineering*, 27(2):56.
- Svensson, R. B., Feldt, R., and Torkar, R. (2019). The unfulfilled potential of data-driven decision making in agile software development. In Kruchten, P., Fraser, S., and Coallier, F., editors, *Agile Processes in Software Engineering and Extreme Programming*, pages 69–85, Cham. Springer International Publishing.
- Weflen, E., MacKenzie, C. A., and Rivero, I. V. (2022). An influence diagram approach to automating lead time estimation in agile kanban project management. *Expert Systems with Applications*, 187:115866.