Success Factors for Effective Process Metrics Operationalization in Agile Software Development: A Multiple Case Study

Prabhat Ram, Pilar Rodriguez, Markku Oivo
M3S, Faculty of Information Technology and Electrical Engineering,
University of Oulu
Oulu, Finland
{firstname.lastname}@oulu.fi

Silverio Martínez-Fernández
Fraunhofer IESE
Kaiserslautern, Germany
silverio.martinez@iese.fraunhofer.de

***Abstract*— Existing literature proposes success factors for establishing metrics programs. However, very few studies focus on factors that could ensure long-term use of metrics, and even fewer studies investigate such factors in the context of Agile Software Development (ASD). Motivated by this knowledge gap, we aim to identify success factors for operationalizing metrics in ASD, particularly, factors that could help in the long-term use of metrics. We conducted a multiple case study, where we operationalized process metrics at two software-intensive companies using ASD. We learned that data availability and development process are the two fundamental success factors for process metrics operationalization, albeit less prominent in literature. Companies prefer iterative and incremental operationalization of stable and functional process metrics, which is analogous to the agile way of working. Metrics trustworthiness plays a key role in successful operationalization of process metrics, and is potentially vital to ensuring their long-term use. By comparing the identified success factors with the existing literature, we conclude that success factors concerning data availability, development process, and metrics trustworthiness warrant greater attention, especially to maximize the chances of long-term use of process metrics.**

Keywords—process metrics, trustworthiness, agile, GQM

I. INTRODUCTION

Software metrics programs enable objective and quantitative evaluation of software processes, leading to continuous improvement and learning in industrial software development context [1]. Many success factors have been proposed to facilitate establishment of such programs [2]–[4], but adherence to these factors does not guarantee success [5], as more than 80% of these programs fail within the first 18 months [6]. There have been several studies that recognize the contribution of metrics programs in Agile Software Development (ASD) [7]. Owing to continuous production of data in ASD processes, metrics are useful in identifying and fixing process problems [7]. Overall, metrics programs in ASD can be help in planning, understanding and improving software quality, fixing software process problems, and motivating people [7]. Despite the importance of metrics in ASD, studies that identify success factors for a metrics program in ASD are scarce [7]. There are even fewer studies about factors that can help sustain these programs in ASD.

Recommendations for a successful metrics program in [2], [3], and [4] cover a wide gamut of factors, but their focus is only on the success factors for a metrics program's implementation. Long-term feasibility of such programs remains debatable, though, as not much supporting literature is available. Moreover, most of these studies were carried out when modern software development methods like ASD were not as prevalent as they are now. One of the few known studies in this regard is by Staron and Meding [8], where the authors recommend 11 success factors for a metrics program's long-term success in ASD. A dearth of related studies and the need to add to this body of knowledge serve as the primary motivation for this paper. With this guiding motivation, we devised the following Research Question (RQ): *What factors characterize effective process metrics operationalization in software-intensive companies using ASD?* By 'effective operationalization', we mean not just successful development and deployment of process metrics but also identifying factors that could be conducive to their long-term use.

We conducted our research in the context of the H2020 Q-Rapids[1] project, which aims at developing an agile-based, data-driven, quality-aware rapid software development framework [9]. This grants us a suitable opportunity to study the factors needed for a successful metrics program in ASD. We conducted a multiple case study at two Q-Rapids industrial partners, which are software-intensive companies using ASD. We built upon the progress made in [10], where we helped the industrial partners to elicit process metrics using the Goal-Question-Metric (GQM) paradigm [11]. The subsequent collaborations to operationalize these metrics, to guide their process improvement actions in the context of Q-Rapids, is the focus of this multiple case study. Our main contributions are:

- Providing an empirical account of operationalizing metrics in software-intensive companies using ASD.
- Complementing existing body of knowledge concerning success factors for metrics programs by identifying new factors that deserve consideration, such as data availability and metrics trustworthiness.
- Identifying success factors that could positively influence a metrics program's long-term use in ASD.

In the remainder of the paper, Section II covers background and related work on the research topic, with Section III describing the research methodology, followed by the multiple case study results in Section IV, and discussion in Section V. Section VI discusses the threats to validity, with conclusion and future research directions in Section VII.

II. BACKGROUND AND RELATED WORK

We first provide an overview of metrics in ASD. Following that, we provide an account of the state of the art, and the reasons why our research is a distinct addition to that body of knowledge.

[1] https://www.qrapids.eu/

A. Metrics in ASD

Kupiainen et al. [7] provides a systematic review of the use and impact of software metrics in industrial ASD. The authors report that metrics are used for sprint planning, tracking progress, improving software quality, fixing software process, and motivating people. Metrics like 'velocity' and 'effort estimate' are very popular in ASD, but companies also use custom metrics to measure aspects such as business value, defect count, and customer satisfaction. Furthermore, the use of metrics in ASD is similar to that in traditional software development, as the emphasis in both is on planning and monitoring.

Operationalizing metrics programs is challenging [5], [6], and so it follows that it will be challenging in ASD, too. Therefore, it is important to identify and understand success factors for operationalizing metrics programs in ASD.

B. Related Work

Broadly, studies reporting success factors for establishing metrics programs either validate or complement one another. Still, very few studies emphasize on factors needed to ensure long-term use of these programs.

Mendonça and Basili [2] propose that a measurement framework (same as metrics programs) should be sound (adhere to applicable environment), complete (user-goal oriented), lean (measure only what is needed), and consistent (comply with user goals). Aligning with the authors' recommendation of using GQM to address above requirements, the GQM paradigm was applied to elicit metrics in the Q-Rapids project, too. Our multiple case study is a step further into investigating success factors for operationalizing metrics programs, and assess if any of these factors could help the program's long-term use.

Hall and Fenton [3] compiled a consensus of 11 success factors for metrics programs from the literature. Based on a multiple case study, the authors found that these factors applied in their studied cases as well. Additionally, they emphasize three points that contribute to a metrics program's success: (i) an organization's willingness to research other metrics programs; (ii) acknowledge developers' concerns and involve them; and (iii) use incremental implementation. Similarly, Iversen and Mathiassen [4] report six success factors from a case study of an organization that implemented a metrics program to measure the effects of its improvement efforts. Some of the success factors call for an organization to pilot the program in a real project, start simple, and widely publish feedback on the quality and relevance of the program. The idea of feedback has also been discussed in [1], claimed to be one of the critical success factors, promoting a culture of learning. The Q-Rapids solution addresses most of these success factors through its value additions [9], [10], [12]–[14], discussed later in Section V. Our attempt is to determine if these factors indeed play a role in operationalizing a metrics program, or if different factors have emerged owing to the modern-day software development methods.

The study by Staron and Meding [8] is among the few studies that investigate factors for long-term use of metrics programs. The authors studied a five-year old metrics program at Ericsson AB, where projects were often executed using Agile and Lean principles. The authors recommend 11 success factors, discussed later in Section V. Commenting on a metrics program's long-term survival, Niessink and Vliet [5] believe that it is critical for a program to generate value rather than just data. The requirement of value proposition, as positioned in [5], is the cornerstone of the Q-Rapids project. One of the objectives of our multiple case study is to identify factors that also appear in literature like [8], for their potential role in long-term use of metrics programs.

Excluding [8], none of the above-discussed literature characterize their study's context as involving companies using ASD. Similarly, most primary studies (case studies) in a recent systematic literature review on metrics in ASD [7] focus on implementing metrics programs and measuring their impact in organizations using ASD. None of these studies reports on the factors that aid in successful operationalization of metrics programs, which is the primary objective of our multiple case study. Similarly, except [8], none of the above discussed studies reflects on success factors for long-term use of metrics programs. Our multiple case study takes into account all of the above concerns in investigating factors for operationalizing a metrics program in ASD, and potentially facilitating its long-term use.

III. Research Method

We followed the guidelines recommended by Runeson and Höst [15] to conduct and report the multiple case study. The accompanying semi-structured interviews were analyzed using thematic synthesis [16].

A. Research Setting

The goal of the EU Horizon 2020 Q-Rapids[2] project is to develop an agile-based, data-driven, quality-aware rapid software development framework [9]. The framework includes tools and methods to elicit and manage quality requirements. The solution enables a systematic and continuous assessment of software quality based on a set of strategic indicators that are derived using GQM+Strategies™ [17], Quamoco [18], and GQM [11]. In the solution, process metrics enable assessment of process performance for making data-driven process related decisions. The solution also includes an informative dashboard highlighting indicators concerning product quality, process performance, on-time delivery, product readiness, and blocking (conditions blocking project progress) [12]. The objective of this solution is to help practitioners in making data-driven decisions in rapid cycles.

B. Case Study Context

The following table presents the software development context at the two case companies where the Q-Rapids solution is piloted:

TABLE I. Case Study Context

Context Characteristic	Case A	Case B
Size	Medium	Large
Product	Production testing software framework	Software modeling tool
Development Method	Scrum	Customized Agile
Development Team	Single team (6-7 members)	Single team (9 members)

Case A, a medium-size company (over 600 employees), develops secure communication and connectivity solutions for

[2] https://www.qrapids.eu/

multiple industry domains. To achieve efficiency and reduce time to market, the company adopted Agile and Lean software development practices over a decade ago. Using software metrics, the company wants to measure ASD process to introduce high-level transparency, and a more data-driven and evidence-based decision-making process. The company is piloting the Q-Rapids solution on a project to develop a hardware-testing framework, used by the development team to test secure solutions that the company develops.

Case B is a large-size company (over 900 employees) that develops a modeling tool for model-driven development. The product is mature, with multiple releases already in the market. The company aims to improve the quality of its ASD process through early detection of anomalies in the development. The company uses various software development methods that adhere to the Agile principles. They engage in iterative development, but do not have any pre-defined sprint cycles. A single team working on the modeling tool is piloting the Q-Rapids solution.

C. Data Collection

Multiple GQM workshops were conducted to elicit and define process metrics. This process and the related scientific contribution is reported in [10]. In the context of operationalizing process metrics from [10], we collaborated with the two case companies for five months between June 2018 and October 2018. Here, operationalization includes process metrics redefinition (if needed), development, and deployment, along with regular feedback. Case A is co-located, and so we collaborated with them at their premises. However, Case B is located overseas, and online collaboration was the only feasible option. Table II provides details of these collaboration sessions, where both the researchers and practitioners worked together as a team to refine, develop, and deploy the process metrics identified in [10].

TABLE II. COLLABORATION SESSIONS

Collaboration Characteristics	Case A	Case B
Total number of sessions	Six	Three
Designation of the collaborating practitioner in the Q-Rapids project	Lead Developer / Lead Technical Contact / Use Case Champion	Product Development Team Lead/Use Case Champion
Average length per session	2 hrs.	40 min.

First, we transformed the raw process metrics presented in [10] into assessed metrics. An assessed metric is a combination of two or more raw metrics. Raw metrics and assessed metrics are analogous to base measures and derived measures as defined in ISO/IEC 15939 [19]. Here, we studied the data sources to ensure that the data needed to compute metrics were collected. There were requirements for new 'connectors' (automatic data collection plugin) [12], [13] for certain tools, which the case companies then developed. The practitioners would then either approve or suggest changes to process metrics, which we would address, and then repeat the cycle. During this process and based on data availability, some process metrics were redefined, and even dropped or postponed.

We started developing the process metrics only after the practitioners greenlit all the assessed metrics. We developed the metrics using Query Domain Specific Language [3], and this process was iterative, similar to raw metrics to assessed metrics transformation described above. Once the practitioners were satisfied with the developed metrics, they deployed them to make sure they work in their project context. We collaborated to fix any issues that surfaced during this deployment. After the practitioners were convinced of the metrics' functionality and stability, they piloted the process metrics in their respective projects mentioned in Table I. The list of process metrics operationalized in each case company is available in *Appendix A* (https://goo.gl/tsXkrK)

Collaboration with both the case companies led to many informal exchanges of ideas and opinions, including feedback. We also shared information over many emails that were exchanged during the course of process metrics operationalization.

After both case companies had piloted the process metrics in their respective projects, we produced an initial result document listing the factors we observed that had influenced process metrics operationalization during our collaborations. Although the observations made were similar for both the case companies, we prepared case-specific results to leave room for interpretation. We shared the results with the two case companies, and requested a one-hour interview to test the validity of our results, and gain further insights. Table III provides the details of these interviews conducted at the end of our collaborations.

TABLE III. INTERVIEW DETAILS

Parameters	Case A		Case B	
	Int1[a]	*Int2*	*Int3*[a]	*Int4*
Total experience	12 years	25 years	12 years	20 years
Experience at the current company	3 years	20 years	12 years	6 years
Company Designation	Principal Engineer	Head of Quality Environment	Product Development Team Lead	R&D Head
Designation in Q-Rapids	Lead Developer, Lead Technical Contact, Use Case Champion	Project Manager	Use Case Champion	Liaison between R&D and Q-Rapids project
Interview length	46 min.	33 min	37 min	
Mode of Conduct	Face to face		Teleconferencing	

[a] Practitioners with whom we collaborated for process metrics operationalization.

The interviews had two objectives: to validate the factors we observed during collaborations, and generate additional insights about process metrics operationalization and long-term use. With these objectives in mind, we chose semi-structured interviews for data collection [20]. Since there were slight differences in the results reported for the two case companies, we had to adapt some of the interview questions.

One researcher transcribed the interview and performed line-by-line coding using NVivo™. The researcher recorded the concepts related to the RQ, and coded them at a higher

[3] https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html

abstraction level to develop descriptive themes. Descriptive themes aggregate the codes into a smaller analytical unit. Based on their interrelationships, the researcher translated these descriptive themes into higher-order analytical theme(s). The analysis results were shared with the other three researchers for validation and feedback.

IV. RESULTS

We report the factors (along with an ID: F#) supporting effective process metrics operationalization in the two case companies, which are also described in Table IV. We elaborate on these factors below, along with illustrative quotes from the interviewees to support the corresponding claims. For further details, the complete thematic synthesis of the interviews data is available in *Appendix B* (https://goo.gl/tsXkrK).

A. F1: Data

Data appears to be one of the fundamental factors to drive process metrics operationalization. A company may believe to have the relevant data while defining metrics, but it is only during operationalization that other mitigating conditions surface. These conditions govern the utility of the data for measurement purposes, elaborated below as sub-factors:

1) Data availability

A basic requirement for metrics operationalization is the availability of relevant data. Assessment of data availability during metrics operationalization should supplement the preliminary assessment of its availability during metrics definition. For example, *Int1* informed how some data started to be available only after its identification in the beginning of the metrics program, "*SonarQube has not been, like very systematically used (at) the company before. So, we started using, really using it more systematically right now*". Lack of systematic use of the tool hindered metrics development, as the necessary data was not available. This tool is used to analyze code quality, and is used mainly for product metrics. However, the underlying rationale of systematic tool use affects data availability, which influences metrics operationalization, be it product metrics or process metrics.

Data availability should dictate what metrics to target. *Int2* clarifies, "*...you tend to focus on the must-have features*". By drawing a parallel with the Scrum method, the interviewee was suggesting that the data available should decide what metrics a company must have before moving on to the metrics that would be nice to have. *Int3* had a similar response, "*The main factor that have influenced the choice of metrics was having clean data, not so much the format, but just in terms of... data are available in the company*".

Unrestricted access to data is another sub-factor. For example, *Int1* informed about the data access situation at the company, *"...one is the Company's security policy and the bureaucratic process in the new premise. So, it has been very difficult to get permissions to use the data, or get people involved. So, that is slowing the process down*". Therefore, a company's data access policy has some bearing on data availability, which further influences the ability to operationalize process metrics.

2) Data Format

The format of available data is also an important criterion. This can be determined by the automatic data collection mechanism (using plugins). *Int1* elaborates, "*Depending (on) what kind of plugins you are using, those APIs might produce different kind of results. But if you have different set of plugins, that structure could be different, then you have to do different kind of stuff to get the data in the right format, and those are the biggest issues that I have*". How different tools capture and store data will naturally be different from one another. However, it is important that these varying data formats are transformed into a standard format for smoother metrics operationalization. Even Case B had to postpone operationalization of few process metrics, because relevant data were available in text form (difficult to parse), rather than a specific field in some tool that could be retrieved automatically.

B. F2: Process

An organization's software development process dictates the choice of process metrics, and by extension, their operationalization. The following sub-factors contribute to this factor:

1) Process Alignment

In addition to data, metrics are reliant on the development process that helps produce that data. Overlooking the development process can result in lot of rework, and in worse cases, foregoing of the metrics. For example, Case B wanted to measure project related tasks, but their development process did not involve use of any tool to document these tasks. Their traditional methods of documenting these tasks impeded the development of a handful of process metrics, forcing them to postpone their operationalization. It was only after they adopted a new tool for these tasks (generating the relevant data) that they managed to operationalize the relevant process

TABLE IV. FACTORS FOR EFFECTIVE PROCESS METRICS OPERATIONALIZATION IN OUR STUDY

ID: Factor	Sub-factors	Description of sub-factors for effective process metrics operationalization
F1: Data	Data availability	The required data to compute the software metrics of the program is available
	Data format	Data is available in a machine-readable format to enable automatic data collection
F2: Process	Process Alignment	Data is automatically generated during development process' activities and metrics align with the development process
	Process Flexibility	Flexible enough development process to accommodate new practices to enable operationalization of valuable metrics
F3: Iterative Operationalization	Incremental Operationalization	Applying iterative agile way of working to the metrics program
	Regular Feedback	Short feedback cycles during operationalization, promoting learning and metric evolution
F4: Championing the Cause	Vision	A multi-project long-term vision driving the technical benefits of metrics
	Champion	Dedicated personnel championing the cause of the metrics program
F5: Metrics Trustworthiness	Familiarity	Using familiar (i.e., known) metrics to establish initial trust
	Metric data source	Linking metrics to the corresponding data source containing the raw data
	Trustworthiness value	Metrics may have an evaluation quantifying their trustworthiness (e.g., based on their change history)

metrics. Similarly, *Int2* from Case A cited their development practice of using 'story points' instead of 'hours', which posed a problem in computing the process metric 'Issues' Resolution Throughput'. The metric is computed by using number of hours spent on an individual issue[4], as recorded in Jira[5]. However, doing so puts the focus on individual developers rather than the team, a philosophy Case A does not subscribe to. Therefore, misalignment with the development process resulted in reformulation of the said process metric, requiring more time and effort, which could have been avoided.

One should also take into account the development process exclusive to a project. For instance, Case A does not record errors that leak into the final product, as their exhaustive testing leaves no room for such a possibility in the concerned project. As a result, they could not operationalize the process metric 'Error Leakage' (commonly known as Escaped Defect), which measures the number of errors missed during development. *Int1* justified, "*Error Leakage, at least in Company …Company-wise that makes sense. For the project, it's not very relevant for them*". *Int1* also opined that metrics should reflect project contexts and distinct software development stages of a project. In Case A, the pilot project is split into two modes. One concerns development (*home-office mode*) and the other concerns operationalization (*factory mode*). Referring to the two modes, *Int1* expressed that, "*They should actually have two separate metrics altogether for all of those modes. When they are in factory, all the metrics that we are monitoring about the development go bad. Actually, they should go bad, because they are not at the home-office doing development, they are putting up production systems, and when they are in home-office, when we just look at the metrics related to what happens at the factory, then of course those go down, because nothing is happening at the factory*". *Int1* further added, "*One project could have multiple quality models depending on the lifecycle of that…where they are at. Normal software development project at Company could have, let's say, initial phase for the first 9 months, some kind of quality mode would apply. Then it would go to the active development phase, there will be some, and then the ramp-down and the maintenance, that would be a third big quality model to do*". Therefore, in addition to aligning with an organization's overall development process, metrics should align with processes that represent a particular way of working at a project or team level.

2) Process Flexibility

Process metrics operationalization can influence a company's software development process. Case B had expressed interest in process metrics like 'Timely Feature Specifications Delivery' and 'Timely Feature Delivery' to measure their project management performance. However, Case B was not using any tool that could support automatic data collection for these metrics. They saw value in these process metrics, and after internal discussion, decided to adopt a new tool just to accommodate these metrics. *Int3* confirms the cause of this change, "*Yes, at this time, the tool is implemented in several projects. Basically, we use it to monitor projects, which are used to evaluate the Q-Rapids solution*". Although the change concerns only a single tool and may seem less remarkable, its introduction also translates into changes to how project management tasks are now documented at Case B. Earlier, they documented these tasks using Microsoft Office applications, but with the new tool (OpenProject[6]), they can now automate these tasks to a considerable extent.

In Case A, the interviewed practitioners did not attribute any marked change in their development process due to the operationalized metrics. However, when asked if there is such a possibility, *Int1* agreed, *"Yes, I totally foresee that, because I'm…if there is some metric that we find by chance, some…somewhere, and we see that it's important, we are totally willing to add to the process to accommodate that*".

C. F3: Iterative Operationalization

The two case companies preferred process metrics operationalization in iterations of small increments:

1) Incremental Operationalization

An incremental approach in operationalizing the process metrics has been a common feature for both case companies. Despite showing interest in numerous metrics in the early stages, the case companies selected only a subset of stable and functional metrics, and advanced to next increment of stable and functional process metrics in subsequent iterations. *Int4* pointed out that such cautiousness comes naturally to any company that has been in business for long, wanting to adopt a new solution, and that, "*if it was not cautious it will not be serious*". *Int3* complemented, "*We have to consider the fact that some issues can stay present in the tools (Q-Rapids) and the data provided by the tools, that's why we are cautious in our approach with the values provided by the tool, because they are in development*". Here, *Int3* is referring to the entire Q-Rapids solution, and that its overall stability dictates the company's approach towards process metrics operationalization.

Incremental and iterative operationalization enables careful assimilation of stable and functional solution, as explicated by *Int1*, "*Project at Company mainly focus now on developing something quite small, and working, and stable, so the same mindset is also applied to the Q-Rapids. We want to see something very stable, running without crashes or any problems quite a long time before somebody wants to say, 'Let's add some new…something new'*". This not only helps in successful introduction of process metrics but also increases the probability of actual metrics use, since target users grow confident when they get to use stable and functional metrics. *Int2's* response, "*Normally, we are doing in iterations things*", further supports the need for incremental and iterative process metrics operationalization.

2) Regular feedback

A typical software metrics program should have feedback sessions between the measurement team and the software development team, to interpret the measurement data and take improvement actions, if required. Generally, these feedback sessions are held after metrics become operational. However, feedback sessions during metrics operationalization carry many benefits, learning about metrics being the primary one. In our study, process metrics operationalization was a co-creation process between the researchers and the practitioners, which is why we both represent the measurement team. Practitioners were the end user of the process metrics, and so they represent the development team, too. We had multiple feedback sessions, which informed the steps for the next

[4] https://confluence.atlassian.com/adminjiracloud/issue-types-844500742.html

[5] https://www.atlassian.com/software/jira

[6] https://www.openproject.org/

iteration of operationalization. This way, each iteration produced a better understanding of the process metrics for both the parties, and *Int1* concurs, "*I would say, totally agree that the more we have those feedback sessions and collaborations sessions that would add more value to the usage, because then you have a systematic way of, let's say, arguing for or against certain metrics*". *Int1* further justified multiple feedback sessions, "*I think it would be better that there would be a larger audience introduced systematically into that process, instead of, let's say, trial and error kind of approach, which consumes much more time than the initial, let's say, discussion and systematic feedback sessions*". This response acts as a counterpoint to the notion that constant feedback sessions are likely to consume more time. Therefore, regular feedback becomes part of iterative and incremental operationalization, as companies can apply knowledge from one iteration to the next.

D. F4: Championing the Cause

Successful operationalization of process metrics is futile if the target users do not use them in their daily activities. The following two sub-factors can help to encourage users and increase the likelihood of process metrics use:

1) Vision

For *Int1*, it was the vision of the Q-Rapids project that helped in recruiting other projects to pilot the solution. Elaborating further, *Int1* states, "*I want to sell that kind of vision to the other projects inside the company. When I first talk about the Q-Rapids to any new project inside the company, I do not even mention in the first 10 minutes that it is a technical tool. I just tell them the story about the vision*". *Int1* is referring to the entire Q-Rapids solution, and the process metrics play an integral role in realizing its vision.

2) Champion

Dedicated personnel to champion the cause of metrics is essential in propagating both short-term and long-term benefits to interested parties. *Int1* had adopted multiple roles in the project, and use-case champion was one of them. Juggling multiple responsibilities affected *Int1's* role as a champion, as conveyed here, "*You could have the technical guys who set up the systems, get the stuff, but then they do not have the time to start training all the people, just talking to them that why…why something is easy, is important. So I think this use-case champion could have been already a role that could have been, let's say, introduced way before*". It was only after the load of technical tasks reduced that *Int1* could find the time to address the role of a use-case champion. *Int1* adds, "*My technical load had already lessened at that point, because I had stabilized the systems. So, I had more time to go and talk to people, and how to use the systems properly*".

E. F5: Metrics Trustworthiness

A user should be able to trust a metric in order to use it with confidence. Following sub-factors influence a user's perception about a metric's trustworthiness:

1) Familiarity

One of the easiest and quickest way to win target users' trust is to use metrics that are familiar to them. For example, Case A chose some metrics for Jira and Jenkins tools, because target users were already familiar with those. *Int1* justifies, "*If you have the same metrics (from another tool), as mentioned before, that you already trust those, and you can show them that, 'This another tool shows those same metrics in a different way, but they come from the same place', then it is easier to convince those end users to adopt the new system, because they know the underlying data*". *Int1* believes appealing to the target users' familiarity is critical in establishing initial trust. Later, it becomes relatively easier to add more metrics that the users may be unfamiliar with, but more inclined to trust them. *Int1* confirms, "*When the users are willing to use those (familiar metrics)…that new tool as a daily driver, then we can start adding little by little those new metrics, which have no relevance, let's say, history in the company in some other areas, or in some other tools. So people will gain the initial trust via the Q-Rapids solutions. Because if you introduce a new system, the human psyche works that you look for the familiar patterns*". By leveraging a user's past experience and the natural tendency to look for patterns, initial trust for metrics can be established, which can later be harnessed for trust for new metrics.

2) Metric data source

Another way of convincing users of a metric's trustworthiness is to provide them access to the data source for those metrics. *Int2* clarifies, "*So, the confidence comes also that from this metrics that you can really drill down the metric…where is the value coming from…why it is like that, and things like that*". Link from metrics to the actual data source helps in building target users' trust in the metrics. The added transparency allows users to interpret the whole data flow from the source to metrics, and decide on a metric's trustworthiness independently.

3) Trustworthiness Value

A unique proposition by *Int1* was to develop a feature that calculates a metric's trustworthiness. The value will depend on how often a metric is refactored or modified in a given period. *Int1* describes, *"You could, let's say, all the metric were super quite reliable if the (trustworthiness) value would be over 90, and if people do not touch the…if people do not refactor the metric, if they do not touch it, it means that they trust it. The more you need to configure and trust a metric, more it seems that you do not trust it. So, if you put some metric there, and it has been untouched for a year and a half, probably, people are trusting it, so they do not need to change it*".

To sum up this section, factors *F1* and *F2* were key to operationalizing process metrics. The two case companies have firm plans to use process metrics beyond the current pilot project. *Int1* informed about the organization-wide interest, "*The two medical site (medical projects), and the (Project name), those were the most eager to participate after the pilot, and now I have discussed with three more projects about the actual usage, and it would seem that overall there are eight or nine more projects interested*". Similarly, *Int4's* comment, "*I'm looking for the solution to link it to all the other initiatives…research initiative that we are carrying out in the company*", is suggestive of the long-term potential of the Q-Rapids solution, and the process metrics by extension. In view of this, factors *F3 – F5* seem to be the potential contributors to the long-term use of the operationalized process metrics.

V. Discussion

We discuss our multiple case study findings by comparing the identified factors with the success factors for establishing metrics programs by Hall and Fenton [3], and the success factors for long-term use of metrics programs by Staron and Meding [8]. The former factors (*HF1 – HF11*) are seen as a

consensus for a successful metrics programs, and the latter (*SM1 – SM11*) are factors that facilitate long-term use of metrics programs in ASD. The comparison is presented in Fig. 1 (the tables). We also highlight the stages where the different factors were observed during the process metrics operationalization. We structure the discussion in the form of factors that are rarely or never mentioned in relevant literature but deserve consideration (factors in bold in Table V), factors that we could validate (Table VI and Table VII), and factors that we did not find any evidence for in our multiple case study (factors in bold in Table VII). It is important to note that steps taken early in the Q-Rapids project [9], [10], [12]–[14], but not explicitly a contribution of this multiple case study, helped to validate many success factors from the state of the art.

A. Factors deserving consideration

The most fundamental factor of *Data* (*F1*) has not received the desired type of attention in the list of factors from [3] and [8]. Hall and Fenton [3] limit their discussion about data till automatic data collection, which we emphasize under the *F1* sub-factor of *Data format*. We drew a similar conclusion from the study by Gencel et al. [21], where the emphasis is to prioritize metrics and collect only relevant data, in response to an organization's limited resource. This is similar to *Int2's* comment about choosing only the 'must-have' metrics, and *Int1's* rationale for using a smaller set of metrics for operationalization. The mapping study by Tahir et al. [22] identified five studies that advocate the need for data availability, but only one of those studies [23] concerns software engineering (SE). This suggests that one of the most fundamental requirements of a metrics program is overlooked in the related SE literature. Modern development methods like ASD and related practices of Continuous Integration (CI) involve use of a wide range of tools (like SonarQube, Jira, Jenkins, Gerrit), and these tools are rich sources of data, storing insights about an organization's development process. Therefore, for operationalization of metrics programs in ASD, it is now more important than ever to emphasize on data.

Similarly, the factor *Process* (*F2*) and all its sub-factors fail to find any mention in [3] and [8]. Based on the literature review in [24] and mapping study in [22], this factor appears to be overlooked in the state of the art as well. In our multiple case study, misalignment with the development process dictated data availability, data format, and, ultimately, feasibility of operationalizing certain process metrics. The factor of *Process* appears to influence the factor of *Data* in many ways [25], and so we argue that *Process* is also a fundamental factor deserving due consideration, especially to operationalize process metrics in ASD.

One potential reason for the relative absence of the above two factors, *Data* and *Process,* from the state of the art could be the modern software development methods of ASD. Except [8], none of the relevant literature proposes success factors in the context of ASD. The flexibility in development practices, data collection and storage practices is a result of the use of ASD, and this could be why *Process* and *Data* emerged as critical factors in our study, but not in similar studies from the state of the art. We believe the key is integration of data-driven process metrics in different activities of ASD process, to support transparent and evidence-based decisions. For instance, analyzing the way of working of an agile team based on process metrics during sprint retrospective meetings.

The factors *Iterative Operationalization* (*F3*) and *Metrics Trustworthiness* (*F5*) find support in existing literature, but not at the granularity of the sub-factors that we report here. For example, Hall and Fenton [3] recommend implementing a metrics program over time (*HF10*). Based on our findings, incremental implementation is necessary in ASD, as it needs to be compatible with an organization's agile way of working, involving short cycles of software development. Similarly, with respect to the *F3* sub-factor *Regular feedback*, authors in [1] and [3] discuss the importance of traditional feedback sessions between the measurement and development team, once metrics have been operationalized. We posit that multiple feedback sessions are necessary even during metrics operationalization, as it helped both the case companies build their understanding about the process metrics and their confidence in them. In view of the claim that without feedback sessions a metrics program will gradually perish [1], we argue that the sub-factor *Regular feedback* could play a role in long-term use of metrics programs as well. This could be because metrics programs need to evolve over time, and regular feedback can facilitate this evolution.

The factor *Metrics trustworthiness* (*F5*) appears to be the most important factor, especially from the standpoint of long-term metrics programs use. The sub-factor of *Familiarity* has been raised by Gencel et al. [21], but only from the utilitarian perspective of relying on existing standards for defining, using, and reusing metrics. However, our finding about how familiar metrics can help to engage target users and establish initial trust, which later translates into sustaining that trust, is underrepresented in existing literature. Similarly, a mechanism to make explicit a metric's trustworthiness by computing a *Trustworthiness value* deserves further investigation to test its true effectiveness.

B. Validated factors for establishing a metrics program

GQM was used in the Q-Rapids metrics program for a goal-oriented approach, involving all the stakeholders that provide regular feedback to a dedicated metrics team. This helped to validate most of the factors (*HF1 – HF6*) from [3]. GQM has been cited as a success factor for metrics program in [5] and [26] as well. Similarly, the automatic data collection plugin developed in Q-Rapids [12], [13] satisfies the factor *HF9*, aimed at minimizing efforts for data collection. Gencel et al. [21] also emphasized the importance of data collection, and how an organization's resources can be a limiting factor.

The *F3* sub-factor *Incremental Operationalization* validates the success factor of incremental implementation (*HF10*) from [3]. An organization's simple, stepwise introduction of a metrics program [27], in iterations [5] is necessary to increase its chances of success. Incremental implementation in iteration enabled both the case companies to streamline process metrics operationalization, by deploying only the stable and functional metrics. Literature recognizes such an approach as less risky than a 'big bang' approach [3].

The study by Hall and Fenton [3] is among the few studies that call for an external metric guru and an internal metrics champion (*HF11*). The former helps to raise initial enthusiasm for the program, while the latter helps to ensure sustain it. The *F4* sub-factor of *Champion* validates *HF11*. As observed in Case A, *Int1* championed the metrics program across the organization, by marketing not just the technical benefits of the metrics program but also the overarching vision of the entire solution that the process metrics constitute. In Case A, Q-Rapids is being piloted in two more projects, and 'eight or nine' projects have expressed interested in using the solution.

TABLE V. OUR FINDINGS

ID	Factors
F1	**Data** - ***Data availability*** - ***Data format***
F2	**Process** - ***Process alignment*** - ***Process Flexibility***
F3	Iterative Operationalization - *Incremental operationalization* - *Regular Feedback*
F4	Championing the Cause - *Vision* - *Champion*
F5	**Metrics Trustworthiness** - ***Familiarity*** - ***Metric data source*** - ***Trustworthiness value***

TABLE VI. SUCCESS FACTORS FOR ESTABLISHING METRICS PROGRAMS [3]

ID	Factors	Validated by…
HF1	Goal-oriented approach	GQM, part of Q-Rapids [10][13]
HF2	Usefulness	
HF3	Developer Participation	
HF4	Feedback	
HF5	Dedicated metrics team	
HF6	Practitioner training	
HF7	Transparency	*F5: Metrics Trustworthiness*
HF8	Metrics Integrity	
HF9	Automated data collection	Tool plugin (connector), part of Q-Rapids [12]
HF10	Incremental Implementation	*F3: Iterative Operationalization*
HF11	Gurus and champions	*F4: Championing the Cause*

TABLE VII. SUCCESS FACTORS FOR LONG-TERM USE OF METRICS PROGRAMS [8]

ID	Factor	Validated by…
SM1	**Working according to ISO/IEC 15939 standard**	*ISO 25010 basis for Q-Rapids Quality Model [13][18], but practitioners preferred mostly custom metrics*
SM2	**Standard base measures**	
SM3	Providing information quality indicators	*F5: Metrics Trustworthiness*
SM4	Automated data collection	Tool plugin (connector), part of Q-Rapids [12]
SM5	Minimal measurement collection effort	
SM6	Individual stakeholders per measurement system	GQM and Quality Model, part of Q-Rapids [9] [10] [13] [28]
SM7	Direct benefits to the organization	
SM8	Devoted measurement team	
SM9	Not to use programs to assess individuals	
SM10	**Reusing base measures**	*No supporting evidence found*
SM11	**Using measures specifications and specification of their instantiation**	

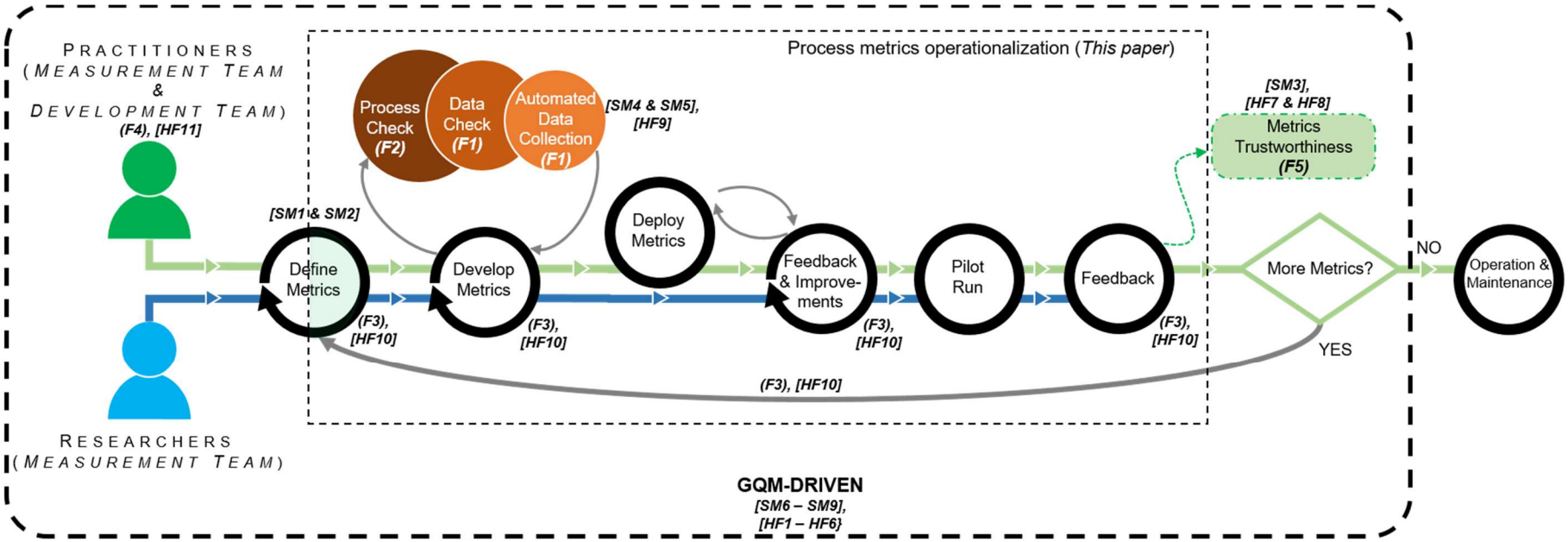

Fig. 1. Factors observed during process metrics operationalization

Note: (F#), [HF#], [SM#] - Factors whose evidence were found during process metrics operationalization

These developments highlight the impact a champion can have on 'selling' a metrics program in an organization.

Although Hall and Fenton [3] do not explicitly state the factor of metrics trustworthiness, they allude to it by requiring a metrics program to be transparent (*HF7*) and of integrity (*HF8*). *HF7* is validated by the *F5* sub-factor *Metrics data source*, which enables a user to find out the actual data source used to compute a metric. The factor *HF8* is limited to the collected data and not the computed metric, but the former does act as a prerequisite to build trust for the latter, which is also central to our identified factor of *Metrics Trustworthiness* (*F5*).

C. Validated factors for long-term metrics program use

Among success factors for long-term metrics program use endorsed by Staron and Meding [8], automatic data collection plugins developed for the project [12], [13] addresses the factors about using automatic data collection (*SM4*) and using minimal efforts for the same (*SM5*). Similarly, factors from *SM6* to *SM9* are validated by using GQM and a Quality Model as the main drivers for the metrics program in the project [9], [10], [13], [28]. The Quality Model accommodates measurement requirement for developers, managers, and the top management [13], satisfying factors *SM6* and *SM9*. Similarly, the goal-oriented approach of GQM helps provide direct benefits to the organization (*SM7*), which requires use of a dedicated measurement team (*SM8*).

Staron and Meding [8] state that a metrics program should provide information that is up to date and reliable. These qualities are inherent in our factor *F5* (*Metrics trustworthiness*) and its sub-factors (*Familiarity* and *Metrics data source*). Metrics trustworthiness is one of the lesser-known factors in both successful operationalization and long-term use of metrics programs. This factor does not feature in one of the recent systematic literature reviews on software measurement program [24] either. Similarly, the study on validation of software metrics by Meneely et al. [29] does not mention metrics trustworthiness as a factor. One could argue that metrics validation is a precursor to metrics trustworthiness. However, the evident knowledge gap justifies further empirical investigation into this topic, especially if metrics trustworthiness guarantees long-term use of metrics programs in ASD, as claimed in [8] and as found in our study.

D. Success factors that could not be validated

We found evidence contrary to the factors of *SM1* and *SM2,* which require metrics programs to be based on recognized standards, specifically ISO/IEC 15939, and to use standard base measures (*metrics*). The Q-Rapids Quality Model [13] is based on a comparable ISO 25010 standard. However, practitioners prefer mostly custom metrics over standard metrics, as reported in [10]. This suggests that existing standards may provide an initial base to conceive a metrics program. However, in practice, practitioners prefer custom metrics, as it grants them more freedom and flexibility to measure process aspects in which they are interested.

We could not find evidence to support factors *SM10* and *SM11*. These factors require metrics program's reusability beyond the context of one project, something that can be analyzed only after companies start using the program in other projects. At present, the two case companies are piloting the metrics program in just one project, but have expressed plans to include more projects in due course. Case A has been particularly promising in this regard, as they have started to pilot the program in other projects, with even more projects expressing interest. Another case study in future could be an appropriate research instrument to investigate the validity of the two factors of *SM10* and *SM11*.

Overall, factors (and the sub-factors) of *Data* and *Process* are the two basic constituents necessary for operationalization of metrics programs in ASD. *Iterative Operationalization* of metrics programs is another factor that is influenced by the modern development methods of ASD. It also allows practitioners control over using only those metrics that are relevant and valuable, which could contribute to its long-term use. A dedicated champion for *Championing the Cause* of a metrics program can help to reinforce its long-term vision. Most importantly, *Metrics Trustworthiness* deserves a special mention, as it exhibits the strongest potential to contribute to a metrics program's long-term use. However, more studies are needed to test these claims.

VI. Threats to Validity

In adherence to the case study guidelines in [15], we discuss below the threats to the validity of our research.

Misinterpretation of collaboration results and interview data by the researchers pose a threat to our study's construct validity. We shared the results with the case companies and validated them in the interviews to address this threat. Similarly, interview data analysis was shared with the interviewees and other researchers to mitigate this threat.

It is possible that the perceived advantages of the overall Q-Rapids solution influenced the interviewees' perception about the benefits of process metrics, thereby posing a threat to our research's internal validity. We made the distinction between other objects of study in the project and the process metrics operationalization explicit in our interview questions, and we elicited responses that focused only on the operationalized process metrics. Wherever we have made a claim, we have included the supporting response from the corresponding interviewee, which helps to address this threat.

Threat to external validity exists because we studied only two software-intensive companies using ASD. One of the case companies is not following textbook ASD process, which can also be a threat to the study's external validity. However, the reported factors necessary for effective process metrics operationalization have also been recommended in the state of the art, which suggests that our findings are not exclusive to just the two case companies. Findings like metric trustworthiness and influence of metrics on development processes have limited coverage in the state of the art, but we found them to be important at the two case companies. Overall, our results are applicable to organizations that are similar in context to the two case companies.

Multiple researchers helped to validate the findings from the interview data analysis, but only one researcher was involved in data collection. This may affect the reliability of our research.

VII. Conclusion and Future Work

The complexity of software and the process to develop it has been increasing at a regular rate, which carries implications for the quality of both product and process. Software metrics have emerged as one of the effective ways to quantify and evaluate this quality [7]. With over 80% of metrics programs failing within the first 18 months [6], it is

imperative to use approaches that would increase the likelihood of success. Existing literature proposes some success factors to establish a metrics program, but empirical research for factors that ensures long-term use of such programs is scarce, and investigation about these factors in the context of ASD is scarcer. In the context of the Q-Rapids project, we have tried to address this knowledge gap.

We conducted a multiple case study at two software-intensive Q-Rapids industrial partners, focusing on five-month collaborations for process metrics operationalization. We reported that in view of modern development methods like ASD, availability of data and alignment with development process are fundamental to effective operationalization of process metrics. Incremental operationalization of metrics programs in iterations is necessary in ASD, as it conforms to the agile way of working of developing software in short cycles. Regular feedback contributes to successful operationalization and long-term evolution of metrics programs, potentially contributing to the program's long-term sustainability. A dedicated champion can help to 'sell' a metrics program in an organization, generating interests conducive to adoption and use of metrics programs. However, metrics trustworthiness should be the focus to maximize the chances of a metrics program's long-term use in ASD.

Our future work will involve monitoring behavior of the operationalized process metrics, and determining their impact on the case companies' development process. However, our immediate focus is to investigate different ways to ensure metrics trustworthiness, as this sole factor seems to exhibit strong potential in dictating metrics programs' long-term use.

ACKNOWLEDGEMENTS

This work is a result of the Q-Rapids Project, funded by the European Union's Horizon 2020 research and innovation program, under grant agreement No. 732253

REFERENCES

[1] R. van Solingen and E. Berghout, "Integrating goal-oriented measurement in industrial software engineering: industrial experiences with and additions to the Goal/Question/Metric method (GQM)," *Proceedings Seventh International Software Metrics Symposium*, pp. 246–258, 2001.

[2] M. G. Mendonça and V. R. Basili, "Validation of an approach for improving existing measurement frameworks," *IEEE Transactions on Software Engineering*, vol. 26, no. 6, pp. 484–499, 2000.

[3] T. Hall and N. Fenton, "Implementing effective software metrics programs," *IEEE Software*, vol. 14, no. 2, pp. 55–64, 1997.

[4] J. Iversen and L. Mathiassen, "Cultivation and engineering of a software metrics program," *Information Systems Journal*, pp. 3–19, 2003.

[5] F. Niessink and H. Van Vliet, "Measurements should generate value, rather than data," *Proceedings of the Sixth International Software Metrics Symposium*, pp. 4–6, 1999.

[6] L. G. Wallace and S. D. Sheetz, "The adoption of software measures: A technology acceptance model (TAM) perspective," *Information and Management*, vol. 51, no. 2, pp. 249–259, Mar. 2014.

[7] E. Kupiainen, M. V. Mäntylä, and J. Itkonen, "Using metrics in Agile and Lean software development - A systematic literature review of industrial studies," *Information and Software Technology*, vol. 62, no. 1, pp. 143–163, 2015.

[8] M. Staron and W. Meding, "Factors Determining Long-term Success of a Measurement Program: An Industrial Case Study," *e-Informatica Software Engineering Journal*, vol. 1, no. 1, pp. 7–23, 2012.

[9] X. Franch *et al.*, "Data-driven requirements engineering in agile projects: the Q-rapids approach," *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017*, pp. 411–414, 2017.

[10] P. Ram, P. Rodriguez, and M. Oivo, "Software Process Measurement in Agile Software Development : A Multiple-Case Study," in *International Conference on Product-Focused Software Process Improvement*, 2018, pp. 272–287.

[11] V. R. Basili, "Software modeling and measurement: the Goal/Question/Metric paradigm," *Quality*. p. 24, 1992.

[12] L. López *et al.*, "Q-rapids tool prototype: Supporting decision-makers in managing quality in rapid software development," in *Lecture Notes in Business Information Processing*, 2018, vol. 317, pp. 200–208.

[13] S. Martínez-Fernández, A. Jedlitschka, L. Guzmán, and A.-M. Vollmer, "A Quality Model for Actionable Analytics in Rapid Software Development," in *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 2018, pp. 370–377.

[14] R. Kozik, M. Choras, D. Puchalski, and R. Renk, "Data analysis tool supporting software development process," *2017 IEEE 14th International Scientific Conference on Informatics, INFORMATICS 2017 - Proceedings*, vol. 2018–Janua, pp. 179–184, 2018.

[15] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, Apr. 2009.

[16] D. S. Cruzes and T. Dyba, "Recommended Steps for Thematic Synthesis in Software Engineering," *2011 International Symposium on Empirical Software Engineering and Measurement*, no. 7491, pp. 275–284, 2011.

[17] V. Basili, J. Heidrich, M. Lindvall, J. Münch, M. Regardie, and A. Trendowicz, "GQM+Strategies - Aligning business strategies with software measurement," *Proceedings - 1st International Symposium on Empirical Software Engineering and Measurement, ESEM 2007*, pp. 488–490, 2007.

[18] S. Wagner *et al.*, "The Quamoco Product Quality Modelling and Assessment Approach," *2012 34Th International Conference on Software Engineering (ICSE)*, pp. 1133–1142, 2012.

[19] ISO (International Organization for Standardization), "ISO/IEC/IEEE 15939:2017(en) Systems and software engineering — Measurement process," *International Organization for Standardization*, 2017. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso-iec-ieee:15939:ed-1:v1:en.

[20] C. B. Seaman, "Qualitative methods in empirical studies of software engineering," *IEEE Transactions on Software Engineering*, vol. 25, no. 4, pp. 557–572, 1999.

[21] C. Gencel, K. Petersen, A. A. Mughal, and M. I. Iqbal, "A decision support framework for metrics selection in goal-based measurement programs: GQM-DSFMS," *Journal of Systems and Software*, vol. 86, no. 12, pp. 3091–3108, 2013.

[22] T. Tahir, G. Rasool, and M. Noman, "A systematic mapping study on software measurement programs in SMEs," *E-Informatica Software Engineering Journal*, vol. 12, no. 1, pp. 133–165, 2018.

[23] N. Ohsugi *et al.*, "Using Trac for Empirical Data Collection and Analysis in Developing Small and Medium-Sized Enterprise Systems," in *International Symposium on Empirical Software Engineering and Measurement*, 2015, vol. 2015–Novem, pp. 206–214.

[24] T. Tahir, G. Rasool, and C. Gencel, "A systematic literature review on software measurement programs," *Information and Software Technology*, vol. 73, pp. 101–121, 2016.

[25] A. Janes, "Non-distracting, Continuous Collection of Software Development Process Data," in *Synergies Between Knowledge Engineering and Software Engineering*, Springer, Cham, 2018, pp. 275–294.

[26] A. Gopal, M. S. Krishnan, T. Mukhopadhyay, and D. R. Goldenson, "Measurement programs in software development : Determinants of success," *IEEE Transactions on Software Engineering*, vol. 28, no. 9, pp. 863–876, 2002.

[27] J. Iversen and L. Mathiassen, "Lessons from implementing a software metrics program," in *Proceedings of the 33rd Hawaii International Conference on System Sciences*, 2000, vol. vol.1, pp. 1–11.

[28] M. López-Benítez, T. D. Drysdale, S. Hadfield, and M. I. Maricar, "Prototype for multidisciplinary research in the context of the Internet of Things," *Journal of Network and Computer Applications*, vol. 78, pp. 146–161, Jan. 2017.

[29] A. Meneely, B. Smith, and L. Williams, "Validating Software Metrics: A Spectrum of Philosophies," *ACM Transactions on Software Engineering and Methodology*, vol. 21, no. 4, pp. 1–28, 2012.