Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Measuring and Improving Agile Processes in a Small-size Software Development Company

MICHAŁ CHORAŚ[1,2], TOMASZ SPRINGER[1], RAFAŁ KOZIK[1,2], LIDIA LÓPEZ[3], SILVERIO MARTÍNEZ-FERNÁNDEZ[3,4], PRABHAT RAM[5], PILAR RODRÍGUEZ[5,6], XAVIER FRANCH[3]

[1]ITTI Sp. z o.o., Poznań, Poland
[2]UTP University of Science and Technology in Bydgoszcz, Poland
[3]Universitat Politecnica de Catalunya, Barcelona, Spain
[4]Fraunhofer IESE, Kaiserslautern, Germany
[5]University of Oulu, Finland
[6]Universidad Politécnica de Madrid, Spain

Corresponding author: Michał Choraś (e-mail: mchoras@itti.com.pl).

ABSTRACT **Context:** Agile software development has become commonplace in software development companies due to the numerous benefits it provides. However, conducting Agile projects is demanding in Small and Medium Enterprises (SMEs), because projects start and end quickly, but still have to fulfil customers' quality requirements. **Objective:** This paper aims at reporting a practical experience on the use of metrics related to the software development process as a means supporting SMEs in the development of software following an Agile methodology. **Method:** We followed Action-Research principles in a Polish small-size software development company. We developed and executed a study protocol suited to the needs of the company, using a pilot case. **Results:** A catalogue of Agile development process metrics practically validated in the context of a small-size software development company, adopted by the company in their Agile projects. **Conclusions:** Practitioners may adopt these metrics in their Agile projects, especially if working in an SME, and customise them to their own needs and tools. Academics may use the findings as a baseline for new research work, including new empirical studies.

INDEX TERMS Agile Software Development, Process Metrics, Software Engineering, Software Quality, Rapid Software Development, SMEs.

I. INTRODUCTION

Agile development methodologies are widely adopted nowadays by software development companies of every kind [38]. Industry surveys show that virtually all organisations use Agile methods to some extent, and over half of them have Agile as their usual approach to software development[1]. Practitioners report many benefits, ranging from reduced time-to-market, to increased customer satisfaction and reduced development costs, among others[2]. However, managing Agile projects may be challenging [10], especially in the case of Small and Medium Enterprises (SMEs). The challenge for the Product Owner and Scrum Master is at least twofold: to assure software product quality and to facilitate the effectiveness of the team and the process.

Currently, in many software development companies, teams are using various specific tools (such as Jira, GitLab and SonarQube) in order to support the development process and the quality of the code and products. This is usually done in a regular retrospective meeting that involves all the team. As far as the code quality is concerned, those tools provide sufficient information for the Scrum Team. However, there is still a gap and the need for more solutions reflecting team effectiveness and process quality. It can be stated that, at present, process improvement activities are mainly based on developers' perceptions and little support is given to make process wise data-driven decisions.

The major contribution of this paper comes in the form of a set of metrics that measure the Agile software development

[1]13th Annual State of Agile Report, 2019. https://www.stateofagile.com/#ufh-i-521251909-13th-annual-state-of-agile-report/473508

[2]Hewlett-Packard Enterprise. Agile is the new normal, 2015. https://www.softwaretestinggenius.com/docs/4aa5-7619.pdf

process (which we call *process metrics* hereafter) in an SME-type of company, and the discussion on how those metrics helped the Scrum Team in the development of a commercial product. The metrics were built as part of an Action-Research collaboration involving a team of researchers and a Polish small-size software development company, ITTI Sp. z o.o, working together during the development of the ITTI's CONTRA commercial product, in the context of the Q-Rapids EU project[3].

This paper is structured as follows: Section II provides the background in process metrics and the Q-Rapids project. Section III surveys the state of the art in process metrics used in software development. Section IV presents the research method. Section V defines a set of process metrics. Section VI includes the discussion on the results. Section VII enumerates the threats to validity of the study. Finally, Section VIII concludes the paper.

II. BACKGROUND

A. PROCESS METRICS FOR SOFTWARE DEVELOPMENT

The scientific literature shows that measurement is integral to understanding, predicting and assessing software development projects [12], [43]. Software development involves many processes, and measurement enables us to characterize, control, predict, and improve those processes [28]. Being a human-centered activity, software processes are prone to problems [14], which lends further credence to why they should be continuously assessed and improved, to meet the expectations of the customers and the stakeholders of an organization [14]. Software process measurement can help in achieving the desired level of performance, capability, and quality [13], [30]. Moreover, measuring software processes also allows learning about the quality of the software product [33], [42].

Owing to the relevance of measurement in software development, software metrics have been studied for decades [21]. However, the increasing popularity of Agile software development (ASD) [38] makes understanding of software metrics in Agile context more relevant. Research recognizes the need for Agile organizations to use metrics, but empirical research on metrics in industrial ASD remains scarce [22]. Particularly, the rationale behind the metrics mentioned in the literature (e.g., burn-down charts, test-pass rates, and suitable pace) and how they are actually used in practice have received little attention [22]. In addition, although the aim of measuring in ASD is similar to that in traditional approaches (i.e. to plan and track Agile sprints or cycles, to monitor product quality, and to identify and fix process-related problems), the measurement programs are quite different in practice [22]. Agile's focus on lightweight practices, continuous delivery of working software, flexible development phases, and minimal documentation make it necessary for measurement programs to be well aligned with the Agile mindset and the principle of simplicity [22]. In the particular case of process metrics, software processes are complex and intangible, making software process measurement challenging in practice [19], [42]. Moreover, due to the time, budget and resource constraints, software measurement is rife with challenges, particularly in SMEs [9], [23], [37].

[3]https://www.q-rapids.eu/

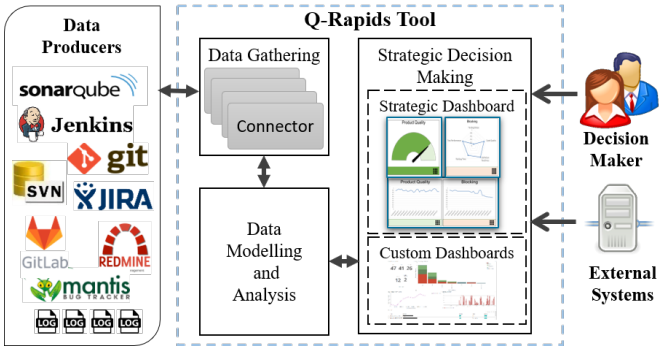

FIGURE 1: Q-Rapids Tool conceptual architecture

With considerations to resources [9], [23], [37], metric selection methods [3], infrastructure facilities, team size [8], and a well-planned software measurement program [23], [37], process metrics can assist SMEs in measuring and improving their process performance.

B. THE Q-RAPIDS PROJECT

Q-Rapids was a collaborative industry-academy project (funded by the European Commission under the H2020 Framework), involving three research partners and four companies. It proposed innovative methods and tools to support the software development industry in improving their quality levels (software and process) when using Agile and Rapid software development [6]. All the partners worked together under a co-creation strategy. Besides, every company adapted the results as they were produced, to the specific needs of the company.

The Q-Rapids approach [16] is based on gathering and analysing data from several sources (software repositories, project management tools, system usage and quality of service) [20], [26]. Data is aggregated into quality indicators that are rendered to the different stakeholders by means of the Q-Rapids tool [25].

The Q-Rapids tool, as a result of the project, provides continuous assessment of the quality-related strategic indicators to decision makers. Figure 1 shows an excerpt of the conceptual architecture of the tool. The main modules are *Data Gathering*, *Data Modelling and Analysis*, and *Strategic Decision Making*.

The *Data Gathering* module is composed of different Apache Kafka connectors to enable gathering data from heterogeneous external data sources, such as static code analysis (e.g., SonarQube), continuous integration tools (e.g., Jenkins), code repositories (e.g., SVN, Git, GitLab), issue tracking tools (e.g., Redmine, GitLab, JIRA, Mantis), and usage logs.

The *Data Modelling and Analysis* module processes the data to assess software quality (product and process). Concretely, metrics are calculated from the gathered data. These metrics are aggregated into quality factors related to development and usage. Finally, the quality factors are aggregated into high-level indicators named strategic indicators, which can be aligned to the strategic goals of the organisation [27].

The assessment data processed by the *Data Modelling and Analysis* module is visualised by the dashboards included in the *Strategic Decision Making* module. The data is visualised as aggregated data to the end-user through the web-based GUI, also named *Strategic Dashboard*. The strategic dashboard also includes links to customised dashboards that can be developed to visualise dedicated charts including the data ingested from the data producers' tools. This dashboard allows the user to display those data calculated for the current stage of the project, as well as the evolution of the metrics, factors and strategic indicators over the time. Another configurable property of the data visualisation is the possibility to adjust grid of the time-based charts to the current needs and present evolution of data with granularity from days up to months. The dashboard also allows navigating through the different elements, which provides traceability and enforces the understanding of the assessment.

Other software analytics tools similar to Q-Rapids have recently emerged in the software engineering landscape. Some of them are domain-dependent, e.g. the European Cooperation for Space Standardization (ECSS) metric framework to improve transparency of software development in customer-supplier relationships of space missions [35]. Also, some commercial tools are available in the market with similar characteristics as the Q-Rapids tool. For instance, Squore[4] provides a similar dashboard to Q-Rapids' and includes several software metrics and indicators measuring software quality, although they are not comparable with the set of metrics analysed in this paper.

III. RELATED WORK

There is a long history of research on metrics programs (MPs) [24], and plenty of literature recommending success factors for their implementation [17], [28], [34], [45]. However, literature on SMEs using MPs in the context of Agile is rather scarce. Moreover, the literature on measuring software processes and their role in improving SME processes is even scarcer. Measuring software processes with the use of process metrics enables objective and quantitative evaluation of software processes, which can lead to continuous improvement and learning [34], [43]. However, measuring process metrics is a challenge [44]. Software processes are inherently complex and intangible, which makes their measurement more difficult than their product counterparts [19], [42]. Ideally, measurement activities should consume little effort and time, while being adequate enough to meet an organization's measurement demands. Software organizations need to weigh in cost-efficiency while prioritizing measurement objectives and targets. SMEs have the added constraints of limited budget, ambitious deadlines, and short-term strategy [46]. Due to these reasons, measuring software processes, especially in an SME, becomes a bigger challenge.

Kupiainen et al. [22] conducted a systematic review of the use and impact of software metrics in ASD in industry. The authors reported that software metrics are mainly used for sprint planning, tracking progress, improving software quality, fixing software process, and motivating people. The authors reported that metrics like velocity, effort estimation, customer satisfaction, defect count, technical debt and build are used prominently in ASD. In their systematic review, Tahir et al. [47] observed that metrics for defects, effort, size, duration, productivity, employee commitment, and customer satisfaction are commonly reported in the state of the art. These findings complement another review by Gómez et al. [15], where complexity and size were found to be the most measured attributes in MPs. Other usage of metrics in ASD discussed in the literature are for planning and tracking software development [22], understanding development performance and product quality [49], measuring process quality [51], estimating effort [48], and reporting progress and quality to stakeholders not involved in the actual development [4]. Taken together, metrics targeting sprint planning, fixing software process, effort estimation, development performance, and software defects can be used to measure an organization's process performance. However, this objective is not expressly stated in any of the reviews mentioned above. On the contrary, [22] remarked that more studies are needed to explore the rationale behind utilizing the metrics the authors found in their review.

Most of the studies present initial emerging results of MP implementation in organizations, which have not been evaluated within a larger industrial context. One of the exceptions is the study by Dubinsky et al. [11] reporting on the experience of using an MP at an extreme programming (XP) development team of the Israeli Air Force. The authors found that using metrics to measure the amount, quality, pace, and work status could lead to more accurate and professional decision-making. A similar study by Diaz-Ley et al. [9] proposed a measurement framework, customized for SMEs. One key benefit the authors reported was better measurement goals that align with the company's maturity. Specific to process metrics, most studies focus on using process metrics mainly to predict software faults/defects [29], [36], [41]. However, the role of process metrics in improving an organization's overall process performance, especially in the context of SME and ASD, is missing from these studies.

There have been studies evaluating MPs in SMEs, but the scope has been limited to a particular region, which makes it difficult to generalize their findings. For example, with the goal of evaluating MPs in the Pakistani software industry, Tahir et al. [47] conducted a systematic mapping study combined with a survey among 200 practitioners to highlight the state of measurement practices. Forty-two per-

[4]https://www.vector.com/int/en/products/products-a-z/software/squore/

cent of the organizations that responded to the survey were SMEs. Overall, SMEs fared poorer than their larger counterparts. For instance, SMEs have the lowest share among organizations that have any defined measurement process, measurement standards, and usage of measurement models and tools. Furthermore, 65% of SMEs tend to use MPs primarily at project level, and only 13% of SMEs implement it across the organization. One of the positive findings, with respect to measuring software processes, was that 70% of the SMEs reported to either focus on measuring process or a combination of process and the other two entities. However, the corresponding primary studies were unclear on the context in which the process measurement was undertaken, and the focus on process metrics for process improvement was missing. For example, the study by Díaz-Ley et al. [9] reported the experiences of a Spanish SME in implementing an MP and reported that the practitioners could now objectively evaluate the trade-off between on-time releases and software product reliability. Tosun et al. [50] collaborated with a Turkish healthcare SME to institutionalize process improvement practices, and reported improvements in the organization's time allocation for requirements, coding, and testing steps. Furthermore, the authors found that defect rates as well as testing effort estimation decreased. One of the more interesting approaches for improving an SME's process was documented in an experimental study by Caballero et al. [5]. The authors introduced Scrum to enhance process productivity without impacting product quality at a very small enterprise. The authors claim that Scrum can prove to be a good alternative for process improvement in an organization with very limited resources, which has been a long-time concern in implementing MPs in SMEs. It is evident from these studies that the evaluations of MPs in SMEs are concerned mainly with the overall software process improvement, where the role of process metrics towards this objective is either implied or absent altogether.

As per the state of the art, there is extensive reliance on measurement experts and experience [47], and organizations tend to prefer employee perception to objective measurement processes for process improvements [32]. In contrast, our study provides empirical evidence of using process metrics for improving process performance, and even facilitating decision-making. The empirical validation is especially a distinguishing aspect of our study, as it has been identified as a research gap in [22]. Furthermore, it should be noted that the Q-Rapids solution, embodying the MP, integrates basic features like automatic data collection, support for diverse data sources, expert-based metrics elicitation, and visualisation; something that is absent from the MPs reported in the literature.

IV. RESEARCH METHODOLOGY

A. CONTEXT

ITTI is a software development and consulting company based in Poznan, Poland. ITTI currently has about 70-90 employees. ITTI delivers software products in a number of application domains (e.g. administration, utilities, e-Health, and crisis management). In this paper, we report an Action-Research study focused on one particular ITTI software product, named and branded CONTRA. CONTRA is an enterprise class integrated software system for Warehouse (WMS) and Manufacturing (MES) management and deployed in the form of web application.

ITTI applies Scrum in their software development projects, including CONTRA. Typically, from 7 up to 10 developers work daily on specific deployments or on the new features to improve the product. The Scrum team holds weekly sprint meetings on the last day of the sprint. Each Scrum Team meeting consists of the following parts: review, retrospective and planning for the next sprint.

B. ACTION-RESEARCH APPROACH

To conduct this research, we applied an Action-Research cycle: diagnosis, action planning and design, action taking, evaluation, and specifying learning [31], [39]. ITTI participants in the Q-Rapids project played a double role as researchers and project champions in the company.

The action started in September 2018 with the *diagnosis* of the industry needs in the form of process improvements that ITTI wanted to address. This originated our research goal and research questions (documented in Section IV-C).

As participant of the H2020 Q-Rapids project, in order to tackle these improvements, ITTI decided to customise the Q-Rapids approach and tool with extended process metrics. In October 2018, both the team of researchers and the CONTRA Scrum team decided the company repositories to be used and how to make actionable this data with the outcomes of the Q-Rapids project (*action planning and design*).

Next, the same joint researchers-practitioners team jointly elicited 25 candidate process metrics fed with the selected company data. The resulting process metrics were used by the CONTRA Scrum Team during their meetings to reflect on the process performance and quality of the product as well as to estimate tasks (*action taking*). This action took place from November 2018 to May 2019.

The process metrics were evaluated in a retrospective session with the Scrum team of CONTRA in June 2019 (*evaluating*).

Finally, the team learned the subset of process metrics which are more effective for the diagnosed problems (*specifying learning*). Since July 2019 to the present, such subset of process metrics has been used in other projects in ITTI.

C. GOAL AND RESEARCH QUESTIONS

In the regular meetings reported in Section IV-A, ITTI Scrum Teams diagnosed the need to: (a) monitor the process performance of the team, (b) keep a stable product quality level while adding new features, and (c) improve the estimation of tasks during sprints.

Bearing in mind these industrial needs, ITTI considered the customisation of process metrics, a concept already implemented in the Q-Rapids project, and applied them to

CONTRA, serving as the pilot case. Following the Goal-Question-Metric approach (GQM) [2], we can define the resulting research goal of this study as: **Analyze** process metrics **with the purpose to** evaluate them **with respect to** monitoring and estimating the Agile process performance **from the view point of** the Scrum team **in the context of** an SME.

We break this generic research goal into three research questions, aligned with the needs anticipated above:

- RQ1. *Do process metrics help the Scrum team of an SME to monitor their own process performance?*
- RQ2. *Do process metrics support the Scrum team of an SME in keeping a stable product quality level while adding new features?*
- RQ3. *Do process metrics help the Scrum team of an SME to improve the estimation of tasks during sprints?*

D. INSTRUMENTATION

In the *action planning and design* phase of the Action-Research cycle, we decided to use GitLab as the data source for the pilot. GitLab is an open source, web based tool that provides support to the full software development life cycle, with focus on repository management and issue tracking, among other capabilities. GitLab is used extensively in all ITTI projects and in particular, the CONTRA Scrum Team affirmed that it is the tool that may best reflect the process followed by the team during development.

ITTI gathered data from GitLab from about 12 months history of the CONTRA development, so that the process metrics could be assessed during a long life span. During this time, a total of 31 unique assignees opened up to 2.975 issues from which they closed up to 2.651, and there happened 40.947 events describing the changes of status of tasks or issues.

Table 1 includes the collected data for each issue from GitLab. This data is stored in a dedicated index in the ElasticSearch engine. From this data, during the studied period, the Q-Rapids tool provided a total of 1.830 metrics, 1.098 factors, and 732 strategic indicators assessment data points.

In order to evaluate the usefulness of the process metrics defined in CONTRA, the Scrum Team proceeded as follows:

- They implemented the connectors to GitLab that allowed to effectively gather the data to start the measurement process (see Section II-B).
- Given the existence of two strategic indicators provided by the Q-Rapids project, related to the two first research questions (*Process performance* and *Product Quality*), they used the Q-Rapids dashboard for the process metrics related to these questions.
- For the third research question, they preferred to deploy Kibana dashboards in order to assess task estimation. In order to provide an integrated solution, these Kibana dashboards were integrated into the Q-Rapids dashboard.

Figure 2 illustrates the Q-Rapids dashboard by showing some example of historical data views. In these charts we can see that *Density of tickets pending testing* and *Density of bug* have been stable in low and high quality values respectively in the period of fifteen days. On the other hand, in the same period, there is a clear improvement in the *Average resolved issues' throughput (last 7 days)* and some improvement in *Density of estimated development tickets (last 7 days)*.

Figure 3 shows an example of the use of Kibana dashboards. In this case, the presented view aggregates several metrics related to the average elapsed time for tasks (according to their state).

V. RESULTS

A. PROCESS METRICS DEFINITION

In the *Action taking* phase of the Action-Research cycle, the ITTI CONTRA Scrum Team and the research team discussed and analysed what type of GitLab-based process metrics they consider candidates for assessing the Agile development processes at CONTRA. These metrics were implemented in the CONTRA case in order to understand their significance.

The CONTRA Scrum Team provided relevant observations that drove the design of the candidate set of metrics:

- The three main concepts that they use in their daily practices to monitor progresses are: Task, Issue/Bug and Effort.
- The state transition among development tasks (opened $\rightarrow$ completed $\rightarrow$ closed) is particularly important in analysing progress.
- The effort is particularly interesting in relation to its estimation, because resource planning in the team (e.g., developer allocation) greatly depends on its accuracy.
- The concepts above may be analysed mainly from two perspectives: numerical (e.g., number, accumulated sum or average) and time dimension.
- The metrics should be measurable using GitLab data.

Considering these principles, the team consolidated a proposal of 25 candidate metrics. Although many other metrics appeared interesting, the Action-Research team preferred to keep the proposal manageable in this first iteration, thus focusing on those metrics which the developers agreed upon being the most determinant. The metrics can be divided into several categories as shown in Table 2:

- *General metrics*. Following the Scrum Team observations, we propose an indication of the total number and the average number of development tasks (metrics #1 and #4), number of tasks based on their status, e.g. completed, closed, etc. (#2 and #3), and average time of tasks lifetime (#5 and #6). Metrics #2 and #3 can be extended and the number of tasks marked as "in progress", "testing", "ready", etc., can be also analysed, but we discarded this in order to keep the approach simple in this first iteration. Each of the general metrics can be calculated in different dimensions, i.e. per developer, per specific project, per area (frontend/backend), per sprint or release and narrowing the timespan to the specific range.

TABLE 1: Issue data gathered from GitLab

Attribute	Description	Type
assignee	assigned developer	text
author	member creating the issue	text
dstart	date when the issue appeared in the backlog for the first time	date
dstop	date when the issue was indicated as completed	date
elapse	time elapsed since the opening (in days)	double
estim	time estimation (in days)	double
icreated-at	issue creation time	date
iid	issue identifier	integer
isactive	indicates whenever the issue is still on the sprint board	text
labelid	identifier of the the sprint board's column (e.g. backlog, in progress, testing, etc.)	text
repetition	the number of times the issue returned to the backlog column (e.g. because of the correction)	integer
spent	time spent to close the issue (in days)	double

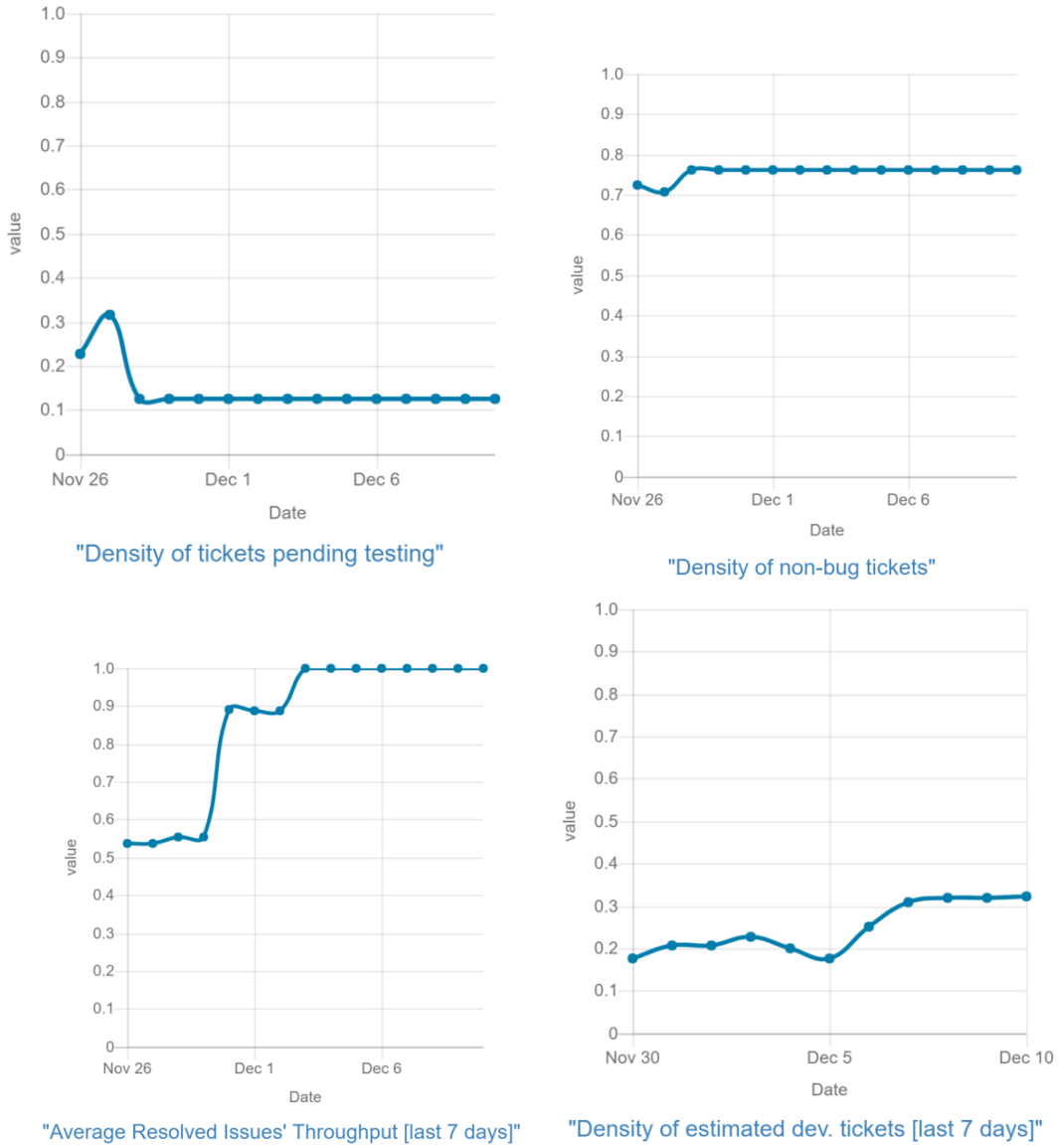

FIGURE 2: Visualization of Process Metrics using the Q-Rapids Dashboard

TABLE 2: Set of candidate process metrics for CONTRA

No	Category	Metric	Value
1	General	Number of development tasks	[number]
2	General	Number of completed (closed) development tasks	[number]
3	General	Number of incomplete (open) development tasks	[%] [number]
4	General	Average number of development tasks	[number]
5	General	Average time of development task in the project board – from the moment it was added, to the moment it was closed	[number/time]
6	General	Average time needed to resolve an issue	[number/time]
7	Task estimation	Effort estimation accuracy for development tasks	[%]
8	Task estimation	Average difference between estimated effort ("estimated" attribute) and real effort ("spend" attribute)	[number/time]
9	Task estimation	Number of development tasks without estimation of effort ("estimated")	[%] [number]
10	Task estimation	Number of development tasks without real effort ("spend")	[%] [number]
11	Task estimation	Total sum of estimated effort values ("estimated")	[number]
12	Task estimation	Sum of effort actually spent ("spend")	[number]
13	Task implementation	Average task implementation time based on project board	[number/time]
14	Task implementation	Average time-to-implementation of task based on project board	[number/time]
15	Task implementation	Number of tasks with unassigned "Milestone" (sprint)	[%] [number]
16	Task implementation	Number of tasks not yet assigned to any developer	[%] [number]
17	Task implementation	Number of ongoing development tasks not belonging to the current sprint	[%] [number]
18	Bug fixing	Number of development tasks with reported bugs	[%] [number]
19	Bug fixing	Average time of task correction based on project board	[number/time]
20	Bug fixing	Average time-to-correct of task based on project board	[number/time]
21	Bug fixing	Percentage of 'non-bug' type tasks with respect to total tasks on the board	[%] [number]
22	Testing	Average time-to-test of development tasks	[number/time]
23	Testing	Average testing time based on project board	[number/time]
24	Testing	Percentage of tasks waiting for testing	[%] [number]
25	Others	Number of merge requests without discussion / comments during the code review	[%] [number]

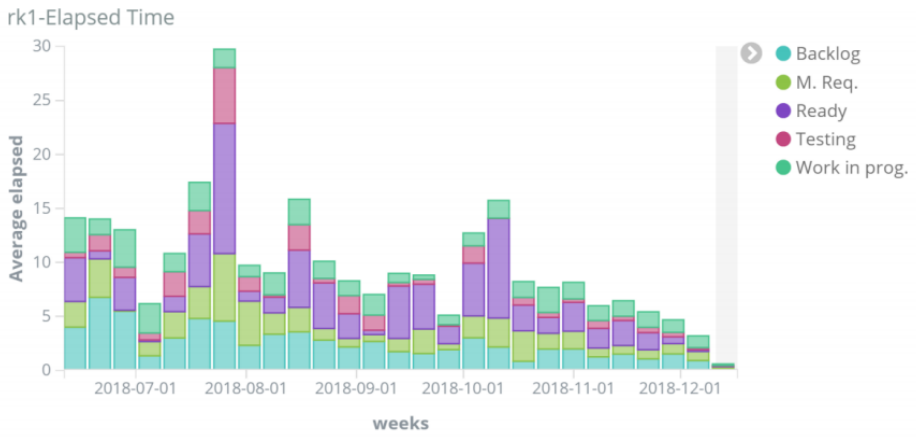

FIGURE 3: Visualization of Process Metrics using Kibana

- *Task estimation metrics*. This category includes the metrics related to planning an effort allocation and analysis of effort/resources consumption. They can indicate the accuracy of such estimation, including average deviation of estimation in relation to the real effort consumption (metrics #7 and #8), total sum of estimated or used resources (#11 and #12) as well as completeness of task estimation (#9 and #10). Similarly to general metrics, those related to task estimation also can be calculated in different dimensions, such as taking assignees, projects, sprints or time range into account.

For the specific stages of the development process, we distinguished the metrics related to task implementation, bug fixing and testing.

- *Task implementation metrics*. This category includes time-based metrics indicating average time of implementation and average time of waiting for implementation (metrics #13 and #14). The other three metrics in this category are related to the task implementation status, namely assignment to the given sprint (#15 and #17) and to a given developer (#14).
- *Bug fixing metrics*. They include the number of tasks with reported bug (metrics #18 and #21) and indication of average time needed/pending time to fix the bug (19 and 20).
- *Testing metrics*. Similarly, metrics #22 and #23 indicate average testing time and average pending testing time, while metric #24 shows the current percentage of the pending task to be tested.
- *Other metrics*. Moreover, we propose a metric showing the number of non-commented merge requests (metric #25), identified as relevant by the CONTRA developers.

B. PROCESS METRICS ASSESSMENT

Next, we report the impact from using the process metrics in the CONTRA pilot project (corresponding to the *evaluating* phase of the Action-Research cycle) on the three research questions: (a) monitoring process performance, (b) keeping a stable product quality level, and (c) improving the estimation of tasks during sprints.

The process of using the metrics was led by the CONTRA Project Owner. With respect to process performance and product quality, the Project Owner analysed monthly the strategic indicators rendered by the Q-Rapids dashboard for these two concepts.

1) Process Performance

Figure 4 shows an example, in which we can see that *Product Quality* remains at a steady level, while in the case of *Process Performance*, there are significant changes.

The Product Owner and the Scrum Master wanted to know the reason behind the process performance change of behaviour, so they used the Q-Rapids detailed view capability applied to the *Process Performance* strategic indicator (see Figure 5). As a result, they were able to identify that the *Issues Velocity* factor suffered a significant improvement over the month, i.e. the development team is increasing the development velocity. Instead, the *Process Performance* just experienced minor fluctuations.

Next, in order to understand in more detail the reason behind the issues' velocity improvement, they used the detailed view capability to visualise the evolution of their influencing metrics. Figure 6 indicates that the *Average Resolved Issue Throughput* (last 7 days) metric improved over the month.

When discussing in the team the reason for this improvement, it became apparent that metrics visualization via the Q-Rapids dashboard allowed the Product Owner to improve his understanding on several aspects of the Scrum process, which had remained unknown before using Q-Rapids. In other words, Product Owners were relying on anecdotal evidence rather than real-time collected data of their development process.

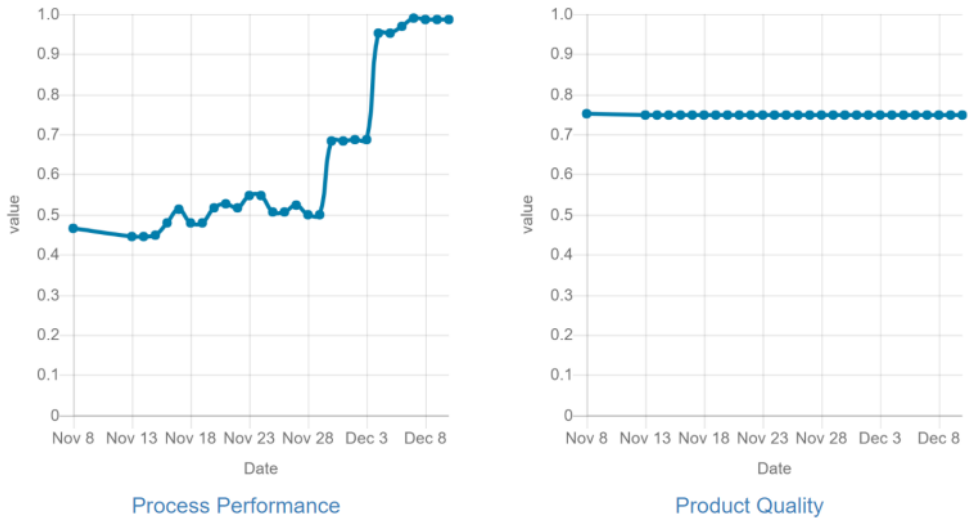

FIGURE 4: Strategic Indicators Evolution View

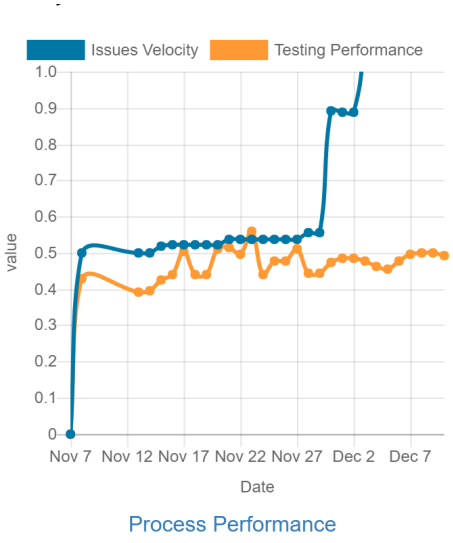

FIGURE 5: Detailed View for Process Performance indicator

2) Product Quality

As Figure 4 shows, the metric shown in the Q-Rapids dashboard did not bring extra value to the development process in this pilot project. The reason may be that other tools like SonarQube were already in place in the company, and therefore the code quality factors were already addressed. In any event, the Scrum Team considered it positive that despite adding new features to the products (i.e., many tasks being closed during the sprint), the product quality remained at a steady level without acquiring technical debt.

3) Task Estimation

The lack of mechanisms for visualizing task estimation was reported before starting the study. By using the process metrics visualised with Q-Rapids, the Product Owner was able to see, as part of the *Testing Performance* factor, low values for the metric *Density of Estimated Development Tickets (last 7 days)* (Figure 6 (b), green line). In order to learn more, the Product Owner switched to the Kibana view on metrics and noticed that there were 37 non-estimated issues in the analysed timespan (see Figure 7).

This capability to smoothly switch into Kibana from Q-Rapids was highly appreciated by the CONTRA team. For instance, by using the Kibana dashboard, looking at the circular diagram, the Product Owner can identify key persons, e.g. the developers with the highest number of assigned issues (see Figure 8). The Kibana dashboard also includes information for analysing other statistics related to a given developer, e.g. the average time of fixing a bug or correction (see Figure 9). After clicking and selecting the particular developer's

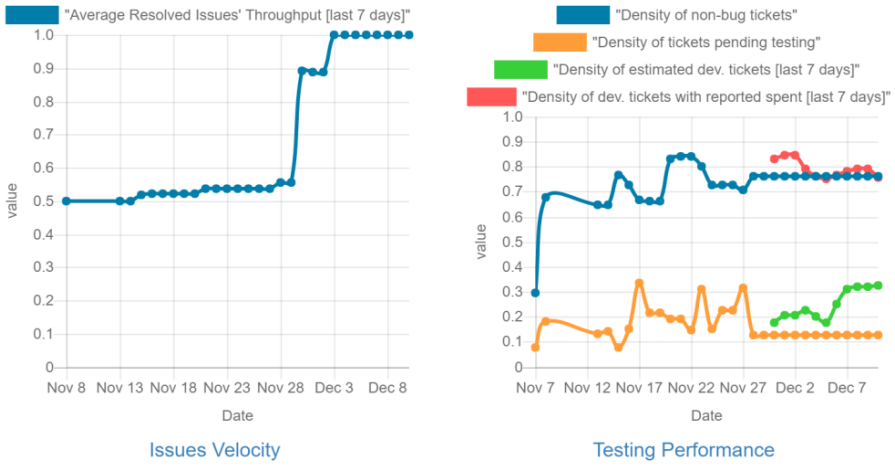

FIGURE 6: Detailed View for Process Performance factors

filters	Unique count of iid	Last 50 iid
READY	65	659, 677, 717, 712, 659, 714, 724, 720, 699, 706, 630, 712, 659, 478, 718, 721, 721, 478, 693, 715, 712, 659, 706, 630, 698, 681, 630, 706, 683, 711, 645, 659, 710, 709, 704, 623, 707, 645, 697, 704, 694, 697, 704, 645, 694, 658, 697, 683, 696, 695
estim=0	37	659, 717, 712, 659, 720, 712, 659, 712, 659, 645, 659, 710, 709, 704, 623, 707, 645, 697, 704, 694, 697, 704, 645, 694, 658, 697, 696, 695, 687, 696, 660, 645, 645, 686, 479, 684, 657, 682, 682, 680, 645, 657, 481, 672, 646, 663, 631, 666, 653, 625
spent=0	10	717, 714, 704, 707, 704, 704, 696, 695, 696, 660, 686, 667, 650

FIGURE 7: Kibana dashboard including non-estimated issues

surname, the data related to this developer is filtered out. Now, the Product Owner can see that 7 out of the 9 tasks completed by this developer were non-estimated in terms of completion time. This helps to identify those tasks that were not estimated and take actions, which results in improving the estimation of tasks from sprint to sprint.

C. METRICS SELECTION

In order to evaluate the usefulness of the metrics for the CONTRA Scrum Team (*specifying learning* phase in the Action-Research cycle), we executed a retrospective session involving the Scrum Master and one developer from CONTRA, and three researchers from the Q-Rapids project. The retrospective session was divided into three main activities: (a) exploration of relevant process metrics used during six months in the CONTRA pilot project; (b) an open feedback session discussing the reasons of the impacts of using the process metrics; (c) documenting the results of the session in a template for the impacts of process metrics.

As a result of this session, the metrics that were considered more valuable are (see Table 2):

- Metric #7: Estimation accuracy per development tasks (per developer in project in specific timespan).
- Metric #9: Number of development tasks with lacking estimation of effort to be spent ("estimated") per project per developer.
- Metric #10: Number of development tasks with lacking value of effort used ("spend") per project per developer.
- Metric #11: Total sum of estimated effort values ("estimate") per project per developer.
- Metric #12: Sum of used effort ("spend") per project per developer.
- Metric #18: Number of development tasks with reported bug.
- Metric #19: Average time of task correction based on project board.
- Metric #20: Average time-to-correct of task based on the project board.
- Metric #21: Percentage of 'non-bug' type tasks to total tasks on the board.

These metrics significantly improve management of such processes as task estimation and bug fixing, which are crucial in rapid software development of high quality and stable software. Moreover, after applying those metrics, team management is now more efficient and transparent.

VI. DISCUSSION

The overall assessment of the process metrics in the CONTRA case proved their value to the company. The proposed, calculated and visualized process metrics (using either the Q-Rapids dashboard, Kibana views or even ad-hoc visualizations developed at ITTI) were assessed as useful by the CONTRA Scrum Team and some of the metrics are now used in practice not only in the pilot project but company-wide.

A. PROCESS METRICS IN THE SCRUM PROCESS

Once the selection was made, at each Scrum Team retrospective meeting, the team usually spends 15 to 20 minutes on visualizing and analysing these selected process metrics. Process metrics are a great fit since this part of the meeting

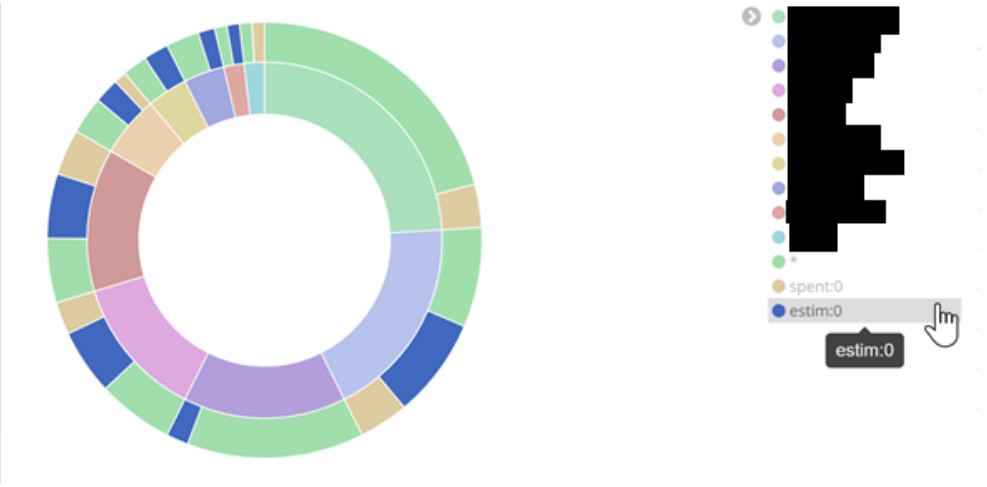

FIGURE 8: Kibana dashboard including a chart for developers' assigned issues per developer. Black boxes are used to hide developers' real names.

filters	Unique count of iid	Last 50 iid
READY	9	809, 713, 804, 628, 713, 775, 796, 700, 747, 688, 713
estim=0	7	809, 804, 628, 775, 796, 747, 688
spent=0	3	809, 628, 796

FIGURE 9: Kibana dashboard including developers' statistics

is devoted to people, processes, tools, lessons learnt and how to improve the way of working. Of course the role of the Product Owner and Scrum Master is to make those discussions and displays interesting, but this turned out to be an easy job for them, because developers usually like statistics and some trends/graphs, such as those shown in the previous section. These results, trends and metrics values are used to motivate the team and improve the process, and also to find the problems in order to resolve them.

B. BENEFITS AND ADOPTION OF PROCESS METRICS AT ITTI

The most important advantage of the process metrics perceived by ITTI is the focus on the process and team effectiveness. The proposed solution has improved the way developers now report time spent on issues/tickets and allows for comparison to the effort planned.

Including dashboards and the process metrics into the ITTI software development process enhanced and improved the willingness and efficiency in reporting spent time and planning the effort. Moreover, the gap between effort planned and spent is continuously decreasing which means Product Owners, Scrum Masters as well as developers estimate much better.

From more practical advantages and possible decisions, the proposed metrics allow for efficient tracking of tasks/issues in the project, per developer and per sprint (or the chosen timespan).

As shown in the previous section, now the efficiency of each developer can be checked. What we found out is that the optimal reported time/effort should be close to 4 days (which means that 1 day is spent on unreported aspects (e.g. when experienced developers help those less experienced ones), and this is well understood and justified). What Product Owners and Scrum Masters mostly seek is the information of bottlenecks of the process, and basically how much time the ticket 'lies' in each phase of the process. The board of the process at ITTI is presented in Figure 10. It consists of 8 steps, and the proposed process metrics nicely show the status of tickets in the project, per phase of the process, per developer and in the given timespan.

At ITTI, in the project CONTRA, the solution showed at first that the bottleneck was in the testing phase. Such situation facilitated a quick decision to engage more testers, but did not solve all the problems. While looking at process metrics, the company found out that the bottleneck shifted to the 'merge request' phase. Such a situation meant that experienced developers (those who can perform code review and merge) do not have enough time and resources to perform tasks in this phase of the software development process. The situation is now solved by granting the rights to perform 'merges' to medium-experienced developers in order to improve the overall process.

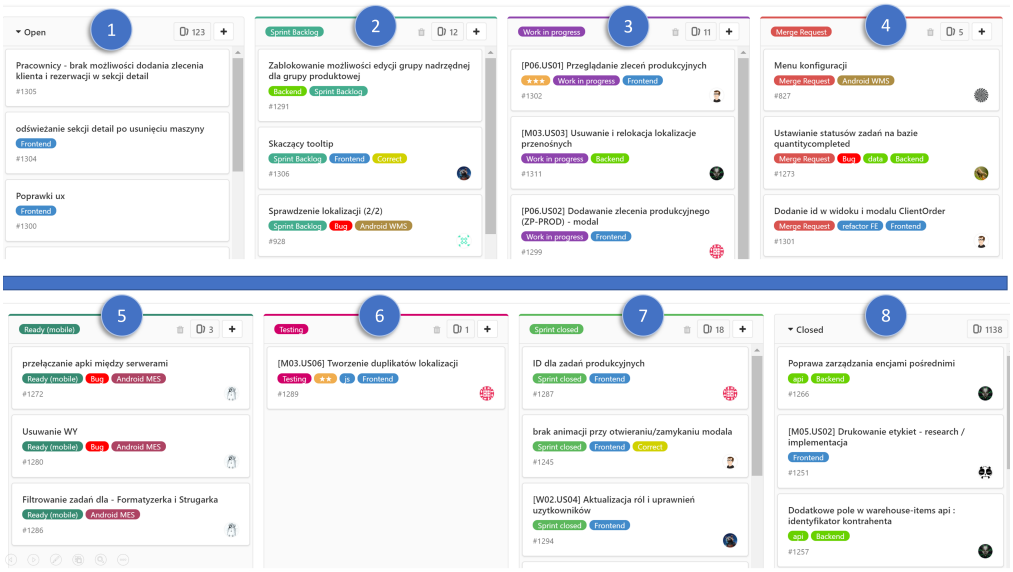

FIGURE 10: The view of the process board used at ITTI (the board is in Polish (the real used one) but any other language can also be used)

C. CONSIDERATIONS ON HUMAN ASPECTS

An important aspect to note is that some of the proposed process metrics, while calculated per developer, have to be used wisely by the Product Owners and Scrum Masters taking into account a plethora of human factors and aspects. This is extremely important especially now (as of 2019), when we have the employee-market in IT world and developers are lacking. At ITTI, the usage of the process metrics is also compliant to the General Data Protection Regulation, as well as to some practical guidelines such as those from the European project CIPHER [7].

VII. THREATS TO VALIDITY

As with any empirical study, there might be limitations to our research method and findings. This section discusses possible threats to validity in terms of construct, conclusion, internal, and external validity and emphasises the mitigation actions applied.

Construct validity. The retrospective session enabled us to further elaborate the practical relevance of the process metrics with two members of the Scrum team. The use of quantitative and qualitative measures and observations reduced the mono-method bias. Furthermore, we aimed at creating a safe environment, encouraging the participants to highlight any negative aspects and make suggestions for the improvement of the process metrics. Finally, some of our results could be caused by not optimal implementation of the process metrics (See Section VI.A). Still, these results are useful for others to learn how to build such an infrastructure in realistic settings.

Conclusion validity. To ensure the reliability of this evaluation, the measurement plan and procedure were documented in detail. Additionally, the results were reviewed by the Scrum team. In this way, we mitigated risks such as fishing for results during the analysis, which would have led to a subjective analysis.

Internal validity. We evaluated the integrated Q-Rapids solution by drawing a convenient sample of a Scrum master and a developer. One limitation of our work is that we were not able to get a random sample of participants in the pilot project. In addition, we defined an evaluation protocol in advance, which included a specific description of our planned procedure and the order of using the materials, i.e., an explanation with all the steps that had to be performed. After all the partners had agreed on the final version of the evaluation guidelines, we executed the evaluation accordingly. This should mitigate the fact that we needed to split the work of conducting the evaluation among different researchers and partners. Some of the five researchers who conducted the evaluation were involved in developing the Q-Rapids tool components. To minimise that bias, we made sure that in each case there were at least two researchers present; one acting as the moderator/experimenter and one as the observer, to emphasise that the participants could speak freely.

External validity. Our results are tied to the context of CONTRA. Our goal was to better understand practitioners' perception. We characterised the environment as realistically as possible and studied the suitability of our sampling (see Section IV.A). GitLab is one of the most extensively used software management tools in SME software development companies. Therefore, we can expect that these metrics analysis may provide actionable insights to a software development company for improving the quality of their processes.

VIII. CONCLUSIONS

In this paper we presented an approach to the definition and utilisation of process metrics related to Agile software development. This has been implemented with: the formulation of

a set of process metrics, their assessment in a real project, and the description of the practical and empirical usage in a particular SME company.

More precisely, in terms of the research questions:

- RQ1: The major contribution of the paper is the very needed solution to monitor the performance of each phase of the software development process. The solution includes a subset of effective process metrics.
- RQ2: Indeed, the major benefit for ITTI is the positive impact on the stability of the CONTRA system while adding new features. In fact, the offered software product (CONTRA) needs customisation for each client (while the domains of clients' businesses vary significantly). The proposed process metrics are continuously used to help assuring the quality and stability of the software.
- RQ3: The value of the proposed solution are the mechanisms for visualizing task estimation, e.g., trace tasks (tickets, issues) live (in real time). By using these process metrics visualisations, the Product Owner in the Scrum Team of ITTI was able to improve task estimations.

Even though our findings are based on a particular SME company and product, we believe that the presented findings on process metrics and Q-Rapids usage can be applicable to a wider context. "If the forces within an organization that drove the observed behavior are likely to exist in other organizations, it is likely that those other organizations, too, will exhibit similar behavior" [40].

In fact, most SME software development companies use SCRUM-like processes as the one presented in Fig 10, and would be interested in the practical metrics related to processes and team effectiveness. Even though the used tools or labels or names of the process phases may be different, the solution is general, although of course it would require some customisation. It is worth to note that, in this paper, we showed real benefits based on a real implementation for GitLab; however other tools such as JIRA can also be used as the data source, and in fact, the connectors to JIRA are already implemented and available as the output of the Q-Rapids project[5].

Indeed, the proposed metrics and related Q-Rapids solutions fill the current need for tools related to the processes in Agile software development. Most tools focus on software quality or continuous integration, without the measures for the process. Basically, there is only one competing solution that could be used to analyze the process, namely GitLab Time Tracker. However, as shown in the paper, we propose a wider set of calculated process metrics, better visualization as well as much more enhanced analysis capabilities.

IX. ACKNOWLEDGEMENTS

This work has received funding from the European Union's Horizon 2020 research and innovation programme under Grant agreement no. 732253. We would like to thank all the members of the Q-Rapids H2020 project consortium. This work is also supported by University of Science and Technology in Bydgoszcz, Poland.

[5]https://github.com/q-rapids/

REFERENCES

[1] E. Arisholm and L. C. Briand, "Predicting fault-prone components in a Java legacy system,", in ISESE '06: *Proceedings of the 2006 ACM/IEEE international symposium on Empirical software engineering*, pp. 8–17, 2006.

[2] V. Basili, G. Caldiera, and H. Rombach, "The Goal Question Metric Approach," *Encyclopedia of Software Engineering*, 1:528–532, 1994.

[3] A.M. Bhatti, H.M. Abdullah, and C. Gencel, "A model for selecting an optimum set of measures in software organizations," in European Conference on Software Process Improvement. Springer, pp. 44–56, 2009.

[4] M. P. Boerman, Z. Lubsen, D.A. Tamburri, and J. Visser, "Measuring and monitoring Agile development status," In *Proceedings of the Sixth International Workshop on Emerging Trends in Software Metrics*, pp. 54-62, IEEE Press, 2015.

[5] E. Caballero, J. A. Calvo-Manzano, and T. S. Feliu, "Introducing Scrum in a very small enterprise: A productivity and quality analysis," *Commun. Comput. Inf. Sci.*, vol. 172, no. May 2014, pp. 215–224, 2011.

[6] M. Choraś, R. Kozik, D. Puchalski, R. Renk, "Increasing product owners' cognition and decision-making capabilities by data analysis approach," *Cognition, Technology & Work*, Springer, 2019.

[7] M. Choraś, R. Kozik, R. Renk, W. Holubowicz, "A practical framework and guidelines to enhance cyber security and privacy," In *International Joint Conference - CISIS and ICEUTE, 8th International Conference on Computational Intelligence in Security for Information Systems*, Burgos, Spain, 15-17 June, 2015, 495, 2015.

[8] V. Claudia, M. Mirna, and M. Jezreel, "Characterization of software processes improvement needs in SMEs, in *International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE)*, IEEE, pp. 223–228, 2013.

[9] M. Díaz-Ley, F. García, and M. Piattini, "Implementing a software measurement program in small and medium enterprises: A suitable framework," *IET Software*, Vol. 2, No. 5, pp. 417–436, 2008.

[10] K.-K. Dikert, M. Paasivaara, and C. Lassenius, "Challenges and success factors for large-scale agile transformations: A systematic literature review," *J. Syst. Softw.*, 119, 2016.

[11] Y. Dubinsky, D. Talby, O. Hazzan, and A. Keren, "Agile metrics at the Israeli Air Force," in *Agile Development Conference (ADC'05)*, pp. 12–19, 2005.

[12] T. Dybå, "Factors of software process improvement success in small and large organizations: An empirical study in the scandinavian context," *Proc. of the 9th European Software Engineering Conference*, pp. 148-157, 2003.

[13] W.A. Florac and A.D. Carleton, "Measuring the Software Process," Addison-Wesley, 1999.

[14] A. Fuggetta, "Software Process: A Roadmap," *Proc. Conf. Future of Software Eng.*, pp. 25-34, 2000.

[15] O. Gómez, H. Oktaba, M. Piattini, F. García, "A Systematic Review Measurement in Software Engineering: State-of-the-Art in Measures," In: *Software and Data Technologies*. pp. 165–176. Springer Berlin Heidelberg, 2006.

[16] L. Guzmán, M. Oriol, P. Rodríguez, X. Franch, A. Jedlitschka, and M. Oivo, "How Can Quality Awareness Support Rapid Software Development? - A Research Preview", in *REFSQ 2017*, pp.167-173, 2017.

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

[18] M. Heričko, R. Torkar, and A. Živkovič, "Software fault prediction metrics: A systematic literature review," *Inf. Softw. Technol.*, vol. 55, no. 8, pp. 1397–1418, 2013.

[19] M. Kasunic, "The state of software measurement practice: Results of 2006 survey," Technical Report, CMU/SEI-2006-TR-009, ESC-TR- 2006-009, Software Engineering Institute (SEI), 2006.

[20] R. Kozik, M. Choraś, D. Puchalski, R. Renk, "Q-Rapids framework for advanced data analysis to improve rapid software development," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, issue 5, 1927-1936, Springer, May 2019.

[21] B. Kitchenham, "What's up with software metrics? – A preliminary mapping study," *Journal of Systems and Software*, Volume 83, Issue 1, Pages 37-51, 2010.

[22] 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, pp. 143–163, June 2015.
[23] C.Y. Laporte, S. Alexandre, and R.V. O'Connor, "A software engineering lifecycle standard for very small enterprises," *Software Process Improvement*, pp. 129–141, 2008.
[24] F. Van Latum, R. Van Solingen, M. Oivo, B. Hoisl, D. Rombach, and G. Ruhe, "Adopting GQM - Based measurement in an industrial environment," *IEEE Softw.*, vol. 15, no. 1, pp. 78–85, 1998.
[25] L. López, S. Martínez-Fernández, C. Gómez, M. Choraś, R. Kozik, Liliana Guzmán, A. M. Vollmer, X. Franch, and A. Jedlitschka, "Q-Rapids Tool Prototype: Supporting Decision-Makers in Managing Quality in Rapid Software Development," CAiSE Forum 2018.
[26] S. Martínez-Fernández, P. Jovanovic, X. Franch, and A. Jedlitschka, "Towards automated data integration in software analytics," In *Proceedings of the International Workshop on Real-Time Business Intelligence and Analytics*, p. 6, ACM, 2018.
[27] S. Martínez-Fernández, A.-M. Vollmer, A. Jedlitschka, X. Franch, L. López, P. Ram, P. Rodríguez Marín, S. Aaramaa, A. Bagnato, M. Choraś, and J. Partanen, "Continuously Assessing and Improving Software Quality With Software Analytics Tools: A Case Study." *IEEE Access* 7, pp. 68219–68239, 2018.
[28] M.G. Mendonça, and V.R. Basili, "Validation of an approach for improving existing measurement frameworks," *IEEE Transactions on Software Engineering*, 26 (6), 484-499, 2000.
[29] R. Moser, W. Pedrycz, and G. Succi, "A comparative analysis of the efficiency of change metrics and static code attributes for defect prediction," in ICSE '08: *Proceedings of the 30th international conference on Software engineering*, pp. 181–190, 2008.
[30] D.J. Paulish and A.D. Carleton, "Case Studies of Software-Process- Improvement Measurement," *Computer*, vol. 27, no. 9, pp. 50-57, Sept. 1994.
[31] K. Petersen, C. Gencel, N. Asghari, D. Baca, and S. Betz, "Action research as a model for industry-academia collaboration in the software engineering context," In *Proceedings of the 2014 international workshop on Long-term industrial collaboration on software engineering (WISE '14)*. Association for Computing Machinery, New York, NY, USA, 55–62, 2014. https://doi.org/10.1145/2647648.2647656
[32] F.J. Pino, F. García, and M. Piattini, "Software process improvement in small and medium software enterprises: A systematic review," *Software Quality Journal*, Vol. 16, No. 2, pp. 237–261, 2008.
[33] P. Ram, P. Rodríguez, and M. Oivo, "Software Process Measurement and Related Challenges in Agile Software Development: A Multiple Case Study," In *International Conference on Product-Focused Software Process Improvement*, pp. 272-287, Springer, Cham, November 2018.
[34] P. Ram, P. Rodríguez, and M. Oivo, "Success Factors for Effective Process Metrics Operationalization in Agile Software Development: A Multiple Case Study," In *Proceedings of the 2019 international conference on software and system process.*, ACM, May 2019.
[35] C.R. Prause, A, Hönle, "Emperor's New Clothes: Transparency Through Metrication in Customer-Supplier Relationships," In: Kuhrmann M. et al. (eds) *Product-Focused Software Process Improvement, PROFES 2018. Lecture Notes in Computer Science*, vol 11271. Springer, 2018.
[36] D. Radjenović, M. Heričko, R. Torkar, and A. Živkovič, "Software fault prediction metrics: A systematic literature review," *Information and Software Technology*, Vol. 55, No. 8, pp. 1397–1418, 2013.
[37] I. Richardson and C.G. Von Wangenheim, "Guest editors' introduction: Why are small software organizations different?" *IEEE Software*, Vol. 24, No. 1, pp. 18–22, 2007.
[38] P. Rodríguez, J. Markkula, M. Oivo, and K. Turula, "Survey on Agile and lean usage in Finnish software industry," *Proceedings of the 2012 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, Lund, pp. 139-148, 2012.
[39] P.S.M. dos Santos, and G.H. Travassos, "Action Research Can Swing the Balance in Experimental Software Engineering," *Advances in Computers* 83: 205-276, 2011.
[40] P.B. Seddon, and R. Scheepers, "Towards the improved treatment of generalization of knowledge claims in IS research: drawing general conclusions from samples," *European journal of information systems*, 21(1), 6-21, 2012.
[41] E. Shihab, Z.M. Jiang, W.M. Ibrahim, B. Adams, A.E. Hassan, "Understanding the impact of code and process metrics on post-release defects: a case study on the eclipse project," In *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, p. 4, ACM, September 2010.
[42] J. Soini, "A survey of metrics use in finnish software companies," In 2011, *The International Symposium is the Empirical Software Engineering and Measurement*, pp. 49-57, IEEE, September 2011.
[43] R. van Solingen, and E. Berghout, "The Goal/Question/Metric Method: A Practical Guide for Quality Improvement of Software Development", McGraw-Hill Publishing Company, 1999.
[44] M. Söylemez, A. Tarhan, "Challenges of software process and product quality improvement: catalyzing defect root-cause investigation by process enactment data analysis." *Softw. Qual. J.* 26, 779–807, 2018.
[45] M. Staron and W. Meding, "Factors determining long-term success of a measurement program: An industrial case study," *e-Informatica Softw. Eng. J.*, vol. 1, no. 1, pp. 7–23, 2012.
[46] M. Sulayman and E. Mendes, "A systematic literature review of software process improvement in small and medium web companies," *Advances in Software Engineering*, pp. 1–8, 2009.
[47] T. Tahir, G. Rasool, W. Mehmood, C. Gencel, "An evaluation of software measurement processes in Pakistani software industry," *IEEE Access*, 6, 57868–57896, 2018.
[48] B. Tanveer, L. Guzmán, and U.M. Engel, "Understanding and improving effort estimation in Agile software development." *Proc. Int. Work. Softw. Syst. Process - ICSSP '16*, 41–50, 2016.
[49] A. Tarhan and S.G. Yilmaz, "Systematic analyses and comparison of development performance and product quality of Incremental Process and Agile Process," *Inf. Softw. Technol.*, 56, 477–494, 2014.
[50] A. Tosun, A. Bener, and B. Turhan, "Implementation of a software quality improvement project in an SME: A before and after comparison," *Conf. Proc. EUROMICRO*, pp. 203–209, 2009.
[51] M. Unterkalmsteiner, T. Gorschek, A.K.M.M. Islam, C.K. Cheng, R.B. Permadi, R. Feldt, "Evaluation and measurement of software process improvement-A systematic literature review," *IEEE Trans. Softw. Eng.*, 38, 398–424, 2012.