Key performance indicators used to measure the adherence to the iterative software delivery model and policies

C Făgărășan[1], C Cristea[2], M Cristea[2], O Popa[1], C Mihele[1] and A Pîslă[1]

[1] Design engineering & Robotics Department, Machine Building Faculty, Technical University of Cluj-Napoca, 103-105 Muncii Avenue, Cluj-Napoca, Romania
[2] Electrical Machines and Drives Department, Faculty of Electrical Engineering, Technical University of Cluj-Napoca, 26-28 G. Barițiu Street, Cluj-Napoca, Romania

[1]office.cristif@gmail.com

Abstract. This paper analyzes the Agile metrics embedded in the widely adopted issue tracking tools and used by agile teams to deliver software products. Organizations face challenges when they are required to measure the performance of all the teams operating within the company to provide a holistic view of the software delivery performance at a higher level. A new set of key performance indicators (KPIs) is introduced that can help software companies standardize their delivery across the organization and provide an overview of the software delivery performance that can serve as an insight for the top management teams. The new metrics are applied to a project as part of a case study. The conclusions highlight that the metrics provided by the issue tracking systems do not provide insights into the non-functional requirements of the software products, the methodology applied during the delivery, and the quality of each increment produced. The proposed set of KPIs addresses these shortcomings.

1. Introduction

Launching a new software product requires adaptability, and time to market has become essential if a competitive advantage is to be gained. For this reason, Agile delivery methods were introduced in software development more than twenty years ago through methodologies like Scrum, Kanban, Extreme Programming, and Scrumban. These methodologies provide frameworks that support complex projects based on an iterative approach [1]. Software development teams lean towards adopting the new ways of working, but this does not mean that the benefits described as part of the methodologies will follow. Nuances of the frameworks need to be understood through practice, and therefore, key performance indicators (KPIs) need to be introduced to measure the effectiveness of new ways of working [2]. Product success depends on the performance of each team member, and how the agile teams use the methods, processes, and tools to deliver software impacts success positively or negatively [3]. As each project is different, it requires a unique approach to decision-making and problem-solving, which implies that the metrics used to measure performance need to be generic so they can be applied across multiple projects and teams. The famous quote by Eliyahu M. Goldratt says, "Tell me how you measure me, and I will tell you how I will behave" [4] underlines a problem in measuring the performance of different organizational functions and projects, given their complexity and specificity. The performance measurement system needs to be adapted for each part of the organization and may differ depending on the product or services offered. A software company that provides services for multiple clients may have hundreds of projects and agile teams. Many metrics can be used to measure agile teams' performance and delivery, depending on the organization's interests and perspectives. For example, the Project Management Office within a software company is interested in measuring project performance regarding quality, costs, and delays, also known as the project management triangle [5], ensuring profitability, rentability, and in-time and qualitative delivery, which guarantees stakeholder satisfaction. The software development life cycle includes multiple phases: planning, analysis, design, implementation, and maintenance. Each of these has numerous sub-steps that need to be completed to consider a phase complete [6]. Several project management systems on the market are used for issue management and software development task tracking that is needed to execute a project successfully. These are used to plan, monitor, and implement software projects efficiently, ensuring data management is not overwhelming. Moreover, they provide a significant number of automation and reports which can be used to increase the efficiency of the implemented workflows and processes. Based on historical data, issue tracking systems are used to store performance data and build a statistical prediction of what can be delivered in a certain amount of time. These tools provide virtual boards where all software development life cycle steps are displayed. An organization will host all its development tasks in this system, providing descriptions, attachments, details about the status, and the possibility to communicate directly on each issue through comments. Jira issue tracking system, developed by Atlassian, is one of the tools that has become popular in the last twenty years. This tool can track the team's performance and offers features for planning purposes like managing the backlog and sprint planning [7]. This paper analyses the most critical metrics in the Jira issue management system. It underlines the challenges organizations face when measuring the quality of the product increments delivered using the iterative approach. A new system is proposed as a solution to the identified challenges, and its efficiency is demonstrated by applying it to a software project.

2. Jira agile metrics used for measuring the software development delivery

Most software companies use the Jira tool, produced by Atlassian to implement an agile software development workflow. It has a built-in workflow management system with visual boards that can be customized for the needs of each software team and project. The built-in reporting section can be used to find out the status of a project iteration or team performance. For example, the tool can predict how many iterations are needed to deliver the project based on team velocity per iteration. It is recommended that the timebox of the iterations remains the same throughout the project to gain predictability and stakeholder confidence regarding the incremental delivery process. As part of project execution, project managers and the team can use a series of reports as input to the decision-making process, correct variations, and control the scope and the schedule. A series of reports were selected and analyzed for a detailed analysis, and an exemplification of the KPIs used to measure the team and project performance.

2.1 Burnup chart

Figure 1. Iteration Burnup Chart [8]

Figure 1 provides a visual representation of the completed tasks concerning the overall scope of the iteration. The X-axis indicates the time in days until the end of the iteration, while the Y-axis represents the total effort required to complete the entire iteration scope. The red and green line distance represents the remaining effort needed to complete the sprint scope.

2.2 Burndown Chart
This graph shows the actual versus the estimated amount of work in a timeboxed iteration. As seen in Figure 2, the X-axis indicates the time, while the Y-axis shows the probability that the team is going to complete all the tasks with the iteration, known as a sprint in Scrum. The total remaining estimated effort is calculated, and it is reported as the entire effort needed to complete all the tasks part of the iteration. Therefore, this is a helpful tool that helps the team manage its progress towards fulfilling the scope of each iteration and adapt according to the variations that may appear. It is recommended that the project manager present this graph in daily team sessions to create real-time awareness about the team's performance.

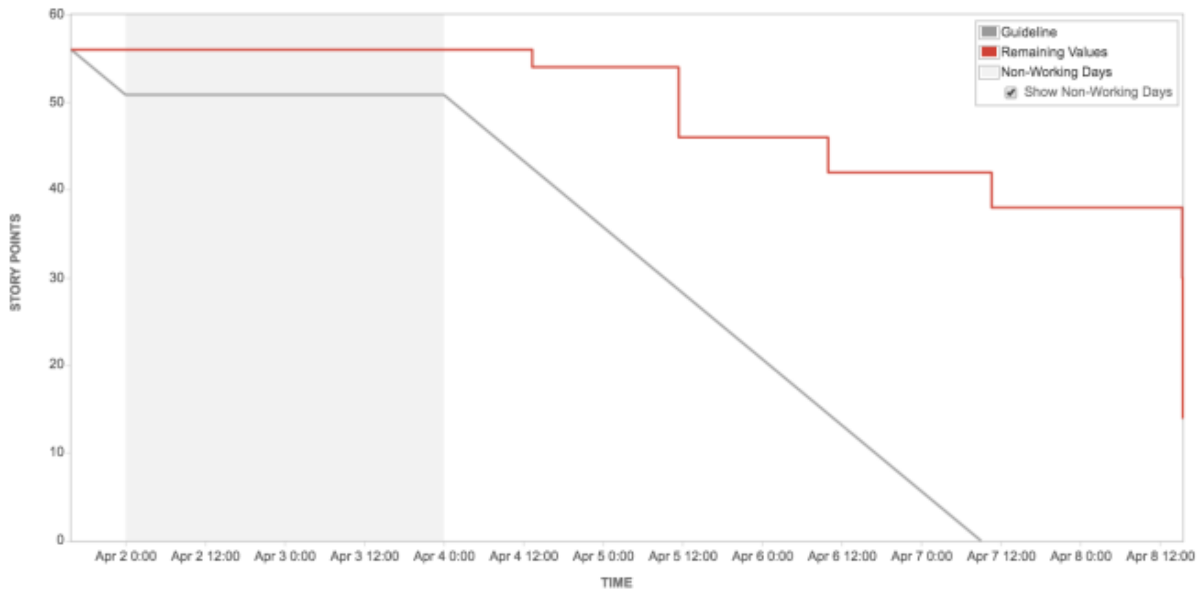

Figure 2. Iteration Burndown Chart [8]

2.3 Epic report
Each feature request added in Jira can be broken down into smaller deliverables that will be completed in multiple iterations. This report shows the development team's progress toward delivering the requested feature. The remaining effort required to provide the functionality is highlighted. Moreover, the project manager can see a list of incomplete, unestimated tasks that need to be developed as part of the feature.

2.4 Version Report
A software product may have several important milestones throughout the development process. These are named versions in Jira and can have multiple features that are completed throughout several iterations. This report tracks the team's progress in relation to the overall scope of the product version. Moreover, it indicates a predicted release date based on the average progress rate of the team since the work on the version started. The effort required to achieve the full scope of the version is also considered. The report provides the following information regarding the completion date of the version:

a. *Average release date, based on the daily average speed of the team and the remaining effort needed to complete the version's scope.*

b. *Pessimistic launch date if the team's delivery speed will decrease by 10%.*

c. *Optimistic launch date if the team will increase their delivery speed by 10%.*

Figure 3. Jira Version Report [8]

2.5 Velocity Chart

Story points are a unit of measure in agile, translating into the effort required to implement a task entirely. The report presented in Figure 4 shows how many story points a development team delivers per iteration. In this report, more iterations are displayed. The development teams use these statistics to estimate the time needed to address the iteration's scope and analyze if Jira's estimations match the project schedule. The main elements of the chart are as follows:

a. *The X-axis displays the last seven iterations completed by the development team. This data is used to calculate the average velocity of the team.*

b. *The Y-axis shows the number of story points.*

c. *The gray bar shows the total number of story points added at the beginning of the iteration. If more tasks are added after the iteration starts, their corresponding story points will not be considered.*

d. *The green bar shows the number of story points completed at the end of each iteration.*

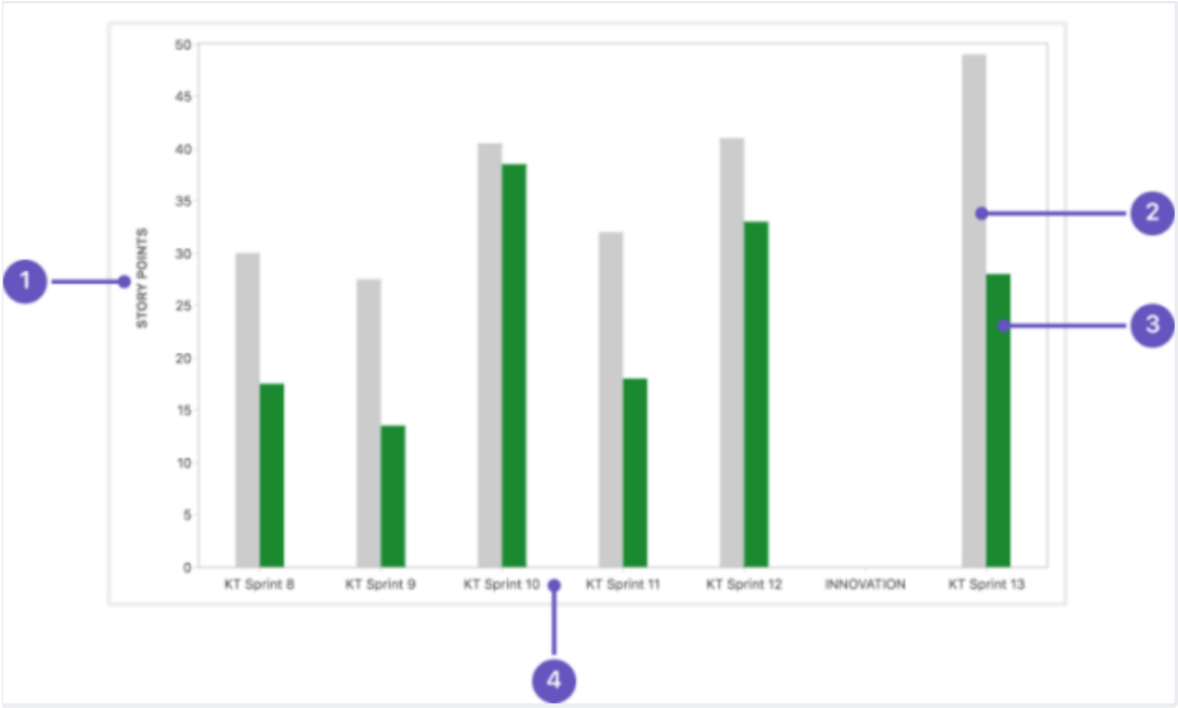

Figure 4. Jira Velocity Chart [8]

2.6 Release Burndown Chart

This report can track the development team's speed, providing a prediction of iterations needed to complete the version based on historical data. Figure 5 shows the team's real progress in relation to the effort required to complete a product version. This chart is similar to the Version Report but optimized explicitly for teams with releases at the end of each iteration. The two gray bars on the right represent the number of predicted iterations needed to complete the release.

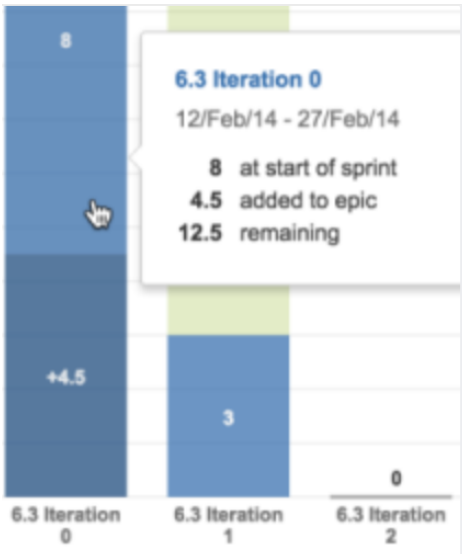

Figure 5. Release Burndown – Iteration Analysis [8]

The essential elements of the release burndown chart are the iterations bar and the number of predicted remaining iterations. Figure 6 shows the components of one iteration bar:

a. *The light green section is the work completed in one iteration.*

b. *The light blue section represents the estimated work remaining that needs to be completed as part of the current iteration.*

c. *The dark blue section represents the work added after the iteration started.*

d. *The sum between the green and the light blue sections represents the total effort added at the beginning of the iteration.*

e. *The sum between the light blue and the dark blue sections represents the total remaining estimated effort at the end of the iteration.*

3. Challenges that organizations face when required to measure the performance of all software development projects and teams' company-wide

The reports presented in the previous section help assess the team's progress concerning the full scope of an iteration or product version. The metrics implemented as part of the project management systems existing on the market show the actual versus planned progress, providing useful information to project managers in identifying variations and estimating based on historical data. The issue tracking systems used for task management increase the likelihood of project success by providing transparency and traceability of project work. The reports are helpful at the project or team level, but when organizations want to measure the aggregated performance at a company level, it proves challenging. Each organization has standards that must be applied to all projects related to the methodology used for implementation or specific internal procedures.

The software delivery model can differ depending on what frameworks are used within the organization. It is usually defined and owned by the Project Management Office (PMO), the department in charge of the project management methodology. One of their primary responsibilities is to streamline and optimize the delivery processes and introduce metrics that confirm the desired quality level. There are parts of the software delivery life cycle that technical managers own, but they need to be validated with the PMO. The defined processes and practices must be applied to each project, but the project manager can tailor them partially according to the value they bring to its deliverables. Usually, it turns out to be challenging to validate that these procedures are applied, and the company's delivery standards are met.

Most companies use an audit system with a predetermined recurrence. The organization ensures that all these practices are applied and that the quality of the increments delivered through development iterations are in line with company standards. An essential input to this process is the feedback system used for assessing the activity of employees concerning their attitude and contribution to the quality of the delivery processes. All these processes are performed manually, and the company must devote part of its resources to building this system. The major problem is that the whole process can be time-consuming, and the interpretation of the results is not straightforward, which can cause delays in receiving the results, so certain aspects of project delivery can be corrected.

Project information like iteration progress and its quality, as well as feedback on the procedure applied, should reach project managers as soon as possible to allow them to react and apply corrective or preventive actions as early as possible in the development process. The project's success is closely related to the quality of the product increments delivered to the customers. It is important to measure that all increments provided align with the quality requirements and attributes defined at the company level and verify that mandatory procedures and processes are applied to the projects carried out within the organization. The software development life cycle involves multiple functions such as PMO, Architecture, Quality Management, and Business Analysis. All critical roles within the company need to be aligned on the newly proposed metrics that will be globally measured to ensure that all aspects are considered when defining the KPIs that will indicate the quality of an increment and the compliance with the company procedure and processes.

4. Proposed KPIs used for measuring and standardizing the software delivery model

4.1 Defect removal efficiency (DRE)
During an iteration, the quality assurance team verifies the newly introduced functionalities before the delivery to the customer. Quality engineers identify defects, record them in the project management system, and send them to the development team to be addressed. According to the testing principles described by the International Institute of Software Testing (ISTQB), exhaustive testing is impossible, meaning that the delivered increments may include defects that are not identified by the quality engineers [9]. The new functionalities are handed over to the final customer after verification. If the defects are identified in the production or acceptance environment, the testing process used to verify the functionalities can be improved. All the defects are unlikely to be identified before the final delivery. Each identified post-delivery defect can be used to justify improving the quality control processes. These need to be analyzed to determine possible shortcomings in the verification process. The score of this metric contributes to the global score of the increment.

4.2 Major Bugs which were not addressed before increment delivery (MB)
Software organizations may have different quality standards, depending on their policies. According to the criteria defined by ISTQB, defects can be blocking, critical, major, minor, and trivial [10]. When quality engineers identify deficiencies, these are typically sorted and addressed according to their severity. In most software projects, all defects that fall into the first three categories must be addressed before the increment is delivered to the end customer. Minor and trivial flaws do not affect the functionality, but the stakeholders need to decide on fixing them. The KPI is measured as the ratio of minor and major defects that have not been resolved by the delivery time. This metric stimulates the delivery of software products with as few major defects as possible. It is not recommended to deliver increments with major flaws in a production environment. Still, there are situations when stakeholders accept the product along with these known defects for various reasons.

4.3 Reopened Defects (RI)
Once the defects reach the development team and are resolved, they return to the verification stage. These are rechecked and closed if they have been resolved. Otherwise, the defects need to be reopened and sent back to the programming engineers. The defects are highlighted using the project management system through a predefined custom workflow. If multiple defects are reported, a deficiency in the code quality delivered by the development team is highly possible. For each reopened issue, it is necessary to identify the root cause of the recurrence and apply a permanent solution. This metric assesses the efficiency of programming engineers in finding the solutions for the identified flaws.

4.4 Acceptance Criteria definition (AC)
All deliverables need clear and transparent acceptance criteria to provide helpful information to the development team to estimate the work and deliver it within the agreed timeframe. There are multiple ways of defining them, but the Gherkin domain-specific language is the most common. After consulting with various stakeholders, product managers and business analysts define the acceptance criteria. This can be set as a standard at the organizational level, and its adoption can be measured through the AC metric.

4.5 Work breakdown definition (WB)

The Business Analysis department decides the scope definition and granularity standards, depending on the project's phase. When the project is initiated, the project's scope is high-level defined. As the project goes into the planning stage, the requirements are more explicit, and the scope can be broken down into multiple deliverables with a clear definition. As more time is dedicated to planning the project, the estimations available for each deliverable are more accurate. All requirements must be straightforward during project execution, with clearly defined acceptance criteria. In the end, the granularity of the scope offers a higher degree of confidence in the project estimates and helps avoid any project delays. Research shows that nine out of ten global projects are delayed and cause cost overruns [11]. The WB metric ensures that the scope is broken down into small tasks that do not need more than eight hours of work. The KPIs in Table 1 can be grouped into two categories: Increment Global Quality (IGQ) and Increment Scope (IS).

Table 1. Data needed to measure the KPIs

Metric	Category	Data needed	Formula
Defect removal efficiency (DRE)	IGQ	Total number of defects identified before the product increment delivery (RBQ)	$DRE = \frac{RBQ}{RBC + RBQ} * 10$ (1)
		Total number of defects identified by the final customers after the product increment delivery (RBC)	
Major Bugs which were not addressed before the increment delivery (MB)	IGQ	Opened minor defects delivered in a product increment (MD)	$MB = \frac{MD}{MD + MMD} * 10$ (2)
		Opened major defects delivered in a product increment (MMD)	
Reopened Defects (RI)	IGQ	Total number of defects in a product increment (GD)	$RI = \frac{GD}{GD + DI} * 10$ (3)
		Total number of defects that have been reopened (DI)	
Acceptance Criteria definition (AC)	IS	Total number of tasks in a product increment (GT)	$AC = \frac{TAC}{GT} * 10$ (4)
		Total number of tasks with clearly defined acceptance criteria (TAC)	
Work breakdown definition (WB)	IS	Total number of tasks which have an estimate between zero and eight hours (TWB)	$WB = \frac{TWB}{GT} * 10$ (5)
		Total number of tasks that have an estimate bigger than eight hours (TNWB)	

The data needed to calculate the metrics can be extracted automatically from the issue management system used by agile teams to develop software. Each score is multiplied by a product increment's maximum score. The next step is to aggregate the metrics and calculate the delivered increment's final score, depending on the weight (w) of each KPI. The weights assigned to each metric are decided at the organization level, depending on its objectives regarding delivery. These can change over time as software companies focus on new methodologies or introduce new processes that make the metric system scalable in the software development industry. The final formula which will be used to calculate the score of each delivered product increment is the following:

$$\mathrm{PIS} = (w_1 * \mathrm{IGQ}) + (w_2 * \mathrm{IS}) \quad (6)$$

Both categories presented above are calculated by applying the following equations:

$$\mathrm{IGQ} = (w_3 * \mathrm{DRE}) + (w_4 * \mathrm{MB}) + (w_5 * \mathrm{RI}) \quad (7)$$

$$\mathrm{IS} = (w_6 * \mathrm{AC}) + (w_7 * \mathrm{WB}) \quad (8)$$

5. Case study: applying the new set of metrics to a software development product increment

For this case study, let us consider a product increment with the following data presented in Table 2. The data was automatically extracted from Jira, a cloud issue management system, and the values were calculated based on the formulas presented in Table 1. The weights chosen to calculate the score for the object of the present case studies can be seen in Table 2 and Table 3. The objective of the project management office for the project presented in this case study is to increase the quality of the delivered product increments, which is reflected in the values of the assigned weights of each KPI and category.

Table 2. Measured values for the proposed KPIs and their selected weights

Metric	Measured Value	Weight (%)
Defect removal efficiency (DRE)	DRE = 6.9	$w_3 = 50\%$
Major Bugs which were not addressed before increment delivery (MB)	MB = 4.0	$w_4 = 25\%$
Reopened Defects (RI)	RI = 8.5	$w_5 = 25\%$
Acceptance Criteria definition (AC)	AC = 2.3	$\mathbf{w_6} = 50\%$
Work breakdown definition (WB)	WB = 3.2	$w_7 = 50\%$

Table 3. Weights selected to calculate the general score of the product increment

Metric category	Weight (%)
Increment Global Quality (IGQ)	$\mathbf{w_1 = 80}\%$
Increment Scope (IS)	$\mathbf{w_2 = 20}\%$

The final score of this increment for PIS is 5.81. Each company must decide if delivering product increments with a given score is acceptable. The firm where this project was developed decided that product increments with a score of less than eight cannot be delivered to customers. In this case, the project manager in charge of delivery needs to find solutions that can be implemented before the product delivery to increase the score. Given that the IGQ category gives 80% of the final score, the project manager needs to look at the metrics that influence the IGQ category and ensure that the scores are improved. In this case, the DRE and MB scores need to be increased, and the development team should reduce the number of major bugs from the product increment. The IS category's score is low but does not impact it exponentially because it represents only 20% of the final product increment score.

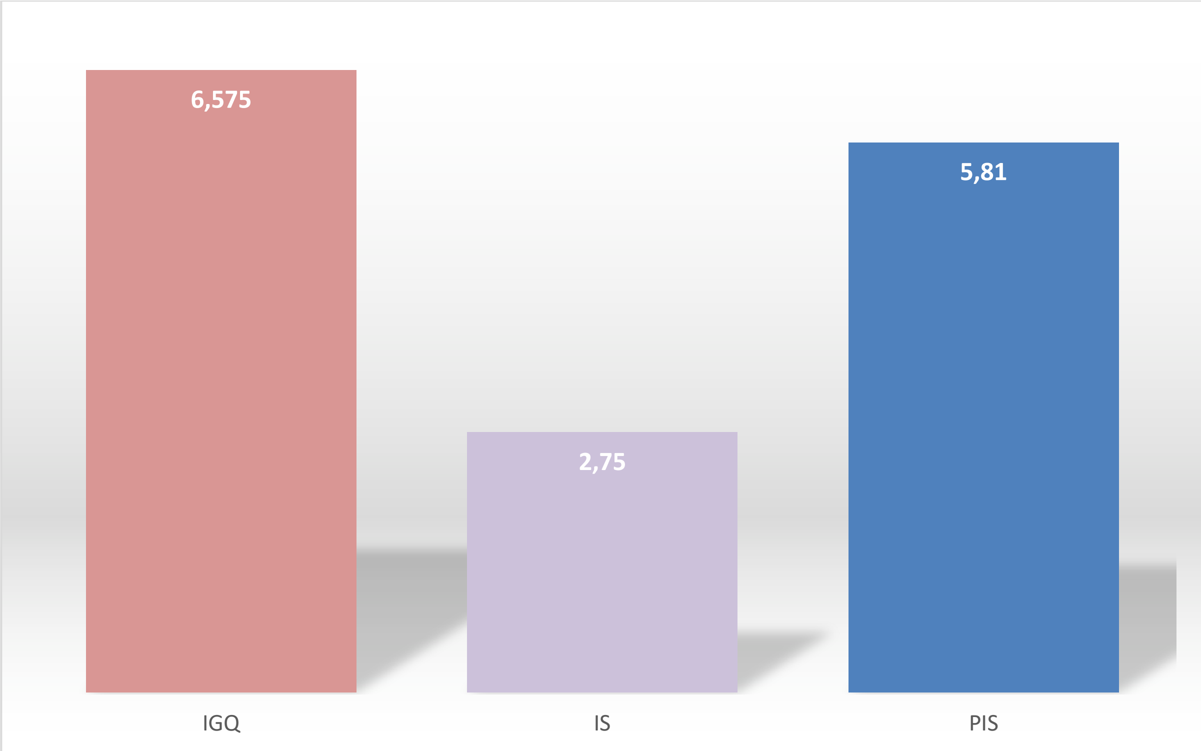

Figure 6. Determined values of each KPI category and the final product increment score

6. Conclusions

The software systems available on the market provide incomplete information to decision-makers regarding KPIs. The provided outcomes focus more on iteration and team metrics rather than specific attributes of the product increment that is being delivered. Many companies still use human resources to audit the quality of the product delivered to customers and to ensure adherence to organizational policies. The proposed model supports the stakeholders in gaining insights into the quality of a delivered increment. As more data becomes available, the score is automatically updated to reflect the status of the increment. At the beginning of an iteration, the score might be low, but as the delivery progresses, the score needs to increase. A lower score signifies that issues with the project or company delivery must be addressed, and it requires no domain-specific knowledge to reach this conclusion. The project manager assigned to the project needs to monitor the product increment score and take preventive or corrective actions to achieve a score accepted within the organization. The new system is scalable and can be applied to all product increments that a software company delivers. Furthermore, the product increment's scores can be aggregated into a project score and even into a company one, which can provide insights into the global quality of the project or company delivery that is useful for top managers. More metrics relevant to the agile delivery model need to be included for implementing a dashboard where all the scores are aggregated to provide transparency in regards to the quality of the delivery at all levels: iteration, project, and company. This is going to be studied in future works.

7. References

[1] Kuhrmann M and Ternite T 2006 *Implementing the Microsoft Solution Framework for Agile Software Development as concrete Development-Method in the V-Modell XT*, International

Transactions on Systems Science and Applications, Special Issue Sections in ENASE'06, volume 1, 119 - 126
[2] Albers A, Heimicke J, Spadinger M, Degner N and Duehr K 2019 *The product developer in the centre of product development: a systematic literature review on describing factors*, International conference on engineering design, ICED19
[3] Heimicke J, Mellert T and Albers A 2020 *Performance Evaluation of Agility in Product Development using targeted KPIs*, ISPIM Connects Global
[4] Davico G 2020 *Tell me how you measure me and I will tell you how I will behave*, https://www.linkedin.com/pulse/tell-me-how-you-measure-i-behave-part-1-gianluca-davico/
[5] Adlane H, Seghiri R, Aouane M, Berrid N and Chaouch A 2021 *The project management triangle assessment in aeronautical industries, morocco: focus on eco-logistics*, Management systems in production engineering, Volume 29, Issue 2, 132-138
[6] Hema V, Thota S, Kumar S, Padmaja c, Krishna C and Mahender K 2020 *Scrum: An Effective Software Development Agile Tool*, IOP Conference Series Materials Science and Engineering 981(2):022060
[7] Ortu M, Destefanis G, Adams B, Murgia A, Marchesi M and Tonelli R 2015 *The JIRA Repository Dataset: Understanding Social Aspects of Software Development*, 11th International Conference on Predictive Models and Data Analytics in Software Engineering
[8] Atlassian website 2022 *https://support.atlassian.com/jira-software-cloud*
[9] Nayyar A 2019 *Instant Approach to Software Testing: Principles, Applications, Techniques and Practices*, BPB Publishers, ISBN: 978-93-88511-162
[10] Yorkston K 2021 *Performance Testing: An ISTQB Certified Tester Foundation Level Specialist Certification Review*, SpringerLink, ISBN: 978-1-4842-7254-1
[11] Egwim C, Alaka H, Toriola-Coker L, Balogun H and Sunmola F 2021 *Applied artificial intelligence for predicting construction projects delay*, Machine Learning

Acknowledgments
This paper was supported by S.C Tremend Software Consulting S.R.L, Bucharest, Romania.