


Scrum Watch: a tool for monitoring the performance of Scrum-based work teams

Florencia Vega

(UNICEN, Tandil, Argentina
vega.florencia@gmail.com)


Guillermo Rodríguez

(ISISTAN (UNICEN-CONICET), Tandil, Argentina

 <https://orcid.org/0000-0003-4125-3998>, guillermo.rodriguez@isistan.unicen.edu.ar)


Fabio Rocha

(UNIT and ITP, Aracaju, Brazil

 <https://orcid.org/0000-0002-0512-5406>, fabio.gomes@souunit.com.br)

Rodrigo Pereira dos Santos

(UNIRIO, Rio de Janeiro, Brazil

 <https://orcid.org/0000-0003-4749-2551>, rps@uniriotec.br)

Abstract: Agile Methods propose an approach for developing software based on an iterative and incremental life cycle model, in which needs and solutions evolve through collaboration between multi-functional and self-organized teams. As such, agile practices in work teams are gaining much momentum. To meet the demanding level of projects, agile software development also has to keep up with several challenges. In this context, software industry has chosen to use several tools to ease development and communication between different teams' members. However, these tools generate overwhelming volumes of data that hamper decision-making by project managers. To address this issue, we present *Scrum Watch*, a tool-based approach that focuses on generating, through cloud-based technologies, graphic elements and reports that assist project managers with information to support decision making. Results obtained from an undergraduate Systems Engineering course through a capstone project confirm the feasibility of the proposed approach, which exploits the benefits of the availability and visualization of process and product metrics.

Keywords: agile methods, project monitoring and control, cloud-based technologies, capstone project, decision-making

Categories: D.0, D.2.1, D.2.9, H.2

DOI: 10.3897/jucs.67593

1 Introduction

In heterogeneous, geographically dispersed and diversified application environments, data generation from software development projects has increased exponentially and involves the systematic compilation, organization, analysis, and extraction of insights from data warehouses [Boscarioli et al. 2017]. Due to the generation of constant feedback in agile methods, it becomes difficult for project managers to keep up with such updates [Papke-Shields et al. 2020]. In this context, we argue that a solution approach should

allow project managers to develop data analysis skills towards effectively addressing project management demands.

Out of the agile methods, Scrum has gained considerable attention. In 2019, a survey conducted by *VersionOne* concluded that 72% of companies surveyed used Scrum or a hybrid that includes Scrum [Version One 2020]. Iterative and incremental development, along with daily and retrospective meetings, enable team members to early provide insights about how agile practices are performed. They also allow the timely identification of issues in the understanding of agile practices and the concept of corrective measures to be taken in subsequent sprints by team members. Unlike waterfall-like approaches, team members receive feedback before expecting the end of a project.

In this context, we aim to address the challenge regarding the decision-making process in the context of agile software development. To reach our goal, we propose *Scrum Watch*, an approach whose main objective is to generate, through cloud-based technologies, graphic elements and reports that allow the project manager's decision-making, as well as risk detection at an early stage. Thus, *Scrum Watch* enables project managers to generate analyses based on the history of teams' work under Scrum methodology. To do so, we applied a research method comprising three phases: *Phase #1*: approach definition, which refers to an ad hoc literature review on agile methods, cloud-based technologies and ETL (Extracting-Transforming-Loading), as well as the identification of related work; *Phase #2*: solution proposal, which refers to the specification and development of *Scrum Watch*; and *Phase #3*: evaluation, which refers to the use of *Scrum Watch* in a real context based on a field study.

To evaluate *Scrum Watch*, data from course projects from the Systems Engineering program during the years 2017-2019 were used. In this course, participants were organized into teams and, following the Scrum methodology, developed requirements asked by the customer (i.e., professor). The teams followed a defined process supported by tools such as Jira¹, Sonar², and XWiki³, among others. The participants of the project are senior students of the 3/4 year degree program at the Faculty of Exact Sciences from the Department of Computer Science—Universidad Nacional del Centro de la Provincia de Buenos Aires (UNICEN). Since the *Scrum Watch* approach is supported by Amazon cloud services, infrastructure knowledge is no longer required to keep it. Regarding data visualization, *Scrum Watch* provides a powerful, fast tool with a vast range of visualizations.

This article is organized as follows apart from this introduction. In Section 2, we describe the background. Section 3 reports the related work. The design of our approach is shown in Section 4. The field study design is then explained in Section 5. Results are reported in Section 6. Discussion is illustrated in Section 7. Section 8 presents the main threats to validity. Finally, conclusions and future work are presented in Section 9.

2 Background

This section presents the theoretical foundations used for the creation of the *Scrum Watch* tool. Firstly, Agile Methods main concepts are introduced. Secondly, two Amazon Web Services (AWS), such as Amazon Quicksight and Amazon Relational Database Service (RDS) are shown. Finally, ETLs (Extract, Transform and Load) are also presented with

¹ <https://www.atlassian.com/es/software/jira>

² <https://www.sonarqube.org/>

³ <https://www.xwiki.org/xwiki/bin/view/Main/>

their phases, processes and models to build them, as well as the data warehouse model to store their output.

2.1 Agile Methods

The advent of the Agile Manifesto brought about improvements within the software development culture [Beck et al. 2001]. Essentially, the Manifesto outlines a new perspective on software development that focuses on mobility, flexibility, soft skills and the potential in a short time to deliver new high-value products and services to the market [Highsmith 2009].

Scrum is an iterative and incremental approach that organizes tasks to make them manageable for small, self-organized and cross-functional teams, as well as allowing teams to systematize software development projects, encouraging good project management practices by fostering teamwork cooperation [Schwaber and Beedle 2002].

The roles in Scrum are quite different from the traditional methods. Clearly defined roles and expectations allow individuals to perform their tasks efficiently. There are three roles: Product Owner, Development Team, and Scrum Master. The team members having these roles constitute the so-called Scrum team. The Product Owner represents the client and controls the Project Backlog, and is working closely with the Development Team to provide user story clarity and acceptance. Completing the user stories is the responsibility of the Development Team. The Scrum Master is responsible for initiating the process, assisting the Product Owner in preparing releases and assisting the success of the Development Team by eliminating impediments and making resources available.

2.2 Cloud-based technologies

In this context of agile software development, it is important to have a support infrastructure to support decision making. Due to technological advances and the continuous growth of small and large companies, nowadays new models for the use of IT services are emerging. These models commonly called as cloud computing are defined as virtualized technologies that form a shared structure that allows adding the web-based feature to make it accessible and where the services that were generally found are installed on their own servers [Gibson et al. 2012].

One of those cloud services is called software-as-a-service (SaaS). SaaS is the model in which applications are provisioned that are accessed through the web and that are used directly by end users without having to worry about or take care of scalability, configuration or service updates. This model is very popular also for its scalability, compatibility and easy access from anywhere in the world for users. *Scrum Watch* is built on two of this type of services, namely Amazon Quicksight and Amazon RDS (Amazon Relational Database Service).

AWS Quicksight is a service capable of automatically discovering the data sources of other AWS services and also working with user-imported data sources. Amazon QuickSight is used to create data visualizations, perform ad hoc analysis as well as obtain implicit and explicit information from the data connected to it. Amazon QuickSight enables organizations to scale from hundreds to thousands of users and offers reliable performance thanks to its robust in-memory engine (SPICE) and also only pay if used. It is designed to perform advanced calculations and make data available quickly.

Amazon RDS is a web service that facilitates the configuration, operation, and scaling of a relational database in the cloud. It provides a cost-effective, resizable capacity for a

standard relational database and handles common database administration tasks [Safvati et al. 2017]. Amazon RDS provides access to the MySQL, Oracle and Microsoft SQL Server database engines, among others.

2.3 The Use of ETL

According to El-Sappagh et al. [El-Sappagh et al. 2011], an ETL (Extracting, Transforming, and Loading) is a piece of software responsible for the extraction of data from multiple data sources, as well as their cleaning, customization, reformatting, integration, and insertion loaded into some final database, e.g., in a data warehouse. As its acronym indicates, an ETL consists of three steps, Extraction, Transformation and data loading (Load). They are usually applied by organizations to migrate heterogeneous information from one or more data sources into a final system which can be used as a repository or for analytical purposes such as a data warehouse.

There are three main areas to develop an ETL: the source data area, the final data area, and the mapping to achieve this transformation. The source area has standard models such as entity and relationship diagrams, and the final area also has standard models such as the star or snowflake schema in a data warehouse schema. In the star schema, there is one or multiple fact tables in the center surrounded by a set of dimension tables. On the other hand, in the snowflake schema, there is a hierarchy between the dimension tables; each table is divided into different levels, which gives different levels of granularity.

Regarding the final data area, the design of the data warehouse is considered an important part in the development of the ETL, because its performance is the main concern for the developers. Such performance is based on the query execution time and the complexity to access the data. The execution time of the queries depends purely on the type of data warehouse design used [Sidi et al. 2016].

On the other hand, we highlight the importance of introducing concepts of cloud-based technologies and ETLs, because they are crucial for the development and execution of *Scrum Watch*. The first step of *Scrum Watch* is the definition of the final data model, i.e., the main structure to obtain the information that assists the project manager in making decisions. The data model defines the mappings and transformations that will be later applied in the ETL process to convert the initial data into the final data. Once the data model has been defined, we seek to integrate different sources of information from other databases using ETLs. This process collects information from the Jira, XWiki, and Sonar databases to transform it according to the detailed business rules and store it in the target system. In this process, cloud-based technologies are used.

3 Related Work

Project management is one of the fundamental elements in any software development process. For this reason, at present, there is a wide variety of tools that capture software metrics to measure the performance of the people who make up the work teams for achieving continuous improvement, thus maximizing the quality of the final product. As such, this section describes a set of studies that analyze information for the evaluation of team performance through the implementation of capstone courses applying agile methodologies for software development. Although the selected studies present proposals that were assessed with students, they aim to be applicable in the software industry and used by both agile teams and project managers.

In [Yildirim et al. 2019], the authors have presented a proper agile project management performance measurement model for start-up software companies. To do so, the authors have reviewed all key performance indicators (KPIs) related to agile development. Then, using expert perspectives gathered through in-depth interviews, KPIs from the literature study and content analysis were reviewed and categorized. The data gathering mechanisms for seven strategic KPIs have been identified and designed. Finally, to maintain the agile development measurement paradigm, adjustments to processes and data gathering are recommended.

The authors in [Alaidaros et al. 2018] explained the contemporary issues in the Agile Kanban method's progress monitoring task. As a result, the findings revealed how to meet the gap by creating a better software project monitoring task model using the Agile Kanban method. The authors first defined the components and factors that influence software project monitoring and then suggested an initial model. Extending progress monitoring, establishing optimal Work In Progress (WIP) limits, and showing essential workflow insights are the three primary components of the original model.

In [Scott et al. 2017], the authors presented a metric based on the standard Burndown chart for quantifying individual variances in project progress. They have also presented preliminary findings using the metric in a specific training environment, demonstrating how learning-styles-based instruction can help students enhance their project progress.

Likewise, in [Firdaus et al. 2019], the authors have introduced the issues of the system XYZ Mart as for the availability of items, specifically goods that are not categorized based on the intensity of their sales. Then, there was a need for a system to monitoring the flow of products movements during the transaction and reporting the results to the user. The authors used Scrum to create the XYZ Mart monitoring system to do so. Researchers employed metrics to evaluate the sprint's activities as it was being carried out. The system's results were evaluated from a User Acceptance Test and got a positive user reaction.

Along this line, the authors in [Haidabrus et al. 2021] addressed a literature review that demonstrates how Data Science methodologies can improve project management and project success in several business situations. This study provides new chances to enhance project management evaluation and results for managers, industry, and delivery leaders. The proposed method provides more accurate project management, portfolio management, and agile development by incorporating best practices and project performance data. Furthermore, the outcomes may result in more efficient benefits for several internal and external stakeholders.

To the best of our knowledge, there is no mechanism for tracking and monitoring the performance of Scrum-based work teams, which generate a huge amount of data from tools to support software development processes.

4 *Scrum Watch*

This section introduce *Scrum Watch*, a tool that aims to assist project managers in monitoring project and process performance of Scrum teams by leveraging cloud-based technologies and visualization techniques. With *Scrum Watch*, project managers can check the product's progress by visualizing information from tools for code versioning. On the other hand, project managers can also check the team members' performance and process execution by visualizing information from tools for assets' maintenance and issue tracking. With such information, a project manager might mitigate risks, anticipate project delays, manage resources, and re-organize teams.

General Goal: to address the challenge of the decision-making process in the context of agile software development. On the basis of the general goal, the following specific goals (SGs) are formulated:

SG1: To allow project managers to visualize teams performance.

SG2: To allow project managers to visualize teammates performance.

SG3: To allow project managers to visualize sprint performance.

SG4: To allow project managers to visualize source code quality.

To develop *Scrum Watch*, we followed a methodology that details all the procedures and provides quantitative results. An ETL was the best strategy to obtain only the data of interest to project managers and a data warehouse was the best strategy to centralize it. The challenge in a data warehouse is to integrate, transform and consolidate large volumes of data from multiple sources, thus providing new unified information for analytical purposes.

Scrum Watch is divided into three important steps:

1. Definition of the final data model;
2. Retrieval and transformation of information through an ETL process;
3. Analysis and visualization of information;

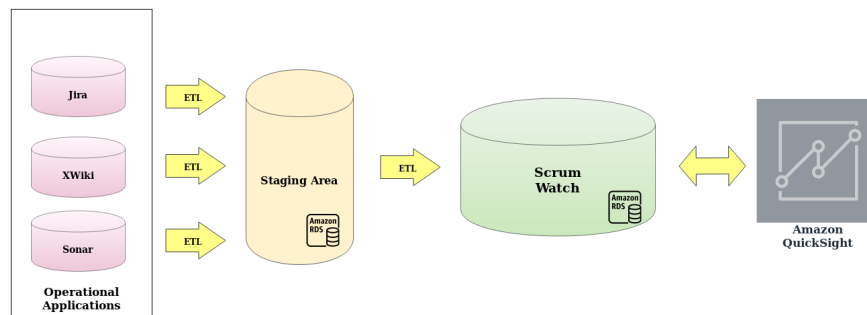


Figure 1: Overview of *Scrum Watch*.

Figure 1 illustrates the general *Scrum Watch* process. Through an ETL, information from three relationship databases is extracted: Jira, XWiki, and Sonar. After that, the data obtained is loaded into a staging area where the transformations defined by the ETL will be applied. Once the data has been transformed, the loading process is carried out to the final data warehouse (star schema). Both the staging area and *Scrum Watch* data warehouses use Amazon RDS. Finally, based on Amazon's visualization service, AWS Quicksight, data analysis, and reporting are enabled to facilitate the manager in making decisions and monitoring Scrum work teams.

For the *Scrum Watch* generation, an ETL was developed. On the one hand, it works on the source code metrics (Sonar tool); on the other hand, it works on the performance of team members based on their interaction with Jira and XWiki tools. Although there is no relationship between source code and teammate in the Sonar data source, it was

included in the same way in order to give the project manager a chance to infer it at the data visualization stage.

Each ETL is developed and designed according to business needs and according to the objectives it pursues. For the implementation of ETL that led to the creation of *Scrum Watch*, a set of technical decisions were analyzed and they will be detailed throughout this section. Each decision was made from searching for the implementation that best suits its goals: assist the project manager in monitoring the performance of work teams under the Scrum methodology; and detect risks in early stages.

To support the process for creating an ETL, there are three different types of models to follow: “Model based on mapping expressions”, “Model using conceptual constructors”, and “Models based on UML environments” [El-Sappagh et al. 2011].

In our approach, the selected model was “Model based on mapping expressions”. In this model, queries to the database are used to carry out the data generation process. These queries are used to represent the mapping between the initial and final data. For the implementation of the mapping, mapping expressions are used, which are documents that indicate how to achieve it. These documents are made during the implementation of the system by a developer based on the needs of the project manager. In the following subsections, the process followed to implement an ETL is explained in details.

4.1 Data Extraction (E)

The ETL extraction process consists of 2 phases:

- Initial extraction;
- Incremental extraction.

In the initial extraction, the data to be transformed and loaded is obtained for the first time. Incremental extraction is where the ETL processes take care of adding the modified information after the last extraction. Since our implementation is tied to static databases, only the first phase of the process will be carried out. However, the approach was implemented with both phases in mind.

In our approach, the data extraction is generated from three relational databases: Jira, Sonar and Xwiki, which are hosted on Amazon’s RDS service. The Jira database provides information on the tasks the team member was working on in a given Sprint. For Jira, we extracted the following fields, among others:

- `jiraissue`: It contains data from the story such as id, name, creation date, description, the person who reported it, to whom it was assigned, workflow to which it belongs, time spent on it etc.;
- `issuetype`: It contains information regarding the type of the issue. The possible types are Bug, Requirement, New Feature, Task, Improvement, Sub-task, Story, Epic, and Technical Task;
- `issuestatus`: It contains information related to the possible status of an issue. Possible values are Open, To Do, Done, Ready for code review, In Progress, Reopened, Resolved, and Closed.

Xwiki provides information on the documents generated in a given period. Regarding xWiki, we extracted the following fields, among others:

- xwikidoc: It lists the document data such as id, full name, title, creation date, and auto-update and creator;
- xwikiattachment: It lists the files attached to the associated document. The author, the creation date, and its size are also listed in the table.

Finally, Sonar has information on the metrics of the developed source code. From Sonar, we extracted the following fields, among others:

- rule: It contains information associated with the rule such as id, name, priority, status and programming language;
- incident: It lists the incidents with their respective data such as id, description, line number where it was originated, severity, status, if it was resolved or not, and a rule that gave rise to it;
- metrics: It lists the metrics that are evaluated on each snapshot.

For data extraction, dumps from the different databases were used and loaded into a temporary staging environment, in which the necessary transformations will later be carried out to convert the input data into the output data of the new data warehouse model. Although the execution can be done on the fly, or as it is commonly said ‘in memory’ in most ETLs, the benefits of adding a new staging area that brings improvements and benefits to the process were analyzed. A staging area is a temporary storage area between the input data and the data warehouse. The inclusion of this area requires an additional cost of resources, development times and duration of the ETL itself. However, this area is a fundamental and widely used area to allow optimal management of the ETL processes and optimal data recovery.

Although the staging area offers high flexibility regarding the data extraction process, it reduces the negative impact on the re-processing of data and reduces the processing time when there are joins with more than two data sources. However, the staging area increases the necessary data space and the total latency of the ETL.

4.2 Data Transformation (T)

The second step of an ETL is the transformation stage. This step is considered the heart of all ETL and is responsible for performing data cleansing, validations and transformations so that the extracted data is compatible with that of the target system.

The goal of transformations, such as cleaning and renaming, is to reduce the amount of data by saving resources and reducing the execution time in transformations that generate data that will later be discarded. For example, Figure 2 shows the “rename” transformation performed between columns of the snapshot table (Sonar source schema) and columns of the final schema (Sonar target schema).

Regarding xWiki, Figure 3 shows the data filtering transformation performed on the three data sources from xWiki. Only the documents generated in the selected period (August-December 2015) are suitable for the subsequent transformations to be applied.

Finally, for Jira, transformations 2 and 3 in Figure 4 convert the time fields rendered in seconds to the YYYY-MM-DD format. For transformation 6, a conditional mapping is performed to evaluate the data in the DUEDATE and RESOLUTIONDATE columns. For the cases in which the RESOLUTIONDATE is greater than the DUEDATE, it assigns the value ‘In Term’; otherwise, ‘Delay’. In this way, there is a new column that indicates whether the issue was completed on time or not.

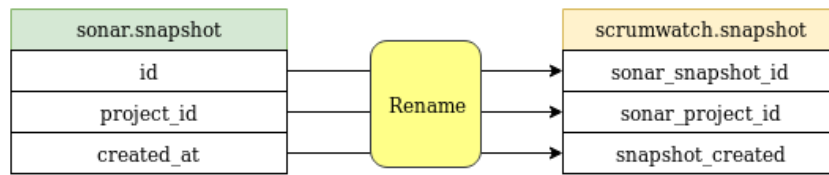


Figure 2: Example of a rename transformation.

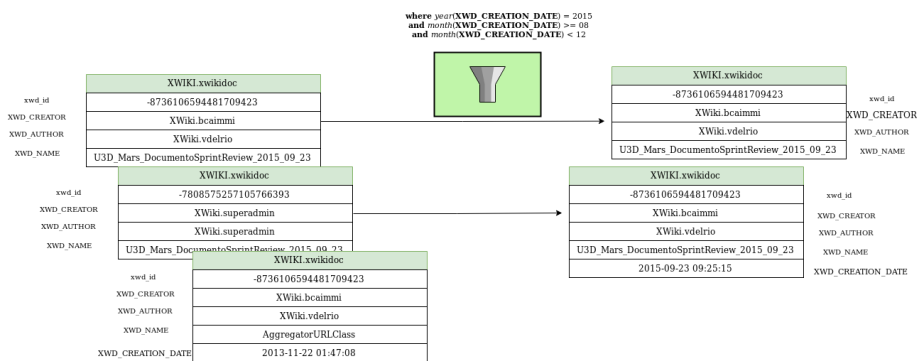


Figure 3: Example of a rename transformation in xWiki.

4.3 Data Loading (L)

The third and final stage of an ETL is represented by uploading the data to the final system. In our approach, the load occurs from the staging area to the final data warehouse. The load is total, i.e., the entire data set is loaded in the first load. Given the advantages offered by cloud services, both the final data warehouse and the staging area of the ETL were hosted in database instances of Amazon's RDS service. The data warehouse was represented with the creation of a new MySQL database instance with the following characteristics:

- Name: *Scrum Watch*
- Version: My SQL 5.7.22
- Instance size: db.t2.micro, 1GB of RAM
- Capacity: 20 GB
- Port: 3306
- Public access: Yes
- Autoscalling: Yes
- Backups: Yes

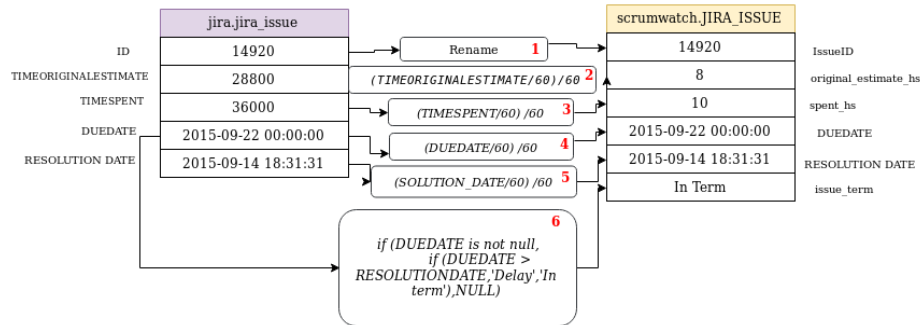


Figure 4: Example of a rename transformation in Jira.

From the aforementioned examples, we present three AWS Quicksight SQL queries to satisfy a project manager's requirement, and the results are displayed in a dashboard. We based on the data warehouse star schema with both dimension and fact tables. The following SQL query gives as a result the dashboard displayed in Figure 5. The requirement consists of obtaining documents created in xWiki grouped by Scrum teams.

```
select T.name, count(D.doc_id) from
Document D inner join Documentation_Fact DF on
D.doc_id=DF=doc_id
inner join Teammate T on T.name=DF.teammate
group by T.name;
```

For Jira, the following SQL query gives as a result the dashboard displayed in Figure 6. The requirement consists of obtaining Jira issues closed grouped by Scrum teams.

```
select T.name, count (JI.type_name) from
Jira_Issue JI inner join
Jira_Fact JF on JI.issue_id=JF.issue_id
inner join Teammate T on T.name=JF.teammate
group by T.name;
```

Considering Sonar, the following SQL query gives as a result the dashboard displayed in Figure 7. The requirement consists of obtaining all code incidents along the project.

```
select I.incident_severity, count(I.incident_id) from
Incident I
group by I.incident_severity;
```

5 Field Study

To evaluate the approach, data from a Systems Engineering course of the Systems Engineering BSc program at the Faculty of Exact Sciences from the Department of Computer Science—Universidad Nacional del Centro de la Provincia de Buenos Aires (UNICEN) during the years 2017-2019 were used, comprising a total of 3 academic semesters. In this course, students were organized into teams and developed requirements requested by the customer (i.e., the first professor), applying the Scrum methodology.

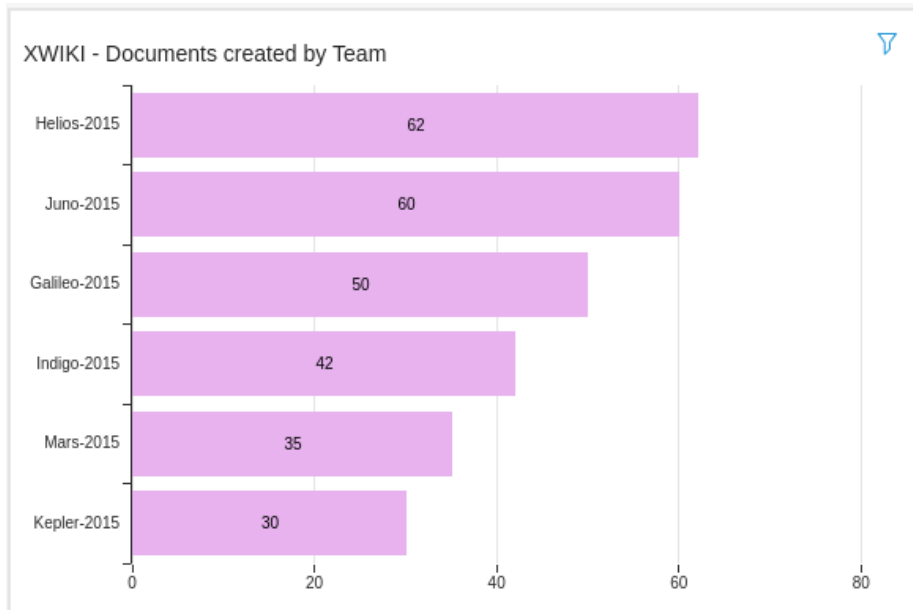


Figure 5: Dashboard showing the created xWiki documents grouped by teams.

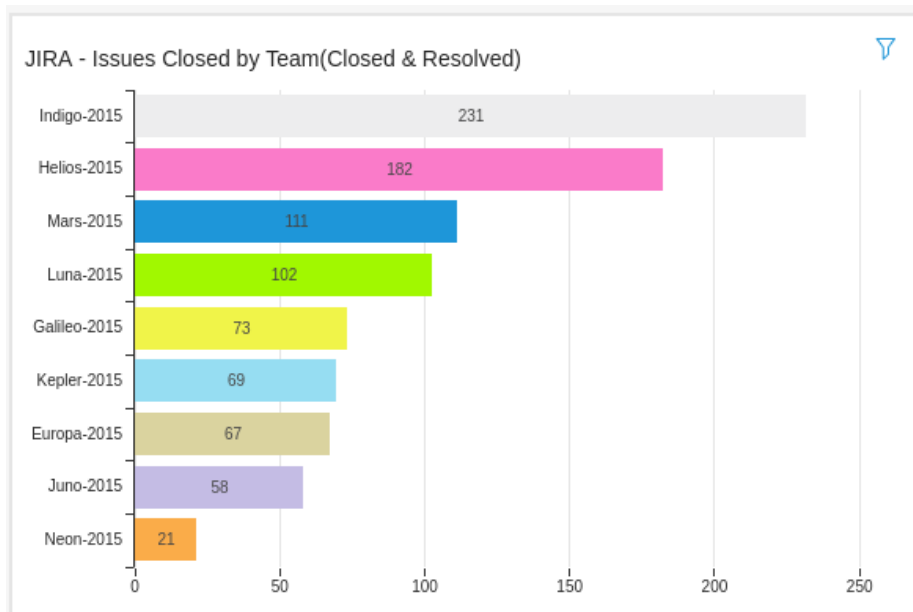


Figure 6: Dashboard showing the closed Jira issues grouped by teams.

A total of 235 students participated in this study (71 in the first semester, 78 in the

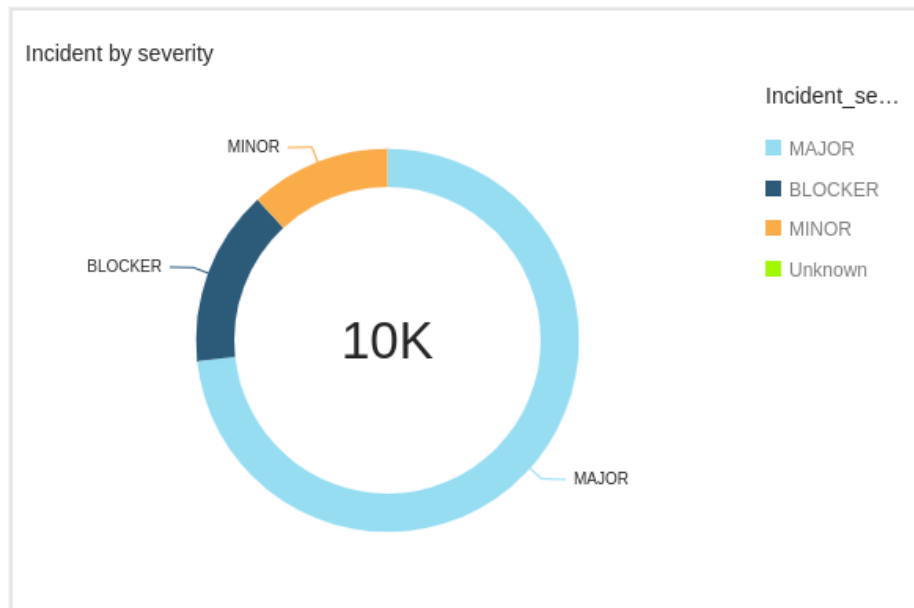


Figure 7: Dashboard showing numbers of Sonar incidents in the project.

second semester, and 86 in the third semester), organized into 30 teams composed by 6-8 members each. The Product Owner (i.e. the second professor) assigned software requirements to each team so that they should carry out a course project during the 3 semesters. The students participating in the study were in their last year of the Systems Engineering course.

The project topic of the course was the same for the three semesters. However, in each semester, the teams had to work on different requirements. In other words, in the first semester (2017), students built a software system following a set of requirements A. Then, in the second semester (2018), students received the software developed in 2017 and added other features by following a set of requirements B. Finally, in the third semester (2019), the students received the software developed in 2018 and added other new features by following a set of requirements C. The sets A, B, and C were different.

The development environment of the teams consisted of three important tools: Jira, Sonar, and XWiki. For the development of *Scrum Watch*, the data extraction is generated from three relational databases of the recently named tools, which are hosted on Amazon RDS services. The course project consisted of designing a virtual campus using the Unity game engine built as architecture with multi-tiered client-server. The virtual campus allows users to access the university facilities, play themed games, and attend virtual classes, using networking channels for chat, e-mail, and website.

6 Results

Four dashboards were created in order to assess our specific goals: *Team Performance*, *Teammate Performance*, *Sprint Performance*, and *Source Code Performance*. Throughout

this section, the most important metrics of each dashboard are displayed along with a brief description of the graphs to represent them. The metrics displayed are those obtained as a result of the survey⁴ conducted with 10 project managers.

6.1 Team Performance (SG1)

The objective of this dashboard is to show metrics at a macro level, in which only the performance of the entire team is evaluated. For this reason, the filters provided are only at the equipment level.

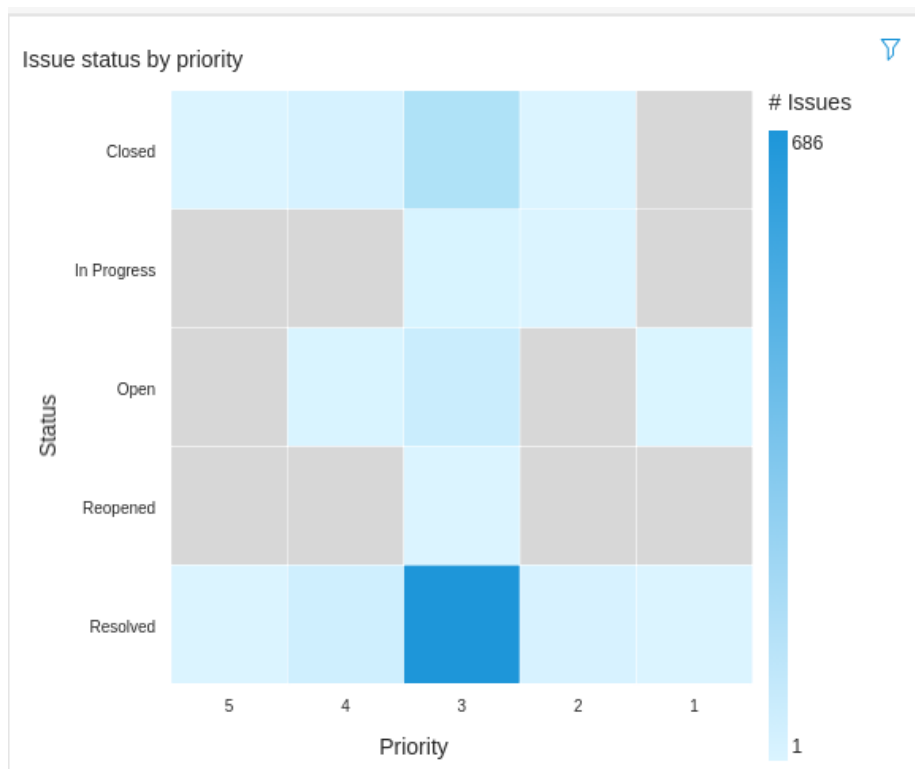


Figure 8: Team Dashboard - Issue status by sprint.

An important finding that arose through the study was the existence (or not) of a relationship between the priority or type of tasks and their resolution. Are the highest priority tasks always solved? What kinds of tasks are never solved? Is there a reason for it? In order to try to obtain a trend or detect atypical cases, thermal or heat maps were used since they allow easy detection with the use of color. Figure 8 depicts a heatmap on a blue scale. The heatmap shows cells containing the number of tasks in a sprint classified into priority and status of resolution. The browner blue, the more issues are

⁴ shorturl.at/qwzAN

in the cell. Remarkably, the heatmap shows that a considerable number of tasks (i.e., issues) with median priority 3 (high=1, ... ,low=5) were resolved during the sprint.

6.2 Teammate Performance (SG2)

In this dashboard category, several graphs are used to assist the project manager in monitoring the performance of the members of their teams. The evaluation and follow-up of a teammate's performance is measured from the issues in which he/she was working and from the documentation generated/updated throughout the sprint in which he/she is evaluated.

The possible statuses of a Jira issue are "Open", "Closed", "Reopened", "Resolved" and "In progress". In order to obtain a comparison of the status of all the tasks of a member, it was decided to make a pie chart to show such division not only in quantities, but also in percentage. With this simple graph, the manager can detect the tendency to close, open, and reopen tasks. A teammate with a high percentage of "Reopened" issues may indicate a low quality of him/her performance since the tasks that were closed at him/her discretion had to be reopened for some reason.

As an example in Figure 9, the metric "Invalid configuration name" refers to the number of documents that a teammate created without following the nomenclature standard defined for the documentation. A KPI (key performance indicator) chart was used to represent it, as these charts compare a key value and an expected target value. In this case, the target value would be the number of documents generated and the key value, the number of documents that do not follow the renaming standard. This graph indicates that this team member did not follow the nomenclature standard in almost 62% of the cases, which could be a wake-up call for the manager to assist the teammate or advise them on such good practices.

6.3 Sprint Performance (SG3)

This dashboard could be considered an intermediate between the Teammate dashboard and the Team dashboard. It offered an intermediate granularity since it allows to see the performance of teams at the sprint level. The objective is to show the improvement of the teams throughout the sprints. As an example of this dashboard, all teams are selected in order to evaluate the overall performance. Logically, it could be done for a particular team.

Figure 10 shows the quantitative metric that represents the number of documents generated by each sprint. As observed, the activity between sprints 2 and 3 has a noticeable difference in comparison to the sprints at the extremes.

6.4 Source Code Performance (SG4)

This dashboard shows metrics obtained from the source code developed by the different work teams. These metrics were extracted from the Sonar tool data source. In an ideal scenario, an attempt would be made to associate the source code with the developer that implements it. However, given the absence of such relationship, only the metrics obtained in the selected period are displayed.

The objective of tools such as Sonar is to make periodic snapshots of the source code to find incidents or issues that violate the rules defined as good practices. The aim is nothing more than to get the most optimal code without possible errors. For this reason,



Figure 9: Teammate Dashboard - Issues General Metrics.

the monitoring of incidents is crucial for the project manager. As such, charts were included in the dashboard and function as counters that show the identified incidents as well as how many were closed or not.

Figure 11 shows some metrics concerning the incidents found in a given team and in a given sprint. These metrics were collected from Jira and Sonar. The metrics show the number of code lines generated, the number of files, the number of methods, the number of directories, the cyclomatic complexity of methods, and the lines of comments of methods, among others. This dashboard could be crucial for decisions made by a project manager regarding teams' monitoring and control.

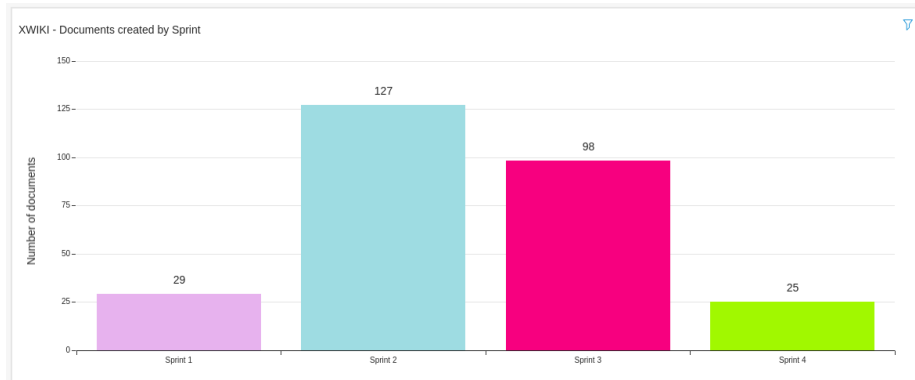


Figure 10: *Sprint performance Dashboard - Documentation by Activity by Sprint.*

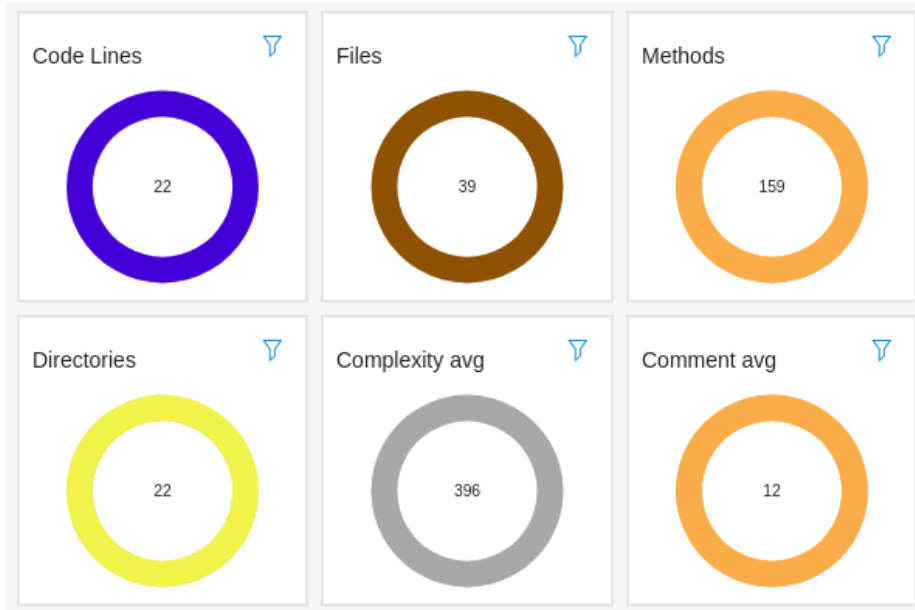


Figure 11: *Source code Dashboard - General Metrics.*

7 Discussion

We found that speed, variety of visualization metrics, data integrity, ease of defining queries to obtain new information, and scalability are some advantages that the *Scrum Watch* approach provides. Our approach uses a data warehouse designed with a star schema as a consolidated data repository. This was chosen because it offers easy maintenance when business logic is dynamic due to the simplicity of its design. We can summarize some benefits of using *Scrum Watch*:

1. there is no limit of team members,

2. project managers might add new metrics to visualize easily,
3. a huge volume of data could be processed and visualized,
4. team members are always aware of the process and project status.

Another benefit of *Scrum Watch* comes from the denormalized table schema, which offers a noticeable speed when it comes to executing data queries. From the perspective of an end-user, the queries to obtain new data are simple since the information is easily retrieved from some conditions and joins operations. Another advantage is that the data generated can serve multiple purposes, even though the data warehouse was designed to assist a manager in making decisions to evaluate the performance of work teams. It is worth noting that the visualizations presented in this article are only a reduced sample of the graphics available in *Scrum Watch*.

Thus, we can state that *Scrum Watch* has reached the proposed objectives, allowing decision-makers to have information fast, monitoring the team in real-time, reducing risks, and allowing several analyses by managers. We believe that the above findings have practical implications to project managers. Given the need for progress monitoring, it was possible to observe that *Scrum Watch* allows project managers to monitor project status in real time, as well as the team's history, providing a more assertive decision making regarding the teams. Moreover, our approach might provide project managers with special skills to effectively satisfy the analysis of such data.

To sum up, some issues should be tackled before using *Scrum Watch*. Firstly, project managers should consider that tools for code versioning, assets maintenance, and issue tracking are available and correctly used. Secondly, *Scrum Watch* should be set up and connected with those tools. Finally, project managers should receive some training regarding Scrum and software metrics, and then guidance on the use of *Scrum Watch*.

8 Threats to Validity

Despite a careful execution of the field study, there are threats that might compromise the results. Regarding construct validity, we found a significant absence of information in the input data sources. For example, the Sonar data source did not show any way of relating a developer to the code analyzed by the tool, even storing a diversified amount of information. In an ideal context, it would have been very useful the inclusion of the performance analysis of a teammate, the amount of code that it generated, and its quality.

Regarding internal validity, the results obtained might be biased. Although most data sources were able to relate the teammate to their documentation and tasks they worked on, there was also an important lack of data, e.g., when relating a Jira project to its repository in Git, since the most of the tables were empty.

When it comes to external validity, as Amazon's free tier services were used. However, some functional ones such as sending reports via email could not be added. The same happens with the functionality that QuickSight provides on machine learning. Secondly, the current development of our approach fails to include metrics regarding the mood of a team or the degree of satisfaction of a teammate. It would have been important to have that information and make it available in an additional dashboard. Finally, to generalize the findings, *Scrum Watch* seems to apply to other studies if both project managers receive training in Scrum, and tools to support software development provide accessible data sources.

9 Conclusion

This research work introduced *Scrum Watch*, a tool that aims to generate graphic elements and reports that allow project managers to make decisions, as well as to detect risk at an early stage, based on the use of ETLs. Since the process to move data from multiple data sources to a data warehouse was created through an ETL (and not from manual work), the following advantages were found: low cleaning time and data transformation; low storage since only the necessary information will be saved; consolidation of data in a single repository; and ensuring effectiveness and quality. Likewise, the integrity of the source data sources is ensured within this process. Since *Scrum Watch* is supported by Amazon cloud services, infrastructure knowledge is no longer required to support it.

Regarding data visualization, *Scrum Watch* provides a powerful, fast tool with a very wide range of visualizations. The speed it provides is given by the in-memory processing it performs. This is an advantage provided by few visualization tools. In terms of visualization, from tables to stacked or accumulated bar graphs, heat maps, line graphs, and KPIs are options offered by this tool to visualize the desired metrics. In addition to the benefits of visualization, it offers an important portability since new data sets can be added to the analysis easily.

As future work, it is proposed to adjust the Sonar's data collection process, given that it limited the join and analysis of the relationship between a team member and the source code he/she develop. Vital metrics to measure a member's technical performance were totally discarded from this analysis. In relation to the Jira data source, work should be done on the completeness of the information associated with the Git repositories where team members store and version their work. Metrics on the amount of pull request, merge, and commits could be added to a new dashboard in order to provide a more complete analysis of the performance of a team member. Regarding the ETL process carried out in this study and since the transformations were executed using SQL queries and manually, an automation process would be ideal to streamline the process.

Finally, and in order to measure the mood of the work teams, we propose the collection of information on the satisfaction and degree of trust of members that form teams based on surveys. Once this information is collected, it could be released in a repository or file that serves as a fourth source data source for the ETL process presented in *Scrum Watch*. Likewise, a new dashboard could be created to visualize the metrics not only to achieve continuous improvement in the final product, but also to ensure the quality of the teams that work on it.

Acknowledgements

We acknowledge the financial support provided by ANPCyT (Argentina) under Project PICT 2018-01456 and by CONICET (Argentina) under a doctoral grant and PIP 2021-2023 ID 11220200100430CO. The fourth author thanks to UNIRIO and FAPERJ (Proc. 211.583/2019) for partial support.

References

[Alaidaros et al. 2018] Alaidaros, H., Omar, M., Romli, R. (2018). Towards an improved software project monitoring task model of Agile Kanban method. *International Journal of Supply Chain Management (IJSCM)*, 7(3), 118-125.

- [Albrecht and Nauman 2008] Albrecht, A., Naumann, F. (2008). Managing ETL Processes. *NTII*, 8(2008), 12-15.
- [Beck et al. 2001] Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... Thomas, D. (2001). Manifesto for agile software development.
- [Boscarioli et al. 2017] Boscarioli, C., DE ARAUJO, R. M., MACIEL, R. S. P. (2017). I GranDSI-BR Grand Research Challenges in Information Systems in Brazil 2016-2026.
- [Dingsøy et al. 2012] Dingsøy, T., Nerur, S., Balijepally, V., Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development.
- [El-Sappagh et al. 2011] El-Sappagh, S. H. A., Hendawi, A. M. A., El Bastawissy, A. H. (2011). A proposed model for data warehouse ETL processes. *Journal of King Saud University-Computer and Information Sciences*, 23(2), 91-104.
- [Firdaus et al. 2019] Firdaus, M. B., Patulak, I. M., Tejawati, A., Bryantama, A., Putra, G. M., Pakpahan, H. S. (2019, October). Agile-scrum software development monitoring system. In 2019 International Conference on Electrical, Electronics and Information Engineering (ICEEIE) (Vol. 6, pp. 288-293). IEEE.
- [Gibson et al. 2012] Gibson, J., Rondeau, R., Eveleigh, D., Tan, Q. (2012, November). Benefits and challenges of three cloud computing service models. In 2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN) (pp. 198-205). IEEE.
- [Haidabrus et al. 2021] Haidabrus, B., Grabis, J., Protsenko, S. (2021, June). Agile Project Management Based on Data Analysis for Information Management Systems. In *Design, Simulation, Manufacturing: The Innovation Exchange* (pp. 174-182). Springer, Cham.
- [Highsmith 2009] Highsmith, J. (2009). *Agile project management: creating innovative products*. Pearson education.
- [Linden 2018] Linden, T. (2018). Scrum-based learning environment: Fostering self-regulated learning. *Journal of Information Systems Education*, 29(2), 65-74.
- [Mahnic 2011] Mahnic, V. (2011). A capstone course on agile software development using scrum. *IEEE Transactions on Education*, 55(1), 99-106.
- [Mathiassen and Pries-Heje 2006] Mathiassen, L., Pries-Heje, J. (2006). Business agility and diffusion of information technology.
- [Nerur and Balijepally 2007] Nerur, S., Balijepally, V. (2007). Theoretical reflections on agile development methodologies. *Communications of the ACM*, 50(3), 79-83.
- [Papke-Shields et al. 2020] Papke-Shields, K. E., Beise, C., Quan, J. (2010). Do project managers practice what they preach, and does it matter to project success?. *International journal of project management*, 28(7), 650-662.
- [Rodríguez et al. 2016] Rodríguez, G., Soria, Á., Campo, M. (2016). Measuring the impact of agile coaching on students' performance. *IEEE Transactions on Education*, 59(3), 202-209.
- [Version One 2020] 14th Annual State of Agile Report. [urlhttp://stateofagile.versionone.com/](http://stateofagile.versionone.com/)
- [Safvati et al. 2017] Safvati, M. A., Sharzehei, M., Mesbahi, M. R. (2017, December). Investigating the features of research environments on cloud computing. In 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI) (pp. 0404-0411). IEEE.
- [Schwaber and Beedle 2002] Schwaber, K., Beedle, M. (2002). *Agile software development with Scrum* (Vol. 1). Upper Saddle River: Prentice Hall.
- [Schwaber and Beedle 2002] Schwaber, K., Beedle, M. (2002). *Agile software development with Scrum* (Vol. 1). Upper Saddle River: Prentice Hall.
- [Scott et al. 2017] Scott, E., Pfahl, D. (2017, November). Exploring the individual project progress of scrum software developers. In *International Conference on Product-Focused Software Process Improvement* (pp. 341-348). Springer, Cham.

[Sidi et al. 2016] Sidi, E., El Merouani, M., Amin, E., Abdelouarit, A. (2016). Star schema advantages on data warehouse: using bitmap index and partitioned fact tables. *International Journal of Computer Applications*, 134(13), 11-13.

[Yildirim et al. 2019] Yildirim, N., Ersöz, S., Altun, B. (2019). Measuring Software Development Project Performance: A Case Study on Agile KPI's for Software Start-Ups. In *Agile Approaches for Successfully Managing and Executing Projects in the Fourth Industrial Revolution* (pp. 315-345). IGI Global.