An approach for eliciting requirements from digital sources in organisations using the Scrum method

Stylianos Georgiadis[1,2], Jelena Zdravkovic[2] and Janis Stirna[2]

[1] *European Dynamics, 15125 Maroussi, Athens, Greece*
[2] *Stockholm University, Kista, SE-16 407, Sweden*

Abstract

The business world is nowadays characterized by complexity due to rapidly evolving market and customer requirements. As a consequence, software providers are facing the challenge of delivering products with higher pace and innovation. The agile methodology has a big impact on how software systems are developed - it should facilitate business value in short iterations. Requirements are the base of all software systems, and consequently, Requirements Engineering (RE) plays one of the most important roles in system development. Traditional elicitation techniques relying on stakeholders' requests do not cover the increasing demands for considering unintended data from organisations' related digital sources, internal (transaction logs, sensors) or external (e.g., microblogs), amplifying thus the need for the elicitation of data-driven requirements. This study proposes a process that combines data-driven and traditional RE approaches for Agile software development, and specifically for the Scrum method. The process intends to assist Agile professionals to elicit requirements from digital sources in combination with intended data derived from the stakeholders without impacting the main Agile practices. The motivation for the research origins from the case studies carried in few companies having the challenge to include data-driven requirements into their Agile approaches. The usage of the proposal is illustrated on an enterprise software case, while several Scrum professionals were interviewed to evaluate its correctness and importance.

Keywords

Enterprise Agile Frameworks, Data-Driven Requirements Engineering, Scrum, User Stories

1. Introduction

Today's dynamic business environment requires flexibility for organizations to endure and evolve. The agile methodology is increasingly considered for enterprise system development to satisfy dynamic and demanding customers' needs and thus remain competitive in the market share [1]. Agile methods, such as wide-spread Scrum [2, 3], are highly iterative and incremental, and where the development team works in a close collaboration with the customer [4, 5].

Agile methods argue that system requirements evolve so rapidly that the focus must be set on the implementation as soon a change is requested. Pohl argued that requirements elicitation is the main activity of RE, where its first sub-activity concerns identifying relevant sources for eliciting requirements within a system's context [6]. If the relevant sources are not identified properly, the requirements specification for the system becomes incomplete.

In the traditional elicitation approaches, requirements are derived from human stakeholders as the main source. In agile methods like Scrum, the information related to the system to be developed is collected during interviews between the agile team and system's stakeholders, and then requirements, i.e., *user stories*, are created [3].

The requests for system's changes are intensely increasing due to a vast emergence of digital data sources, as well as due to the ability of the users to give online feedback within hours or even minutes.

Companion Proceedings of the 16th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling and the 13th Enterprise Design and Engineering Working Conference, November 28 – December 1, 2023, Vienna, Austria
stylianos.georgiadis@eurodyn.com (S. Georgiadis); jelenaz@dsv.su.se (J. Zdravkovic); js@dsv.su.se (J. Stirna)
0000-0002-0870-0330 (J. Zdravkovic), 0000-0002-3669-832X (J. Stirna)

Owing to this phenomenon, the interest in considering digital data as new sources for requirements acquisition in addition to traditional - stakeholder-driven, has significantly increased. Data-Driven Requirements Engineering (DDRE) has become an emerging sub-discipline where the requirements are gathered from vast digital sources such as microblogs, reviews, electronic documents, or sensor-readings and computer logs, at the same time enabling increased automation in requirements elicitation [7, 8, 9]. Dynamic data offers new opportunities to leverage a broader range of user requirements. There is hence the interest for the development methods for managing big and rapidly growing volumes of data to contribute to a continuous evolution of enterprise software systems. Focusing on data from digital sources enables the elicitation of up-to-date user requirements, which in turn improves customer satisfaction. Even the inclusion of massive amounts of digital data bears challenges for all the actors involved in a software development project, it also strengthens the interaction between them - analysts, developers, stakeholders, and especially end-users.

The data from digital sources most often is not intended for the purpose of requirements elicitation; therefore, it lacks structure and completeness. A number of studies have addressed this issue [9], by proposing the methods to link heterogeneous digital data to some initial requirements artefacts. However, there is still a lack of knowledge of how these efforts can become a part of and assist to the agile methods in use and their practitioners, especially since the actors related to agile methods are not able to interact with the authors of the proposed requirements, nor they can control the pace of the incoming digital data. Because the Scrum method has become wide-spread in agile development, and in addition - the first author of this study is a Scrum practitioner, the problem concern has been set to the lack of an effective process for managing data-driven requirements in a Scrum environment.

The purpose of this study has been to provide a contribution to the above-described challenges and the problem; therefore, the following research goal has been explicated - *How data-driven requirements engineering can be used in a development project following the Scrum agile method in parallel with stakeholder-driven requirements techniques?*

We have aimed to, using Design Science Research [10], design an applicable process for combining the management of data-driven and stakeholder requirements in a Scrum agile environment. The suggested solution intends to assist Agile professionals to elicit requirements from digital sources in combination with intended data derived from human stakeholders without impacting the main Agile practices; i.e. it should be an aid to the roles of the Scrum Master, the Product Owner, and the Development Team, in acquiring user stories from digital sources and transferring them to the development, while cooperating with existing stakeholders and processing the data with a higher degree of automation. The proposal is evaluated by several agile experts and demonstrated on a real-life case, but it has not yet been fully implemented in an agile project.

The remainder of the paper is structured as follows: in section 2, we present the background for this this study and related works. Section 3 presents our main artifact – an integrated Scrum process for dealing with both stakeholder and data-driven requirements. In section 4 we present a summary of the artifact's evaluation. Section 5 demonstrates the artifact on a real business case. A discussion of the results, concluding remarks, and future work, are presented in section 6.

2. Background and related work

2.1. Scrum method

The Scrum method is based on the pillars of transparency, inspection, and adaptation [3]. Throughout an emergent development process, the work and included tasks should be visible to everyone related to the project. Inspection is essential to detect deviations or undesirable variances, while the product must always be adjusted to minimize any deviations and maximize the product's value.

Scrum structures the development into work cycles (i.e., iterations) known as *sprints*. A sprint is a dedicated period of time in which a set amount of work is to be completed (2-4 weeks), with a strict start and strict end-dates.

The method recognizes four main roles participating in the system development: *Stakeholder, Product Owner*, *Scrum Master*, and the *Development Team* [11]. Stakeholder collaborates closely with the other roles throughout the entire project, starting from providing requirements for the system. Product Owner owns the system under development and is responsible for making all the decisions affecting that progress including the creation of the product (system) roadmap, identification of requirements, development, incremental improvements, and maintenance. Scrum Master is responsible for upholding the Scrum method, i.e., that the Scrum process is correctly and efficiently followed, that its tasks are feasible, that the project participants are focused on the project goal, and that potential obstacles in the team and the process are resolved. The Development Team is responsible for transforming obtained requirements into a working product (system).

In Scrum, there are three major artefacts: the *product backlog*, the *sprint backlog*, and the *product increment*. The first contains all the requirements that need to be implemented by the Development Team. The main responsible for the product backlog is the Product Owner and in collaboration with the rest of the Scrum team this role makes sure that the correct requirements (often referred as *items*) are developed and implemented [11]. The sprint backlog collects the items from the product backlog that are forecasted to be completed in an ongoing sprint. It is planned by Scrum Master and for the Development Team, to achieve an actionable plan for delivering the product increment [3] – i.e., the outcome which can be delivered to the customer without any additional work.

The requirements are in Scrum called *user stories*. According to [1], a user story describes the functionality that provides a value to either an end-user or a system's owner and consists of *a card* containing the brief story description created by Stakeholder (i.e., the main requirement statement); *a conversatio*n contains the details discussed through the process between the Stakeholder and the Development Team; *confirmation* contains the conditions to be tested by the team in order to verify that the user story is developed as expected. Every user story card follows the template *'As a <role>, I want <goal>, [so that <benefit>]'* and it is the most important requirements elicitation representation practice in the Agile software development process [12]. All elicited user stories are grouped in the product backlog and further through the sprint backlog assigned to the development team for a next sprint based on the priorities of the project.

The development process in Scrum is steering a number of relevant tasks (aka *actions*), where the major ones include: the start, when a Stakeholder decides to create or upgrade a system by creating a user story; next actions are to, in the collaboration with the Scrum team, *review* the user story for completeness, *rank* its priority, *illustrate* it (if needed), if too big - *split it into smaller user stories*, and finally – place the user story in the product backlog; further is the user story *planned* for the development, i.e., it enters the sprint backlog; from there the Development Team is fetching it into a sprint, developing, and then demonstrating to the Stakeholder for, upon feedback, placing it to a next sprint or delivering an executable version for implementation (i.e., product increment).

In addition to the presented concepts of the Scrum method, collaboration boards are commonly used to, by a visualisation tool (digital, or a physical whiteboard) help teammates understand how the user stories are advancing in their development and thus facilitate communication and operative decision-making [13].

2.2. Data-driven requirements elicitation

Interviews, focus groups, and workshops are the main sources of conventional RE [6]. Recently, organizations have been collecting user feedback through digital sources such as social media, user forums, or even review systems [7]. Software products' success likely depends on user feedback by providing high rates or positive comments. On the other hand, negative comments and low ratings may affect the sales numbers and the product's reputation. Data-Driven Requirements Engineering

(DDRE) takes advantage of a large amount of data retrieved either from user feedback in the form of natural language or machine-generated sources [8, 9].

[9] considered emerging dynamic data sources as possible sources of requirements and categorized them into one or a combination of human-sourced data sources, process-mediated data sources, and machine-generated data sources. Human-sourced data sources refer to digitized records of human experiences. Some examples of human-sourced data sources include social media, blogs, and content from mobile phones. Process-mediated data sources are the records of business processes and business events that are monitored, such as electronic health records, commercial transactions, banking records, and credit card payments. Machine-generated data sources are the records of fixed and mobile sensors and machines that are used to measure events and situations in the physical world. They include, for example, readings from environmental and barometric pressure sensors, outputs of medical devices, satellite image data, and location data such as RFID chip readings and GPS outputs, or data from computer systems such as log files [14].

Even if the data was not initially intended for use in RE, it still provides some essential information from which important change requests can be derived [15]. In [16], the authors emphasised the volume, velocity, and variety of Big Data as the main influential factor for scaling requirements management; they designed and developed a model-based semi-automated process to elicit candidate requirements from digital data sources.

[17] stated in their study that the agile organisations that have relevant heterogeneous data sources could benefit from having the semi-automated elicited requirements directly to their backlogs. Further in [18], the authors argue that the volume, dynamics, and variety of digital data cause the elicitation of requirements to become even more iterative and towards continuous, but also complex and unstructured, which current agile methods are therefore unable to manage in a structure and efficient manner. In our study, we have focused the effort to contribute to this challenge by proposing a synergy of stakeholder and data-driven requirements elicitation and development in the scope of the Scrum process.

3. Integration of data-driven requirements to Scrum method

3.1. Research approach

Methodologically, the study follows Design Science Research [10], an approach fostering incremental and iterative development of research artifact development in the IS domain, following the main steps that guide problem identification, design, development, demonstration and evaluation.

This study is a part of a larger DSR project which started by carrying out two case studies, one in the company involved in game development and the other in online banking business, to identify the problems concerning methodological support for data-driven requirements elicitation [18]. The two companies have in common that their services rely highly on the preferences of customers, whose number is up to *several million*; they showed therefore a high interest for integration of digital data (forum blogs, social media, transaction and user logs, etc.) for elicitation of requirements and for integrating these into their agile development approach. During observations and interview sessions, in a summary, both companies (i.e., Product Owners, Developer Teams, CTO), provided some important insights concerning a lack of a structured method for supporting the company's needs to optimise development resources when dealing with the requirements originating from massive online sources, removal of individual biases for requirements prioritization, and for enabling more rapid system releases.

In this study we have continued the research by taking the focus on a structured proposal for integrating DDRE in the Scrum agile method. We have designed an artifact that integrates elicitation of stakeholder and data-driven requirements into a single Scrum development process (section 3.2). Further, we did a semi-structured evaluation (section 4) and performed a demonstration with two illustrative cases (section 5).

3.2. An integrated process for data-driven and stakeholder requirements

The design of the process started by sketching the main tasks (i.e., actions, according to Scrum) for representing the workflow for the stakeholder-driven requirements elicitation in the Scrum environment. Then, the process was expanded to include and combine the actions relevant to the management of both data-driven and stakeholder-driven requirements, depicting also the Scrum roles interacting in each action, and the related Scrum artifacts (Figure 1, below). This effort was further complemented by the design of a corresponding collaboration board for presenting the progression of the user stories in each of the actions of the integrated process (Figure 2, below).

In the traditional Scrum method, the main roles have the responsibilities as they are described in section 2.1. However, when user stories are derived from digital data sources in addition to stakeholders, the responsibilities of the roles are changed, extended. The unintended data from digital sources are constantly updated. The Scrum team needs to know how to handle potentially a huge amount of new relevant information and how often this information should be taken into consideration as a candidate item to the product backlog. On one hand, user stories from unintended data reduce stakeholders' workload but on the other hand, may cause noise or even deviation from the project goal. It is further important to keep the sprint scope adjustable in a way that will not affect the workload of the development team and the quality of the final software product. Another important challenge is the review and adjustment of the user stories derived from digital sources. To distinguish, review, combine, or exclude potential user stories is a complicated task to be executed in such a short time. The table below summarizes the restructured roles' responsibilities:

Table 1
The responsibility of the main Scrum roles in the integrated requirements elicitation

Role	Responsibility
Stakeholder	Remains the author of his/her user stories. However, user stories derived from digital sources require analysis, assessment, and completion since most of them are ambiguous, vague, and incomplete. Because in most cases the author of a comment on online forum is unreachable, the Stakeholder must complete the user story, help in assessing the risk of the respective user story, help in identifying possible relations to some other user stories in the product backlog, and provide clarifications to the Development Team whenever needed.
Scrum Master	As the user stories are retrieved both from the digital stories and stakeholders, the complexity and magnitude of the Scrum practices and actions is increased. To ensure that all Scrum events are kept within the timebox, Scrum Master needs to provide concise and clear product backlog items and facilitate stakeholders' collaborations as requested. The user stories from post forums or tweets will be continuously created, therefore, he/she should remove the barriers and keep the pace for the whole Scrum team without adding extra concerns or work to the other members.
Product Owner	The role is responsible for the product backlog and for ordering the items within it. With user stories from digital sources, the product backlog is continuously increasing with more and more user story items. The new items must be communicated to the stakeholders, especially since they are not created or requested by them. The Product Owner is accountable for the management of an effective product backlog; and the communication with the stakeholders is essential to achieve the product goal.
Development Team	The team is enriched with *data analyst* responsible for developing and maintaining processing of raw digital data to candidate user stories. The team stays responsible for creating the sprint backlog, now including also data-driven user stories, hence developers review the provided input and assess the potential impact on the existing architecture without affecting the agreed plan, to achieve each sprint goal. When developers need clarifications about the data-driven user stories, stakeholders reply to them as the author of unintended data cannot be tracked.

Consequently to the changed responsibilities of the Scrum roles, the elicitation of the user stories from digital sources impacts the Scrum activities. The process model in Figure 1 intends to provide the information on the key tasks regarding the creation of user stories, their assessment as well as the implementation during the sprints. The process is described from the perspective of the main Scrum aspects: actions, artifacts, and actors. *The white-coloured symbols* in Figure 1 represent the elements that are done the same as in the traditional Scrum approach (section 2.1); *the purple colour*

represents the newly added elements for supporting data-driven requirements, while *the yellow-coloured* elements depict the changed traditional Scrum elements due to extending the process to the elicitation of data-driven requirements.

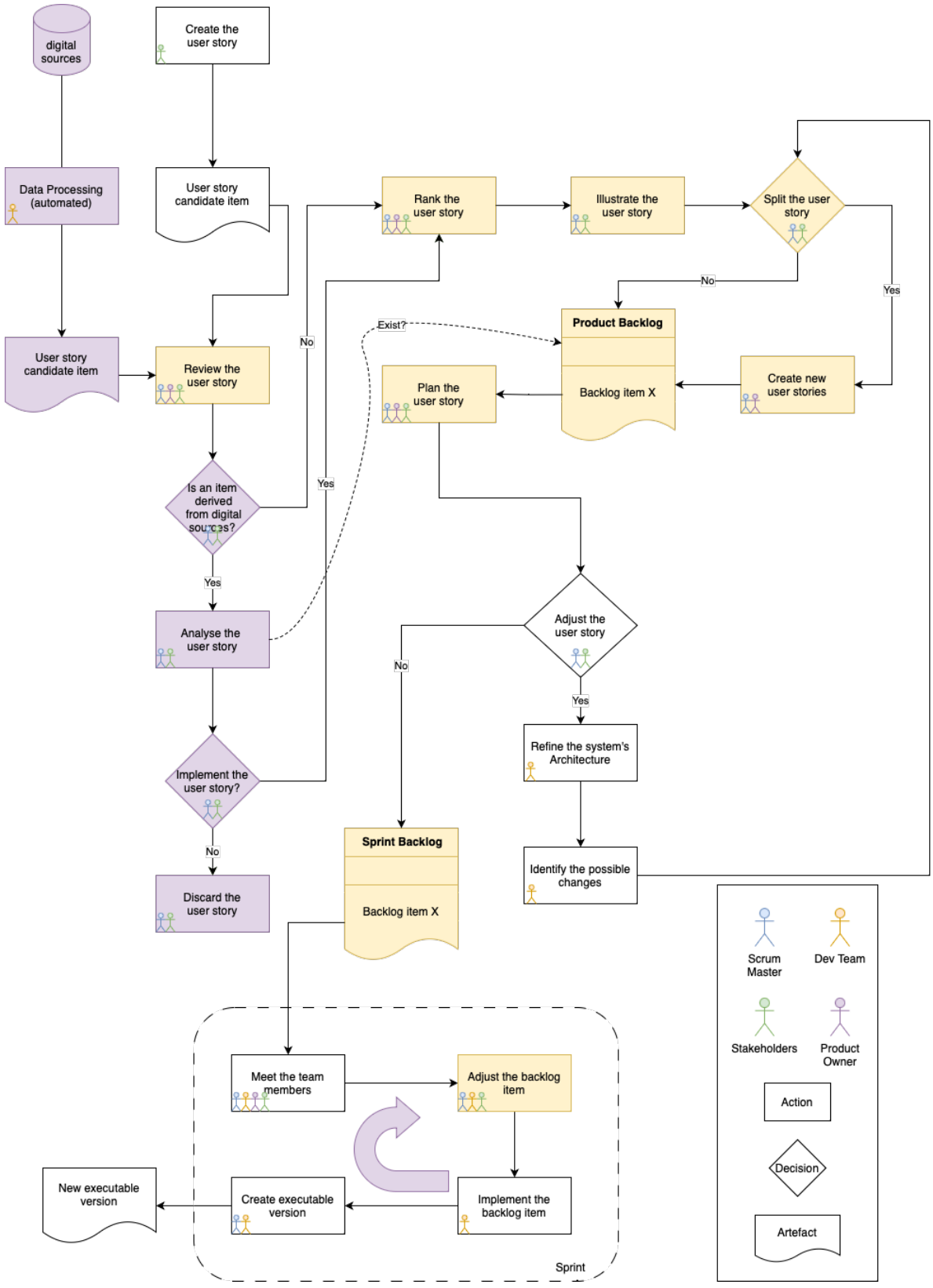

Figure 1: Integrated Scrum process for the elicitation of user stories from digital sources, in addition to stakeholders' user stories

The process beginning is determined by the incoming source type of the information. In case the source is a stakeholder, the process starts with and initial *user story created* by him/her as the main author; or, when some information arrives from a digital source, first *data processing* action is executed, following the approach presented in [16]; as a result, a *candidate user story* is obtained – containing a partial requirement description and a possibly priority (i.e., when applicable - from influence-related data). For event-driven sources, e.g., microblogs, computer logs, sensor data, the data may be collected continuously, either in near real-time or in batches at defined time intervals, from a location, a timestamp is set, and the status that it is fetched [16]. For human-sourced data (Section 2.2), which is mostly unstructured, Natural Language Processing (NLP) is required to extract relevant information. The analytical tasks for this source include: Classification, Sentiment Analysis, and Named Entity Recognition (NER). The outputs of these tasks are associated with a Segment, which allows for the body of the NL data to be divided into smaller units, such as sentences. Each processing step within the action *data processing* is associated with an algorithm or a ML model that was used to achieve a particular data transformation. The action can often be fully automated, however Data Analyst (from Development Team) role is responsible for developing or applying needed algorithms and training of ML models, as well as for monitoring the execution of the processing (details are elaborated in [16]).

Once a *user story candidate item* is obtained, Product Owner leads the *reviews* (section 2.1), but specifically for the data-driven user stories additional *analysis* is needed by Scrum Master and Stakeholders because that these items are often ambiguous and incomplete. If the *analyse user story* shows that it is not feasible to proceed with it (for quality, technical, or other reasons), it is *discarded*. Otherwise, the user story will proceed as Stakeholder's user stories to be *ranked, illustrated,* and *split into smaller user stories* if needed. During the analysis it is also important to compare the user story with existing backlog items, for possible similarity, because the data-driven items may be processed in high pace and amounts, compared to those of stakeholders.

The figure below depicts the changed management the user stories from the perspective of the Collaboration Board (section 2.1): *the yellow-coloured* elements depict stakeholder user-stories, and the *purple-coloured* data-driven ones. After the *Analysing* activity, all the user stories either derived from digital sources or the stakeholders are proposed to be treated the same (*grey-depicted*).

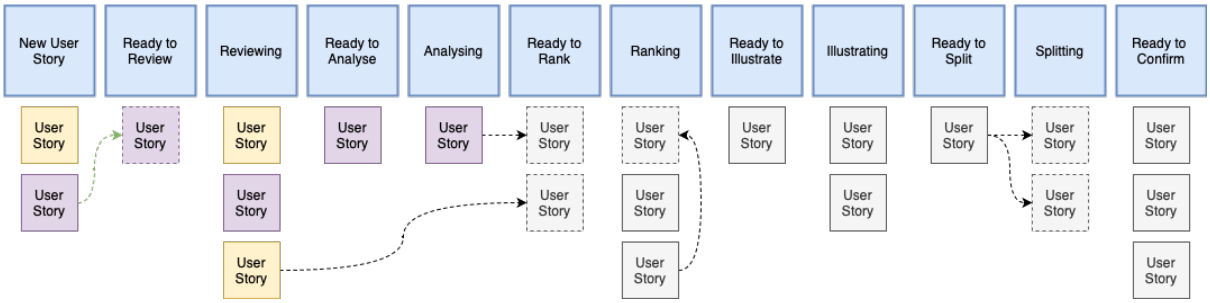

Figure 2: Collaboration board depicting the flow of stakeholder and data-driven user stories

Before the beginning of each sprint, as in the original Scrum process, the Product Owner, the Stakeholders, and the Scrum Master decide which product backlog items should be included in a sprint. During this phase, the Development Team identifies any changes necessary to complete the implementation of the sprint backlog items as well as refines the system architecture to support the new user stories (recall even section 2.1). After the completion of this second phase, the iterative cycle of development work starts – i.e., sprints are executed. A number of meetings, some of which are daily stand-ups, occur during a sprint to review and discuss whether there are any new requirements to be added to the sprint backlog, which becomes critical now when the list of possible items includes even the requirements derived from digital sources. The development Team gathers and consolidates the information retrieved from the meetings (i.e. *adjust*) either with the Scrum Master or with the

Stakeholders (final clarifications may be required, especially for the user stories from unintended sources).

Apart from the extension of the roles' responsibilities and main activities, the artefacts themselves are impacted by the inclusion of digital sources of requirements. The management of both product backlog, even sprint backlog, will be intensified, and the size will be enlarged due to the additional requirements from digital sources. Further, these user stories may not be feasible to be implemented within the period of one sprint because, as not being framed into the user story format from the beginning, they may depict broad expectations for improvements (i.e., epics) and therefore often needed to be split affecting the two backlogs by even more items, which in addition requires an increased effort for the assessment on possible similarities and dependencies between the items.

4. Evaluation

We conducted three semi-structured interviews with a set of the questions prepared in advance. The participants were Scrum Masters working in software houses for more than 2-3 years. They were asked about their concerns and arguments of the proposed process, the roles' responsibilities, and about efficient backlog and user story management for facilitating a smooth inclusion of the data-driven requirements elicitation, in parallel with the stakeholder-driven. The interviews lasted around three hours. The participants were fully familiar with the Scrum method and RE, while their knowledge about Data-Driven Requirement Engineering were little limited. For that reason, they were provided by some DDRE material in advance: As the main artifact for evaluation, they obtained our initial proposal for the integrated process (i.e., the non-evaluated content of section 3.1). In the beginning it has been reflected with the each of the participants that the proposed process i) should not decrease the quality of the developed product; that no changes shall be made that would severely endanger the project goal terms of cost, time; the product backlog should continue to be refined based on the respective needs of the users; the scope of the product shall be refined and negotiated with the Product Owner and the stakeholders as the plan progresses. A brief summary of the obtained feedback is presented in the list below:

- All three experts agreed that elicitation of the data-driven requirements should be integrated with the traditional (stakeholder-driven) process into a single process;
- The experts argued that integration should be done in the way to preserve the (agile) practices of Scrum as much as possible;
- The experts suggested that the data-driven user-stories should be, regardless their different source type, assessed in the process similar to stakeholder user stories wherever possible, to facilitate simplicity and transparency of the process;
- The experts helped substantially in detailing the Scrum actions, finalising their ordering, as well as the flow of the connections (Figure 1) as even the traditional Scrum process has been rarely published as a detailed workflow;
- The experts emphasised that depending on a Scrum project, i.e., the product to be developed, the need for data-driven requirements may vary; that is, based for example on the amount and quality of the available digital data, the expertise needed to process digital sources, automate data derivation, and other
- The experts contributed to the demonstration of the proposal (section 5) by giving the comments about the process for the two presented situations.

In conclusion, the experts acknowledged that DDRE has become an essential part for creating consumer-centric, user-friendly systems, and without defects. Stakeholders always have a different perspective from the end-user, according to these experts; therefore, collecting massive requirements from end-users is very useful for the adoption of the product. DDRE may play an important role in

the Scrum environment by reducing time and costs owing to increased automation of requirements processing, but the efficiency of handling and grouping all the user stories derived from digital sources is a crucial aspect of the process in order to avoid overloading the Scrum team and deviating from the product goal and time plan. One of the experts mentioned that DDRE should not be used during the initial sprints of a Scrum project where the system is not yet concretised but all of them agreed that DDRE will affect the elicitation process more and more as the project moves forward, while the stakeholders' workload will be reduced.

5. Demonstration

The Greek government has recently promoted a new type of financial support for citizens with specific social and financial status (low income, marital status, number of children, etc.). This governmental measure is called "pass". It is a type of voucher and can be used in associated companies such as supermarkets, gas stations, electronic stores, etc.

For this study, we focus on the "fuel pass" service (https://www.gov.gr/en/ipiresies/polites-kai-kathemerinoteta/metakineseis/fuelpass). A citizen applies for the "fuel pass" to receive two vouchers that can be used at specific gas stations. After the application is submitted, the system checks the type of the reported vehicle, the income of the applicant, and several parameters. If the application is approved, the citizen receives two digital vouchers valid for a certain time period. Otherwise, the application is declined, and the citizen is informed accordingly. The system provides also an online forum on which the users are able to post their feedback, complaints, or suggestions for improvements of the service (https://www.gov.gr/en/contact).

5.1. Case A

Table 2
User's post on the online forum

Body	I cannot upload picture file, please fix this before the application deadline!
Service	Forum
Sender	A registered user
Importance	High

Following the process from Figure 1, the obtained post is first processed using the DDRE approach developed in [16] that involves the use of NLP techniques and the supervision by the data analyst team member (recall section 3.2), and where as the output a *user story candidate item* is obtained – containing a partial requirement description:

Table 3
User story candidate item #1 derived from the online Forum using [16]

Body	*Role*	Registered User
	Functionality	Upload image file
	Benefit	N/A
	Conversation	before application deadline
	Confirmation	N/A
	Priority	High
Derived Card	As a registered user I want to upload image file	

After the creation of this candidate item (item #1), the Product Owner, the Scrum Master, and Stakeholders review the user story candidate. Since it is derived from digital sources and is incomplete, further analysis is required (Figure 3, left). This analysis is conducted with available human stakeholders since the author of the user story item is unknown and cannot provide further details. The user story item is completed by deciding on its benefit, and by deriving the conversation part defining exactly the types of attachments should be supported. Another aspect of the respective item could be the size or the number of attachments.

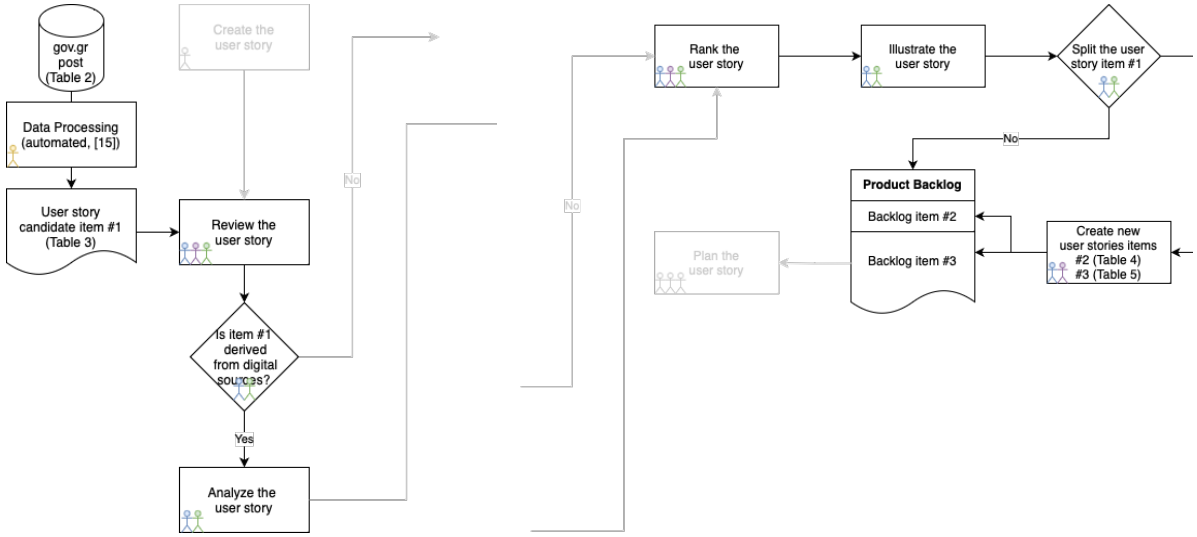

Figure 3: left) Review and analysis of user story #1 derived from digital sources; right) Ranking, illustrating, and splitting the user story item #1 to user story items #2 and #3

Apart from the analysis, ranking is required (Figure 3, right). Item #1 is marked with high priority, therefore, the stakeholders, the Scrum Master, and the Product Owner decide to which rank in the product backlog it will be moved. Illustration for this user story is not needed as the upload functionality is already implemented (just not for the image type of attachment). However, the team decides that the user story item will be split into two different user stories having better-limited scopes: one focusing on supporting additional file formats, and the other for enabling uploading more than one file. These new user story item #2 (Table 4) and item #3 (Table 5) are presented below:

Table 4
User story item #2

Card	As a registered user, I want to upload image file as attachment to be able to upload different types of files other than pdf	
Body	*Role*	Registered User
	Functionality	Upload .jpeg, .png, or .doc/.docx file as attachment
	Benefit	To upload different types of files other than pdf
	Conversation	The development team assesses how the attachments will be stored in the database or the server files.
	Confirmation	- The system shall allow users to upload .png, .jpeg, .png, .doc, .docx files. - The system shall not allow users to upload files with other extensions. - The system shall not allow users to upload multiple files. - The system shall allow users to not upload any extra attachments.
	Priority	High

Table 5
User story item #3

Card	As a registered user, I want to upload more than one file as attachment to submit the application	
Body	*Role*	Registered User
	Functionality	Upload more than one file as an attachment
	Benefit	To upload more files as extra attachments for the submission of the application.
	Conversation	- The development team shall assess the architect of the system to accept more than one extra attachment. - The development team with the Scrum Master shall discuss the maximum number of extra attachments.
	Confirmation	- The system shall allow users to upload only one extra attachment. - The system shall allow users to upload two attachments. - The system shall not allow users to upload more than 10 extra attachments.
	Priority	Medium

The user story cards #2 and #3 are added to the product backlog as candidate items. During the planning phase, the Product Owner and the stakeholders check the product backlog items and decide whether these items will be included in the next sprint. User story #2 (Table 4) is planned for the next sprint, while user story #3 (Table 5) with medium priority will be planned on another sprint. At this phase of the project, the Development Team also assesses whether the implementation of the planned product backlog item requires the refinement of the system architecture. In this case, the developers assess how the system will allow the user to upload new file types or where these files will be stored on the server, but no further adjustments are required (Figure 1). The Scrum Master adds the planned product backlog item to the sprint backlog and the implementation begins. If more clarifications are needed – for this case, how the user will upload the file or which message will be displayed in case of incorrect extension files, the Development Team discusses that with the Scrum Master. At the end of the sprint, a new executable version of the software is created, i.e., after the testing activities are completed and confirmed that the system is following the requirement. In this case, the developer checks that the user is allowed to upload an image file while he tries to submit his application for the "fuel pass".

5.2. Case B

In the current software version, there is already a product backlog item, #4, (Table 6), referring to the ability to delete an application if the due date has not been passed:

Table 6
User story item #4 (in the product backlog)

Card	As a registered user I want to delete my application if the due date has not been passed so that I can submit another application	
Body	*Role*	Registered User
	Functionality	Delete the user's record in the applications table
	Benefit	Make possible to submit a new application

	Conversation	- The development team assesses how the user deletes the latest submitted application when the due date has not passed.
	Confirmation	- The system shall allow users to delete the latest application if it has not been processed and the current date is equal to or less than the due date. - The system shall not allow users to delete the latest application if it has been processed. - The system shall not allow users to delete the latest application if the current date is greater than the due date.
	Priority	High

However, it has been processed that some users wrote on Forum the posts (translated to English): "why can't I remove my application?" or "it should be possible to delete the application after due date so that I use my support later"; this is because they have changed their mind and want to keep the right for the financial support for another occasion in the future. In Table 7 the user story candidate item #5 aggregated from several similar forum posts (using the approach in [16]) is presented:

Table 7
Candidate user story item #5 derived from the Forum using [16]

Body	*Role*	Registered User
	Functionality	Delete application after due date
	Benefit	use the support right
	Conversation	N/A
	Confirmation	N/A
	Priority	High
Derived Card	As a registered user I want to delete my application after due date	

As described in the example case A, after the creation of this candidate item, the Product Owner, the Scrum Master, and the available Stakeholders review the user story candidate item #5 (Table 7). Since it is derived from digital sources and is incomplete, the analysis is required. However, the analysis of item #5 is not executed to the same extent again since there is a related user story item #4 (Table 6) already placed in the product backlog. The Product Owner and the Scrum Master match the new user story #5 with the existing user story #4 enabling the user story card #5 to be updated and completed. Then, the Product Owner and the Scrum Master in collaboration with the Stakeholders rank the user story #5 and decide whether user story #5 will be added to the product backlog and implemented on a next sprint. They have decided to include it on the product backlog and therefore, user story #4 will be removed since it contradicts accepted user story #5. In the next sprint, the Product Owner, the Stakeholders, and the Scrum Master plan to add user story #5.

6. Discussion and conclusions

The emerging presence of DDRE in the software development and the need to therefore integrate it into the agile methodology, specifically in the Scrum method, has been the main motivation for this study. We have proposed a process that can combine stakeholder-driven and data-driven requirements in the Scrum environment in order to benefit from both requirements engineering approaches and to address integration issues in one of the most commonly used Agile methods. The challenge was not only to combine both requirements elicitation approaches but also to adjust them in the Scrum methodology with the least possible impact on the established Scrum practices. The

proposed integrated process is based on a workflow created by the authors to present how user stories are created, reviewed, planned, and implemented in Scrum when they are derived from stakeholders.

Using unintended data from digital sources in the requirements elicitation activity leads to increased customers satisfaction, more transparent decision-making operations, as well as the time and cost management can be improved owing to increased automation of the elicitation. These are some of the main reasons for the Product Owner to, with the rest of the Scrum team, decide to exploit the user stories derived from digital sources equally as those created by the stakeholders.

In contrast with the user stories derived only from the stakeholders, candidate items from digital sources require further analysis. Most of the time, they are ambiguous, vague, and incomplete. During the review of a candidate user story item, the stakeholders and the Scrum Master decide whether it will be added to the Product Backlog, declined, or split into smaller items. Sometimes it is possible that one candidate backlog item is related to an existing one. The relation with existing implementations shall be detected as soon as possible in order to decide if this item will be discarded or developed on a next sprint.

Furthermore, ranking is one of the important activities. Given the time plan and the product goal, a user story is higher or lower prioritised in comparison to the rest product backlog items by the Product Owner and the Stakeholders. Understanding correctly the importance of data-driven user stories based on the impact-related data from the online sources, and the magnitude of similar information, is a new responsibility in the integrated process.

Digital sources are constantly providing data and user stories to be added, therefore, the planning and review of them may cause overload in the Scrum team. A solution to this challenge is for the Product Owner and the Scrum Master to invite the stakeholders to provide advice, spot and relate the most important items, and discard the items which are less irrelevant to the product goal.

In addition to the practical experience of the first author in the domain of RE, agile methodology and its methods, and the corresponding scientific knowledge of both authors, the reliability of this study has been increased by the evaluation process because all of the interviewed Scrum experts have recognized the importance of integration of DDRE to the Scrum method, as well as they provided numerous useful comments and suggestions for improvements.

The study distinguishes itself from related publications as it provides a contribution with the perspective on an entire development method, in particular – Scrum. The proposal provides some relevant insights for scientific research and essential concepts for practical application.

At this point, consideration of the proposal of this study in agile projects is one goal, for the validation purpose and further learning; another our goal is more scientific and concerns increasing the efficiency of the proposed process by increased automation of the integrated Scrum activities such as different analysis, ranking, and structuring actions, for reducing and streamlining the responsibilities of the Scrum team.

References

[1] M. Cohn. User stories applied: For agile software development. Addison-Wesley signature series. Addison-Wesley (2004)

[2] K. Schwaber. Scrum development process. Oopsla'95 workshop on business object design and implementation. *Austin*, USA (1995)

[3] K. Schwaber & J. Sutherland, J. Scrum Guide | Scrum Guides. online] Scrumguides.org. (2020) Available at: https://scrumguides.org/scrum-guide.html

[4] V.N. Vithana. Scrum Requirements Engineering Practices and Challenges in Offshore Software Development. International Journal of Computer Applications (0975 – 8887), Volume 116, No. 22 (2015)

[5] J. Zdravkovic, J. Stirna, J. C. Kuhr, & Hasan Koç. Requirements Engineering for Capability Driven Development. In: The Practice of Enterprise Modeling. PoEM. Lecture Notes in Business Information Processing, vol 197. Springer, Berlin, Heidelberg (2014)
[6] K. Pohl. Requirements Engineering: Fundamentals, Principles, and Techniques. Springer, Heidelberg, New York (2010)
[7] W. Maalej, M. Nayebi, T. Johann & G. Ruhe. Toward Data-Driven Requirements Engineering. IEEE Software, 33(1), 48–54, (2016)
[8] C. Quer., X. Franch, C. Palomares, A. Falkner, A. Felfernig, D. Fucci, W. Maalej, J. Nerlich, M. Raatikainen, G. Schenner, M. Stettinger, & J. Tiihonen. Reconciling Practice and Rigor in Ontology-based Heterogeneous Information Systems Construction. In: Proc. of the Practice of Enterprise Modeling (PoEM), LNBIP vol.335, Springer, pp. 205-220, Springer (2018)
[9] S. Lim, A. Henriksson, & J. Zdravkovic. Data-Driven Requirements Elicitation: A Systematic Literature Review. Springer Nature Computer Science vol. 2/16 (2021)
[10] K. Peffers, T. Tuunanen, M. Rothenberger, & S. Chatterjee: A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems 24(3), pp. 45–77 (2007)
[11] D. Maximini. SCRUM CULTURE: introducing agile methods in organizations. S.L.: Springer (2019)
[12] X. Wang, L. Zhao, Y. Wang, & J. Sun. The Role of Requirements Engineering Practices in Agile Development: An Empirical Study. *Requirements Engineering*, [online] pp.195–209. doi: https://doi.org/10.1007/978-3-662-43610-3_15 (2014)
[13] M. Cardinal. Executable Specifications with Scrum. Addison-Wesley (2013)
[14] D. Firmani, M. Mecella, M. Scannapieco, & C. Batini. On the Meaningfulness of Big Data Quality. Data Science and Engineering, vol. 1(1), pp. 6–20 (2015)
[15] M. van Vliet, E.C. Groen, F. Dalpiaz & S. Brinkkemper. Identifying and Classifying User Requirements in Online Feedback via Crowdsourcing. Requirements Engineering: Foundation for Software Quality, REFSQ. Lecture Notes in Computer Science, vol 12045. Springer, pp 143-159 (2020)
[16] A. Henriksson & J. Zdravkovic. Holistic Data-Driven Requirements Elicitation in the Big Data Era. Software and Systems Modeling, Springer, vol. 21 pp. 1389–1410 (2021)
[17] M. Oriol, S. Martínez-Fernández, W. Behutiye, C. Farré, R. Kozik, P. Seppänen, A. M. Vollmer, P. Rodríguez, X. Franch, S. Aaramaa, A. Abhervé, M. Choraś, & J. Partanen. Data-driven and tool-supported elicitation of quality requirements in agile companies. Software Quality Journal (2020) doi: https://doi.org/10.1007/s11219-020-09509-y
[18] X. Franch, A. Henriksson, J. Ralyté, & J. Zdravkovic. Data-Driven Agile Requirements Elicitation through the Lenses of Situational Method Engineering. IEEE International Requirements Engineering Conference, IEEE Computer Society, pp 402-407 https://ieeexplore.ieee.org/document/9604733 (2021)