Received: 13 September 2020 | Revised: 20 May 2021 | Accepted: 2 July 2021

IET Software

DOI: 10.1049/sfw2.12037

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH PAPER

Managing non-functional requirements in agile software development

Scarlet Rahy | **Julian M. Bass**

School of Computing, Science & Engineering, University of Salford, Manchester, UK

Correspondence

Scarlet Rahy, School of Computing, Science & Engineering, University of Salford, Manchester, UK.
Email: S.Rahy@edu.salford.ac.uk

Abstract

Non-functional requirements define qualities of the software system that ensure effectiveness while embedding any constraints and restrictions on the design. A challenge rises with agile implementation in handling non-function requirements in regulated environments. Thus, a practitioner's perceptions of agile method tailoring are described in relation to inter-team boundaries and non-functional requirements. The research comprises 18 practitioner interviews from two multinational agile software development companies. Interviews were recorded, transcribed, and analysed using an approach informed by grounded theory and information flow models were used to compare and contrast interactions of processes. It was discovered that one of the case study companies managed non-functional requirements as artefacts in their agile methodology, while the other company reverts to conventional plan-based software development practices of documentation, timeline estimations, and safety critical requirements. This research creates a detailed comparison of these contrasting approaches. The main contribution of this study is a set of proposed recommendations to deal with non-functional requirements in a regulated environment using agile techniques. The introduction of two new artefacts, Documentation Work Item and Safety Critical Work Item, is recommended and it is accompanied with an illustrative example, to transform the handling of documentation and safety critical requirements in a more agile way.

KEYWORDS

software development management, software engineering

1 | INTRODUCTION

Software development has become an integral part of our current globalised world. Since the creation of the agile manifesto in 2001, agile software development has been growing vertically within organisations and horizontally across organisations in different business sectors. Initially, agile methods were viewed as best suited for small projects with co-located teams [1]. With the increasing spread of agile methods, large-scale organisations started implementing agile methods [2, 3]. This implementation was associated with agile methods tailoring to fit large-scale organisations [4, 5].

A set of agile principles was introduced to the software development industry in the Agile Manifesto. Agile methods enhance customer involvement in the development process by encouraging regular customer feedback and its prioritisation [6]. Conventionally, agile methods are comprised of ceremonies, artefacts, and roles.

Inter-team communication in agile has been identified as an important topic for researchers [7, 8] and refers to communication between teams in the same company. The right combination of communication mechanisms improves agile implementation while a mismatch may be an impediment [9]. Inter-dependent teams are highly reliant on team

This manuscript is submitted to IET Software and is not under consideration for publication anywhere else. Thank you for your consideration!

communication and networking behaviour, thus highlighting the importance of boundary spanners as the connection between heterogeneous groups [10]. Communication becomes harder in large-scale development projects because of employee turnover, formation of new teams, and the addition of new developers and technical experts, especially in regulated environments. Regulated environments are controlled by rules and regulations such as financial services, food, health, or nuclear sectors, generating additional non-functional requirements (NFR) [11]. Originally, agile researchers were leery of applying 'agile modelling to life-critical systems' [12]. With the progression of research, agile methods are seen as suitable when tailored to the needs of the regulated environment [11]. Thus, additional inter-team communication is required to handle the increasing number of NFR.

Also, addressing NFR in software development remains a crucial factor for the success of any development project since NFR play a critical role in the user's choice among alternative designs and ultimate implementation [13]. Given that the intensity of NFR increases in regulated environments, it becomes necessary to study how agile software development companies handle NFR and inter-team boundaries in regulated environments. Thus, we investigate practitioners' perception on tailoring of agile methods in relation to NFR and inter-team boundaries.

The study compares and contrasts two diverse companies. The first company, which we will call 'DevelopCo', is a software development company providing business solutions for companies and governmental organisations. The second company, which we will call 'HealthCo', is a software development company that manufactures and distributes medical instruments internationally for the healthcare industry.

The study shows how one company converts NFR into agile tasks using user stories, while the other company tailors agile methods in order to overcome the NFR and uses a combination of conventional waterfall and agile methods. The main contribution of this study is that it examines the difference in managing NFR and presents illustrative examples that demonstrate how documentation, regulatory compliance, and safety critical work items can be implemented in an agile software development process.

The remainder of this study is structured as follows: First, a literature review related to inter-team communication and boundaries, agile methods tailoring in regulated environments, and NFR is presented. Second, the research methodology used is explained based on a chosen qualitative research method. Third, the analysis and discussion of the case study results are presented. Finally, each research question is discussed, the limitations are presented, and directions for future research are suggested.

2 | RELATED WORKS

This section presents the literature review on agile methodology. This section then investigates inter-team boundaries in an agile environment, relations between regulated environments and agile tailoring, and NFR along with the approaches taken to handle them.

Authors of [14] argue that the quality attributes of software are expected to improve with the early addressing of NFR in agile software development. Previous studies have examined NFR in agile software development in addition to investigating different approaches to handling NFR in the agile context [15]. For instance, the Traceability Process Model tackles the issue of tracing NFR whenever the customer changes the functional requirements [16]. The literature has also studied the implementation of agile methods in regulated environments [17, 18]. Yet, the literature lacks information on how to handle NFR in an agile software development context and a regulated environment taking into consideration inter-team boundaries.

2.1 | Inter-team boundaries

Large-scale agile software development is based on collaborating software development teams working towards common strategic goals and inter-team coordination [7]. Human resource management is the most significant knowledge area in agile software development [19]. Inter-team coordination or dependency is one of the four main themes of concern in large-scale agile implementation [8]. Practitioners' ability to get along with team members positively affects the self-organising team climate [20].

In agile software engineering, teams work towards consensus-based decisions, with opinions from all team members being valued [6]. The nature of agile software development produces boundaries between different actors, such as inter-team, team and customer, and product owner and developers. This leads to the rise of boundary spanners as communication facilitators [3]. Boundary spanners act as coordinators who provide a source of information, a target for feedback [21], a mediator between different teams, and a socio-material assemblage [10], and help form organisational identities [22]. For instance, the product owner manages geographically separated teams by acting as a communicator, connecting geographically separated teams using multimedia technology, or a traveller, alternating between sites to build trust [23].

2.2 | Regulated environments and agile tailoring

Regulated environments are characterised by businesses where quality assurance, safety, security, or traceability are considered as key concepts that should comply with certain regulations, guidance, and official standards [11]. At first, the implementation of agile software development in a regulated environment did not comply with the initial principles of the agile manifesto. Several recent studies, however, suggest that the tailoring of agile methods to meet the needs of a regulated environment may improve the implementation process [11, 18, 24]. For instance, in a systematic literature review, [25] reveals four problem areas in a safety critical regulated environment: light

documentation, flexible requirements in user stories, iterative lifecycles, and test-first process. The literature lacks the discussion of regulated environments in relation to NFR and agile tailoring.

The tailoring of agile methods came after the failure of several large-scale agile software development projects [26]. Agile tailoring may include the selection of certain agile methods or integration of some aspects into the project [27]. For instance, one of the ways to tailor agile methods is to create both agile and plan-based artefacts to enhance the technological strategies and quality assurance methods of the organisation [5]. Also, agile roles are tailored and new teams might emerge; for example, the role of the product owner may be tailored [28]. The research looks at agile tailoring methods in different contexts but lacks a comprehensive inventory of the agile practices that are common in regulated environments.

2.3 | Non-functional requirements

Within the software development for regulated environments, the number of NFR increases [11]. With the increase in regulatory requirements imposed by agencies, the privacy, security, and safety related NFR increase. Software development requirements are divided into two parts: functional requirements and NFR. The latter represents the behaviour of the system and the factors that differentiate it from similar products, which include, but are not limited to, usability, maintainability, security, performance, reliability, scalability, and availability [15]. NFR analysis and incorporation in the software developed is critical and essential for its success [15]. Customers are not fully aware of all NFR in the initial stages of the project, instead they focus on fully developing functional requirements leading to challenges in terms of software scalability, decrease in software quality, and increase in maintenance cost [29]. Simultaneously, the agile methodology does not provide a clear approach for handling NFR [30]. Also, agile allows space for minimal documentation, which in turn causes traceability issues of NFR provided by the customer or regulatory agencies throughout the development. This leads to an increase in the amount of rework [15]. Researchers proposed a NFR modelling frameworks, NORMAP, with a simulation tool, NORMATIC, for a semi-automated process [31]. Also, NERV is another framework for elicitation, management, and validation of NFR. The benefits of such frameworks lie in automatic capturing of NFR, visualisation capabilities, and risk-driven requirements implementation sequence [32]. Yet, this is not always possible and with limited implementation, especially since NFR outside the text-based documents, such as images, were ignored [33].

There are several approaches used to analyse the NFR [34–37]. Research has also proposed enhanced versions of these approaches [15]. Yet, the literature still lacks in the identification and management of NFR in the agile software development context [38]. The gap in knowledge is even more profound on NFR in the context of cooperating software development teams. This topic is challenging due its cross-functional aspects and lack of clarity in this aspect in most parts of the projects. We aim to reduce the gap in the literature on the relation between agile tailoring, on one hand, and NFR and cross-border software development on the other.

3 | RESEARCH METHODOLOGY

Our research methodology investigates the agile software development implementation strategies at two different multinational companies. The study employs a qualitative research methodology, informed by a constructivist grounded theory approach [39], which allows us to address the complexity of human behaviour in the software development context [40] and advocates constructing a literature review [41]. This study is guided by two research questions:

1. How do the practitioners in this study explain their interactions and information flows on managing NFR?
2. How do practitioners in this study describe the tailoring of agile methods to handle NFR?

To understand the communication strategies and the differences in agile implementation methods, a case study was conducted comprising of practitioners from two multi-national software development companies, which are hereinafter called DevelopCo and HealthCo to protect their anonymity. Both companies develop software that may be subject to regulations. Our unit of analysis includes the first set of practitioners interviewed from DevlopCo and the second set of practitioners interviewed from HealthCo. In addition, this study uses information flow analysis to understand the product development process, highlighting the similarities and differences in practitioner perceptions between companies.

3.1 | Research sites

Data was obtained and analysed from two international companies providing services in software development. DevelopCo is an international company that uses agile to provide business solutions and create custom software. Its headquarters is in The Hague, Netherlands and has a branch in Nairobi, Kenya. The company has been operational since 2005. Clients are spread across Europe and Africa. It works on providing business strategies and solutions for a diverse set of clients. Such clients include government departments that have their own set of regulations and requirements. HealthCo is a medical technology company providing health services worldwide. The study conducted took place in Bangalore, India. HealthCo aims to improve patient healthcare by providing state-of-the-art medical technology. The products include medical imaging, laboratory diagnosis, healthcare IT and electronics.

Purposive sampling was used to select both companies. We chose HealthCo and DevelopCo under a purposive sampling strategy in our research [42]. Through purposive sampling, several qualitative research designs can be used along with

multiple research phases [43]. We followed a diverse case selection method since we intended to have variance in the regulated environment between the two chosen research sites [44].

3.2 | Data collection

The primary source of data collection was face-to-face interviews conducted either in physical presence or through Skype. 18 interviews were conducted with participants. Information on the 18 participants can be found in [45]. Also, a review of the publicly available documentation on the development practices was conducted. The subject's selection was done with the help of the company representatives. The interviews ranged from 45 to 60 min. The data collected was obtained from semi-structured open-ended questions. Probing questions were used to encourage participants to discuss any new topics that were not included in the interview guide. The interview guide and consent form used for interviews may be viewed at [46]. All interviews were recorded after obtaining the practitioners' consent. Field notes were taken during the interviews by the researchers. Then, the interviews were transcribed manually since it ensures correct transcription and reminds the interviewer of the social and emotional aspects that occurred during the interview [47].

3.3 | Data analysis

Within-case analysis and cross-case comparison are two main steps in the data analysis. The data analysis of the interviews for within-case analysis was done separately for each company. The data set from the DevelopCo's interviews was analysed first, then the data set from HealthCo's interviews was analysed subsequently. The transcripts were imported into the data analysis tool Nvivo 11 [48] with both authors participating in the data analysis procedure. In the cross-case comparison, the information flows of both companies were compared and contrasted, then the tailoring of agile methods was analysed in light of NFR. All the interviews were coded, leading to deriving categories, high levels of abstraction, and concepts and patterns of behaviour [49]. The categories were deduced from the transcribed interviews while trying to avoid introducing any preconceived ideas or thoughts.

Line-by-line open coding approach was used on the transcribed interviews. While coding line-by-line, data can be inspected and a special incident can be found in a word, a line, or through several lines [49]. This coding process was organised using the Nodes feature in Nvivo. Each code was given a title and a constant comparison method was used. This constant comparison technique was a key to identifying concepts that were then grouped into categories that were coded. The transcribed interviews were reviewed more than once and each time new categories emerged. This ensured that no data was left unnoticed. This process stopped when no new categories were created.

We were informed by Cruzes's approach for conducting a cross case analysis [50] while replacing the use of matrices with memo-ing. This approach is guided by three major steps: data reduction, data display, and conclusion drawing.

First, we performed data reduction, which involved the transformation of transcribed audio into memos containing the key concepts. Initially, each memo was written in draft form, to ensure that memos are written in the 'passion of the moment' [49]. The memos were then revised and written formally to guarantee correct understanding by using revised English. Also, constant comparison was applied to categories and the participants' responses from the two different geographical locations were compared. Thus, when a difference in opinion occurred, if any, it was indicated in the writing of the memos. Quotes from the interviews were used as evidence in the memo writing.

Second, we performed data display by constructing an information flow model for each company. Information flows within each company were analysed and modelled, using concepts in the memos, by creating a swim lane diagram. Input from other memos were also included using the constant comparison technique. After creating the model, memos were revisited, thus adding further data to the model. Such data was acquired when the questions asked by the interviewer were indirectly related to the information flow.

Third, we drew conclusions to identify the agile tailoring methods in relation to NFR. After the individual study of each organisation, the swim lane diagrams were overlaid to identify similarities and differences. These were then analysed considering NFR and inter-team boundaries.

4 | RESULTS

In this section, the findings from DevelopCo and HealthCo case studies are presented. We present the communication features, inter-team dependencies, communication challenges in multi-cultural organisations, and approach to NFR. We present a swim lane diagram that portrays the main activities in each section while highlighting the network of information flows and communication channels.

4.1 | DevelopCo inter-team dependencies

During a project, several sprints are interconnected as is the case between Team A, the design team, who feeds in the design for Team B, the development team. Unfortunately, this reliability sometimes may cause a problem, especially if the work done by Team A's sprint was not completed on time or needed rework. In such cases, not only will Team A have an incomplete task and need to use another sprint to complete the same task but also Team B will have idle time, where Team B will be waiting for Team A to finish the sprint before it can start its own.

This case has occurred several times between the design team and the development team. Since the development team

is highly dependent on the work of the design team and since the sprint of the design team is highly dependent on the customer's approval, the development team experienced several incidents where they had idle time. The designer described, 'If something goes wrong in my sprint then the developers do not have anything to do'.

On the other hand, the technical lead suggested that 'the delay can be caused by miscommunication'. The lack of communication between teams can cause the delay of sprints and thus create an idle time where a team or more will be waiting. This wait time would have been avoided if communication had occurred. Developer 2 said, 'So if they had told them [about the unfinished work] they could have done a new sprint planning and plan different stuff and finish other tasks'.

4.2 | DevelopCo communication at the boundaries

Each resource prefers a certain type of communication method. This along with other factors causes communication problems at the boundaries between teams, directors or product owners. Boundaries discussed below include the inter-team boundaries and the boundary between the organisation and the customer.

One of the easiest and clearest form of communication between teams, as stated by the practitioners, is using the Kanban board. The technical lead said, 'We use Trello [digital Kanban board] and this is where we set all the user stories for the teams….when you pass by you can see straight away what the status is of the sprint'. Developer 1 showed how communication between different teams can be done using Slack while benefiting from all its options. Developer 1 also highlighted the importance of demos as a briefing opportunity for all teams: 'You can actually get feedback from every member of the team since we are demoing to everyone. So, in the spirit of sprint, we are able to communicate with the whole team and also when at need'.

One of the main issues is that one team tends to assume that other teams know its status and is aware of any setbacks faced. Developer 2 said: 'Teams are like, we know so the rest knows it as well. They are just assuming'.

Another communication boundary rises between the organisation and the customer. We have previously recognised the problems that arise when agencies are involved. Another problem is the clarity of the customer's functional requirements and identification of NFR. Sometimes the customer's request is vague, incomplete, or subject to change. Then, the members might misinterpret what is needed or again make wrong assumptions, especially when they are new to the agile implementation process. The technical lead indicated: 'What the customer wanted was not clear and then the team assumed how to implement it. And then they implemented it and then in the demo they found out that is not what we need'.

4.3 | DevelopCo communication and customer requirements

Agile software development requires the customer's input to the project. Thus, understanding customer needs and building on their requests are of high importance. Lack of information and clarification may also lead to delay in delivery with respect to time. The public relations manager said, 'Also, the lack of information of the project we are doing is a factor that will negatively affect delivery time'. Sometimes the team members, or the scrum master, do not ask the correct questions and base the user stories and sprint planning only on the customer's briefing. Director 2 said, 'I think the most common negative effect on the workload is the lack of clarity and understanding in what is required'.

4.4 | DevelopCo's approach to NFR

DevelopCo works on maintaining an agile software development process. As Director 2 puts it, 'There is nothing I can think about in the company that is not agile'. Director 2 describes how being agile requires continuous learning and improvement:

> One of the things I learned about being agile is that you cannot be 100% agile; as time passes by, we realize how we can get better. We think we are 100% agile and then 2 weeks after, we realize we can make it even better.

DevelopCo has worked in highly regulated environments such as a project designated to develop a business strategy for a government department. The project was implemented in a highly regulated environment requiring documentation and milestones. Director 2 describes the project: 'We have a project with the German Government, which is completely non-agile. The project description, documentation and terminology are non-agile'. Such documentation requirements included the Project Initiation Plan, Stakeholder Impact, Regulation–Business Case Matrix, and the Issues Log.

In such cases, DevelopCo treats all NFR as user stories, incorporating them into the agile process. Director 2 explains, 'We just take an agile approach when dealing with NFR. We treat them as stories'. Thus, even when dealing with NFR, DevelopCo aims at fully implementing agile. By dedicating a user story for each NFR, the progress may also be made visible by showing them on the Kanban board.

DevelopCo has even abolished the manager role. DevelpCo limits the roles in its company to agile roles: scrum masters, product owners, and self-organising teams. In addition, when dealing with non-agile implementing companies, the company takes an agile approach by creating user stories to cope with the circumstances. Director 1 clarified, 'If your customer is non-agile, then the feature of the

product or project you are building just becomes another agile task'.

4.5 | DevelopCo process flow

This section presents the process flow for DevelopCo using the swim lane diagram shown in Figure 1, focussing on one iteration. This swim lane diagram was derived from the memos deduced from the transcribed data. An initial diagram was drawn from the memos that described the information flow. Then, constant comparison technique was used to complete the diagram by accumulating information from memos that are not directly related to the information flow.

The diagram uses four swim lanes to display the main actors in the agile software development. The model starts with the customer's input on product requirements and ends with the retrospective. Activities that are placed on the borders of the swim lanes indicate that the activity is performed by the actors sharing the border. As part of the agile process, the diagram shows how the customer is involved in the product development process. The customer was included in the model since the evaluation and completion of the product backlog is based on the customer's input along with the success of the sprint, which is based on the customer's evaluation of the team's demo. Also, it portrays how common the inter-team communication and the product owner involvement are in the process. Additionally, it displays the scrum master's role and highlights how the scrum master acts as a link between the different actors.

4.6 | HealthCo inter-team dependencies

When working on large-scale projects, inter-team dependencies become a natural process. In the healthcare sector, some dependencies occur in the same geographical location, for instance different teams on the same floor, and other dependencies occur between teams located overseas, especially with regulatory agencies. The architect located in India explained a situation where the algorithm team was in the United States of America. Moreover, he gave another example of dependencies between different teams that are geographically separated: 'Our guy [concept coach] who is sitting in [German city] is interfacing with the clinical coach and the team here [India] for all the processes part'.

Furthermore, the employees reported different types of dependencies between teams; the architect described, 'The algorithm team is from the United States, a Customer Representative team. They give us the deliveries for any new algorithm or any new features that needs algorithm processing'. Also, the architect outlined a case where there are technical dependencies related to NFR.

In addition, there are dependencies related to the sequence of the workflow. For instance, the architect described a dependency between the architect team and the platform team: 'Now, we are reaching the feature phase. Are we going to deliver this new tool within the feature phase or not? So here we have a dependency to the platform team'. The senior manager, who is also the agile coach, described a similar situation: 'Sometimes the team has component dependency and needs something as input from other team'.

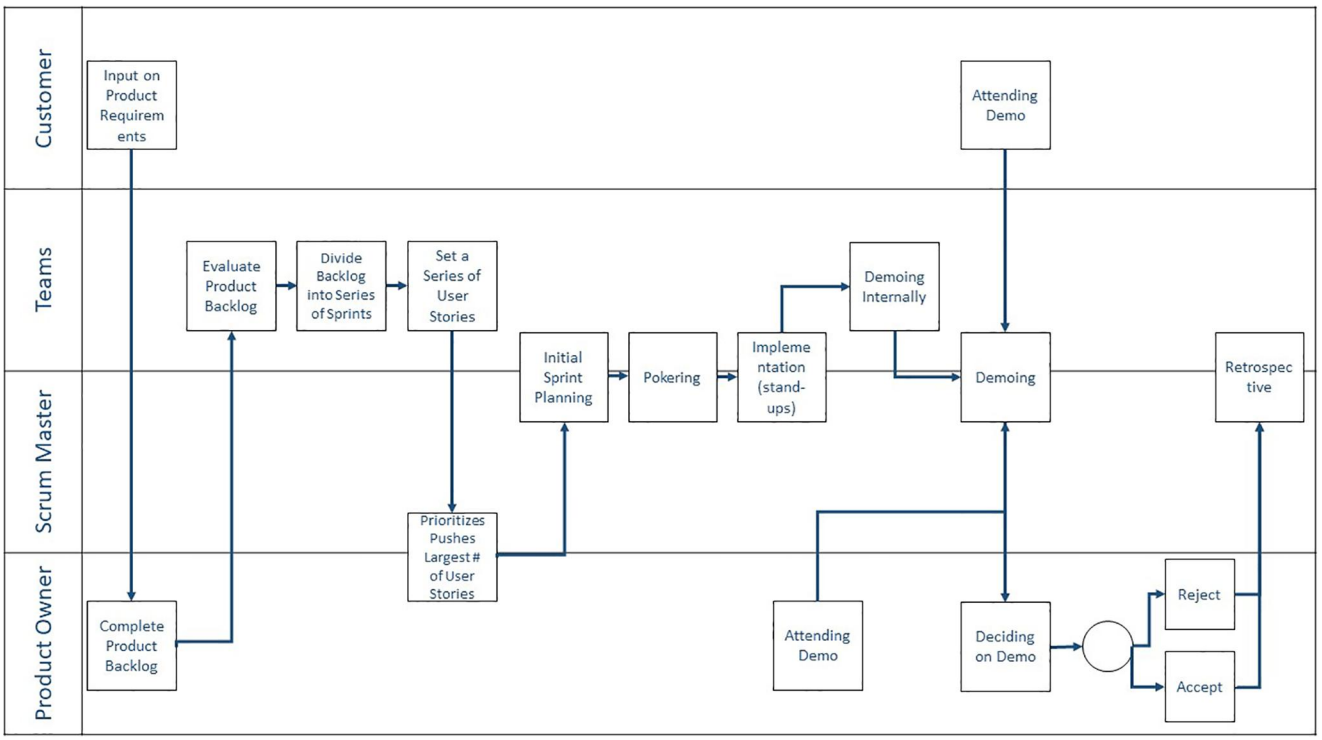

FIGURE 1 Swim lane diagram for DevelopCo

4.7 | HealthCo communication at the boundaries

Communication is encouraged between the team members. The agile coach, who is also the senior manager, highlighted the importance of communication regardless of the geographical location: 'We encourage people to talk to each other as much as possible regardless of which site they are'. The communication is organised through different group meetings that include sprint meetings, weekly reviews, and retrospectives, an information sharing platform, and face-to-face communication. In addition, even though it is not recommended as a platform for communication in the organisation, some employees use WhatsApp in order to connect to the other members in the team for any logistic reason or personal reason. In addition, the employees also use a physical board that aids in tracking the work of the different teams and employees. Furthermore, the physical board is used to identify the different backlogs and the priority of each.

4.8 | HealthCo communication and customer requirements

Customer involvement in the agile process is vital. When implementing agile, customers become major influencers in the software development process. The programme manager explained, 'Our Product Manager frequently meets our customers. He gets new ideas, so he wants to be very flexible in the backlog. The flexibility comes because the content is not predefined, the content could change'.

Communication is directly related to customer satisfaction. In the health sector software development, there are many stakeholders involved. The senior developer highlighted three main communication means that affect customer satisfaction. First, a line meeting is conducted that involves all stakeholders. Second, an agile presentation or demo is held. Third, the retrospectives aid in achieving customer satisfaction. Retrospectives are devoted for the improvement of the agile software development process and for adaptation to changes that arise, such as the customers' needs.

4.9 | HealthCo's approach to NFR

The healthcare sector does not follow pure agile software development methods. Instead, a mix of methods is used to guide the process. The senior manager, also an agile coach, explained, 'We have found a little bit of midway between a very pure agile development and the old style of waterfall development'. In addition, the programme manager estimated the following: 'The Agile we are implementing is not through 100% of the lifecycle of the product, it is 70% or 80%'. This is translated in the way HealthCo deals with NFR.

HealthCo tailors agile methods to deal with NFR. First, HealthCo associates different roles and titles to its employees, with 16 titles associated with different people and their respective roles. These roles include product manager, line manager, project manager, sub-segment head, test manager, team master, and so on. This reveals the combination between agile and traditional methods in HealthCo. The developer said, 'If we go by the pure Agile Scrum methodology, then we shouldn't have any other role other than the product owner and the Scrum master'. On the other hand, some employees take on double roles. For instance, the senior manager holds two responsibilities: line coach and project coach.

Second, the employees perform documentation regularly. One reason for the usage of documentation is quality assurance. Extensive documentation is used since there are international regulations that the company must comply with. For instance, the developer mentioned the FDA: 'There are documents because we are in the medical software [development industry] and are based on the FDA. They have test specifications, safety specifications, design specs, and safety integration test specifications'. These documents include Device Hazard Analysis, Unresolved Anomalies and Health-case Description.

4.10 | HealthCo process flow

This section presents the process flow for HealthCo using the swim lane diagram shown in Figure 2, focussing on one iteration. This swim lane diagram was derived from the memos deduced from the transcribed data. An initial diagram was drawn from the memos that described the information flow. Then, constant comparison technique was used to complete the diagram by accumulating information from memos that are not directly related to the information flow.

The diagram uses five swim lanes to display the main actors in the agile software development in HealthCo. The diagram shows the high influence of the Healthcare Regulatory Agencies on the product development and how they can insert changes to product specifications due to updates in universal medical regulations. As part of the agile process, the diagram shows how the customer is involved in the product development process. Initially, the evaluation and completion of the product backlog is based on the customer's input; lastly, the success of the sprint is highly based on the customer's evaluation of the team's demo. Furthermore, it displays the scrum master's role and highlights how the scrum master acts as a link between the different actors. It also portrays how common the inter-team communication is. The product owner involvement in the process is steady, especially with the weekly backlog grooming sessions that affect the two implementation phases in each sprint.

5 | DISCUSSION

We now analyse the findings from the two cases in the light of the research questions while identifying the differences and similarities.

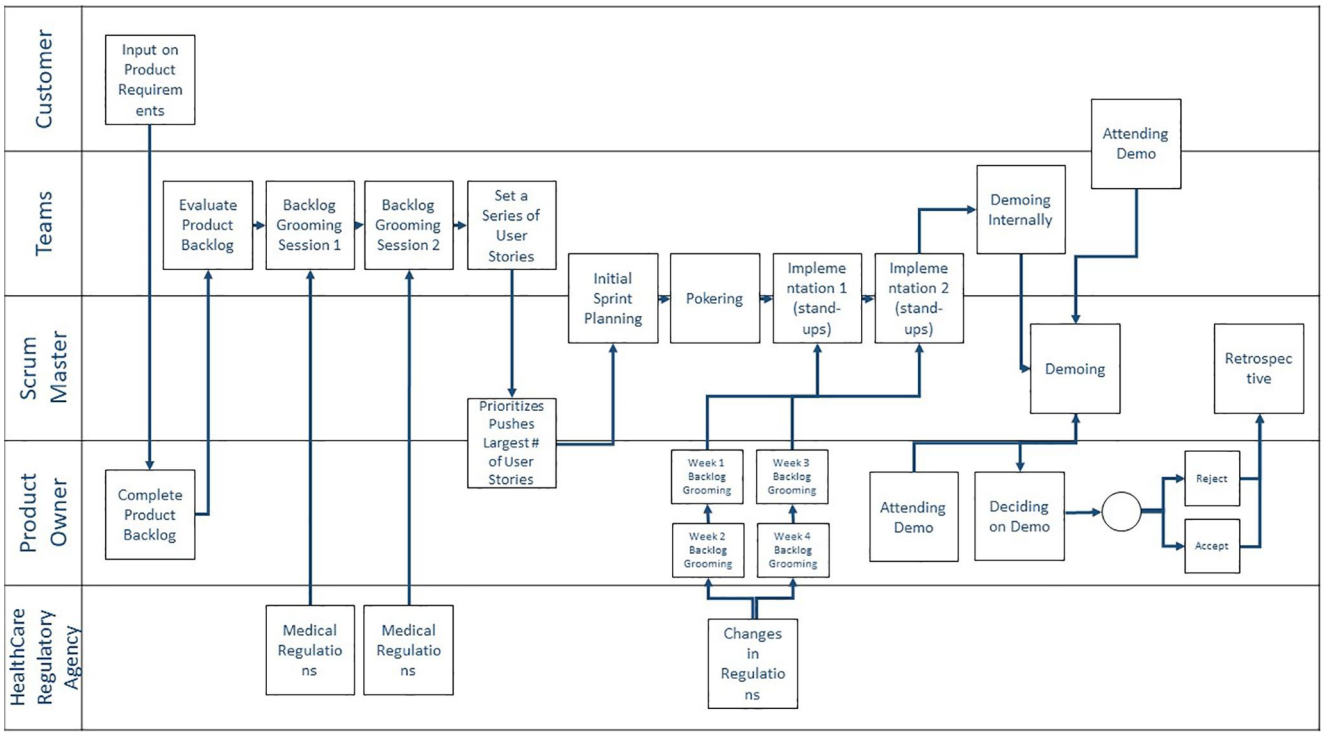

FIGURE 2 Swim lane diagram for HealthCo

First, the practitioners in both companies highlight the challenges faced at the boundaries. Communication at the boundaries is a key challenge for practitioners since it involves the work coordination of several stakeholders who have diverse backgrounds [51]. Practitioners viewed the Scrum masters' role as vital in encouraging communication and scrum coordination meetings [24]. Furthermore, practitioners share information across teams using different means such as a physical scrum board, digital scrum board, instant messaging tools, and internal demos. They stressed on the importance of product owners' communication with the customer that led to high levels of customer involvement in the software development process. In both cases, the regulatory agencies and the customer-agency-organisation triangle complicated the communication.

Second, practitioners emphasised how blurred inter-team dependencies negatively affect the workflow. Disturbed dependencies affect the recognition of NFR. Customer involvement in the process and the changes customers request amidst the development process also caused delays and increased pressure, especially on employees.

Third, agile tailoring was different in both companies. DevelopCo performs little agile tailoring since they are aiming at becoming 100% agile. Practitioners tailor agile methods in the transition state towards achieving a pure agile development process. Contrarily, HealthCo tailors agile methods and applies it along with some traditional waterfall techniques. Practitioners pointed out that they assume roles that are usually not present in agile software development. They revealed that they had to do lots of documentation in order to comply with international regulations [11], record testing and results [18], maintain quality, and comply with design specifications. They performed quality assurance through constant validation of products with developers and customers. Our study opposes previous literature that suggests that quality assurance is very supportive, and the benefits outweigh the inconvenience [11]. Practitioners found that some quality assurance methods were time consuming and sometimes unnecessary.

Additionally, there are numerous differences and similarities that were identified when drawing the swim lane diagrams of DevelopCo (Figure 1) and HealthCo (Figure 2). This was possible by superimposing the swim lane of the two companies. Figure 3 shows the diagrams overlapping, where the grey shaded practices are common for both companies. Two similarities are identified in both swim lane diagrams, customer involvement and scrum master's role as a *liaison*.

The swim lane diagrams revealed two main differences between DevelopCo and HealthCo represented in one extra horizontal lane for the Healthcare Regulatory Agency in Figure 2 and additional responsibilities represented in the added interaction and information flows in the product owner lane in Figure 2. The Agency provided the self-organising teams with medical regulations and guidelines in the first and second Backlog Grooming Sessions as well as the product owner during the Weekly Backlog Grooming Sessions. This caused interaction and information flow increase from the Agency towards the self-organising teams and the product owner and consequently to the scrum master. In

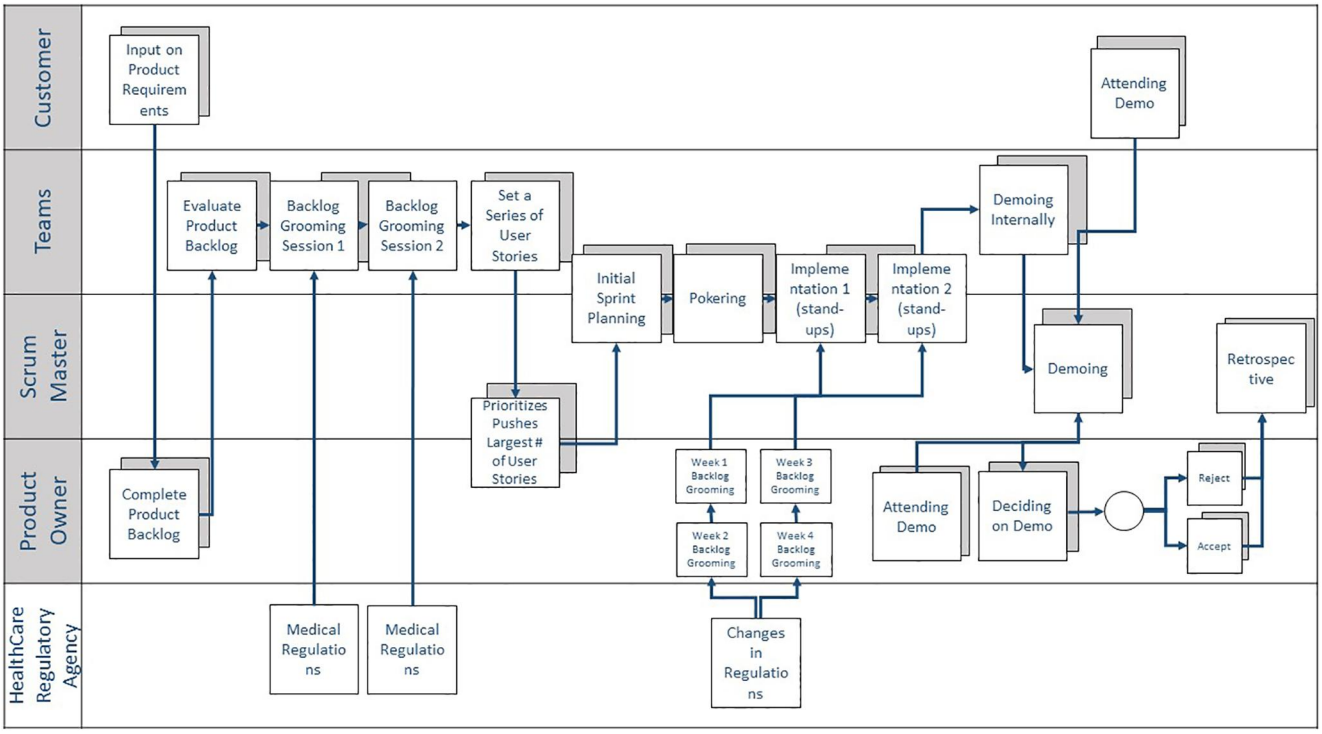

FIGURE 3 Super imposed swim lane infused FLOW diagrams

addition, the product owner is allocated additional responsibilities in HealthCo. The product owner communicates and coordinates with the healthcare regulatory agencies who provide the product owner with the necessary rules and regulations to ensure compliance. In turn, the product owner translates these guidelines into the product backlog. Then, the product owner and self-organising teams discuss the project implications and define the Definition of Done to ensure compliance.

Table 1 demonstrates an inventory of the similarities and differences between two companies in agile methods. This table is deduced from the swim lane diagrams and the approach to NFR. The table also points out the cases where agile methods are implemented in different intensities, the darker the colour the higher the intensity. High intensity reveals that the ceremony is implemented in its entirety as per the defined agile concepts. The asterisk (*) presents the agile methods that are general to Scrum. Table 1 shows that both companies are implementing Scrum effectively.

Our study shows a difference in the way the companies deal with NFR. HealthCo used a mix of Waterfall and agile methods to deal with NFR, while DevelopCo incorporated the NFR into the user stories in the agile process. There are three categories that represent the similarities and differences in the way both companies handle NFR:

First, the swim lane diagrams reveal that both companies have thoroughly implemented agile and highlight the importance of the customers and the scrum master's roles. The customer's involvement in the process is extensive in both cases and understanding the customer's requirements is vital. Here rises the need to consider the customer as one of the main actors in agile software development and to allocate a lane to represent the customer. Also, in both companies, the scrum master plays the role of a *liaison* between the self-organising teams and the product owner. The product owner prioritises the user stories while the scrum master provides input where there are dependencies. The scrum master additionally collaborates with the team through the following activities: sprint planning in initial sprint planning, estimation in pokering user stories, ceremony facilitation in daily stand-ups, and sprint reviewing in the retrospective [52]. This research supports earlier findings of the tailoring of the scrum master's role [24]. The extended scrum master's activities act as a resource for tailoring of agile methods in small and large development companies. These similarities portrayed in different companies are important elements of Scrum.

Second, the swim lane diagrams showed differences in the way NFR are dealt with. These differences lie in the additional actor represented in the swim lane diagram of HealthCo, 'Healthcare Regulatory Agency', and the demonstration of further responsibilities allocated to product owner also in HealthCo. The Agency was involved in the software development process extensively; thus an additional lane was allocated to it. Its input affected the initial and weekly backlog grooming sessions, thus directly impacting the product owner, self-organising teams, and the scrum master. Furthermore, the product owner completes the product backlog, decides on the demo, and collaborates with the scrum master to prioritise user stories. However, in HealthCo, the product owner undertakes

TABLE 1 Inventory for agile methods

Agile methods		DevelopCo	HealthCo
Roles	Scrum master*	x	x
	Product owner*	x	x
	Self-organising teams*	x	x
Ceremonies	Customer input on product requirements*	x	x
	Evaluate product backlog*	x	x
	Divide backlog into series of sprints*	x	x
	Set series of user stories*	x	x
	Pokering user stories*	x	x
	Sprint planning*	x	x
	Weekly backlog grooming session		x
	Daily standup*	x	x
	Changes to product guidelines		x
	Demoing internally	x	x
	Demoing*	x	x
	Retrospective*	x	x
Artefacts	Product increment*	x	x
	Sprint backlog*	x	x
	Product backlog*	x	x

new work functions to be able to cope with the regulated environment. In addition to the roles identified in previous literature [28], the product owner in HealthCo takes the changes in medical guidelines from agencies and translates them into the product backlog. Thus, each organisation deals with the NFR, caused by the external medical and healthcare requirements along with the governmental agencies, using different procedures. HealthCo tailors agile methods to comply with these NFR, whereas DevelopCo takes the NFR and embeds them to the agile process. These requirements are turned into user stories and agile tasks and added to the physical scrum board.

The third category is the differences in the ideology and culture of both companies, which lead to the major differences in the managing of NFR. The findings reveal that HealthCo applies tailoring methods highlighted in Table 1. DevelopCo is more ideologically driven since it is still a growing company in its initial stages. The employees are still junior employees in the company; thus they are open to change. Having their aim to be 100% agile, they take all NFR and turn them into agile tasks through user stories. On the other hand, HealthCo is a fully mature company; thus it is reluctant to fully apply agile. The employees have been working in the company for an average of 10 years and some employees have been in the company for 18 years. Although agile is implemented for 70%–80% of the development process, some activities are performed using a mix of Waterfall and agile.

5.1 | Implications for practitioners

Between the agile-driven method of DevelopCo and the mixed methods of HealthCo, we encourage that practitioners follow our recommendations when faced with regulated environments and vital NFR. The recommendations are explained in this section along with two illustrative examples. As a result of our 18-month study, DevelopCo demonstrates how any required action such as documentation, regulatory compliance, and safety critical work items can be implemented and embedded in an agile software development process. In terms of achieving regulatory compliance, documentation and agile software development are not exclusive since agile developers deliver what is requested by the customer [53, 54]. And in a regulated environment, the customer requires documentation and there are no barriers to prevent this production.

By using cross-case analysis and extensive studies of previous literature, this study developed new artefacts. Also, this study advocates the tailoring of agile methods, using the artefacts, rather than using a mixed method approach of agile and traditional methods. By careful and systematic analysis of our data and extensive study of the previous literature, we developed the undermentioned tables that present illustrative examples demonstrating how documentation and safety critical work items can be implemented in an agile process. We create two new artefacts: Documentation Work Item and Safety Critical Work Item. These artefacts allow the adoption of agile

while complying with the regulatory compliance. Documentation Work Item may include any document required to be submitted by the regulatory agency such as Software Description or Device Hazard Analysis. Safety Critical Work Item includes solid safety architectures and integration of safety features.

Table 2 shows an illustrative example of how practitioners can deal with documentation work items using agile methods. The technical lead gives an example on documentation work items: 'User Experience Design is a necessary document'. The table shows the recommendation, detailing the type and description.

Table 3 shows an illustrative example of how practitioners can deal with safety critical requirements work items using agile methodology such as the one described by Architect 2: 'Quality assurance documentation is a requirement by the FDA.' There are certain regulations and NFR that are applicable across the entire development process. Such regulations may be included in the team's definition of Done, which is considered as a consistent set of accepted criteria applicable to all backlog items.

Tables 2 and 3 can be generalised to companies applying agile in regulated environments when dealing with documentation requirements, such as Regulation–Business Case Matrix, and safety critical requirements, Device Hazard Analysis. This improves knowledge on how to normalise the agile methods implementation in such environments [25].

5.2 | Limitations

Standards of research quality have been achieved through realising representativeness of findings, reproducibility of findings, rigour of methods, and generalisability of finding through confirmability, dependability, internal consistency, and transferability [55]. Seeking to achieve confirmability, our conclusions depend on the conditions and subjects of the study. The conditions of our study follow the qualitative research method that is described in detail. After interviewing practitioners, all the data under study was treated anonymously to eliminate any research biasness. Dependability of our

TABLE 2 Illustrative example of using agile methods to deal with documentation work items

Type	Class	Description
User story	Artefact	Product owner creates user story for documentation purposes.
Definition of done	Artefact	Product owner creates a new definition of done for the documentation purposes.
Product backlog	Artefact	The documentation user story is added to the product backlog.
Product backlog	Artefact	The product owner prioritises user stories in the backlog including any documentation.
Product backlog	Artefact	Identify the high-level product backlog for the documentation.
Sprint planning	Ceremony	Self-organising teams will then estimate all user stories.
Sprint planning	Ceremony	Select the highest priority user story including any documentation user story for the next sprint.
Documentation work item	Artefact	Self-organising teams will work on the documentation user story until it achieves the definition of done.
Demo	Ceremony	Demo the work item (documentation) and the PO decides if the work is of acceptable quality.

TABLE 3 Illustrative example of using agile methods to deal with safety critical work items

Type	Class	Description
User story	Artefact	Product owner creates user stories for any safety critical requirements.
Definition of done	Artefact	Product owner creates a new definition of done for the safety critical requirements. (Since the code is subject to more stringent quality control processes)
Product backlog	Artefact	The safety critical requirement user story is added to the product backlog
Product backlog	Artefact	The product owner prioritises the user stories in the backlog including any safety critical requirement user stories.
Product backlog	Artefact	Identifying the high-level product backlog for the safety critical requirements.
Sprint planning	Ceremony	Self-organising teams will then estimate the user stories including any safety critical requirements user story.
Sprint planning	Ceremony	Select the highest priority user story including any safety critical requirement user stories for next sprint
Safety critical work item	Artefact	Self-organising teams will work on safety critical requirement user story until it achieves the definition of done.
Demo	Ceremony	Demo the work item; PO decides if the work is of acceptable quality.

conclusions is achieved by collecting data from multiple respondents in two different sites to avoid bias. Internal consistency is attained by first ensuring that research sites implement agile and do not simply from the textbooks. In addition, internal credibility is attained by having independent interviews between the practitioner and the interviewer without observation from other employees in the organisations. The complexity of the product software development made it unmanageable to cover the entire process. Thus, it was decided to cover the backlog creation phase to the retrospective phase of the teams' general flow of information. Furthermore, these phases are of high importance, they illustrate the agile tailoring methods, if any, and highlight the communication methods used. Transferability is reached by collecting data from two diverse companies implementing agile and are faced with NFR. This diverseness allows us to present ways in which NFR are dealt with while providing authentic evidence of current practices.

6 | CONCLUSION

In parts of the software development sector, NFR are highly significant. The literature on NFR in agile requires further development. We chose to investigate the issue of agile methods' tailoring by looking at it from an information flow perspective to understand how stringent NFR are managed. We created an inventory that reveals that both companies implement agile practices advocated in Scrum. In addition, we observed that each company takes a different approach to managing NFR. One company adopts conventional non-agile approaches to handling regulatory requirements, whereas the other company converts NFR into agile user stories and handles them as agile tasks. This observation enabled us to present illustrative examples, demonstrating how documentation and safety critical work items can be implemented in an agile process.

This qualitative study is informed by a grounded theory approach as the basis for implementation. The findings of each company were developed separately and independently. They consisted of a case description and a swim lane diagram for each company.

This research contributes to the study of agile methods' tailoring in relation to inter-team boundaries and NFR. The comparison of the two companies along with the swim lane diagrams' analysis enabled us to generate an inventory of common agile roles, artefacts, and ceremonies, which reveals how both companies are successfully implementing Scrum. Our data reveals that the essence of agile implementation lies in these practices since they are common for these two diverse companies, while stressing the importance of the scrum master's role as a liaison and the importance of understanding the customer's requirements. This inventory helped us identify the different ways both companies deal with NFR. HealthCo deals with NFR by tailoring agile methods, whereas DevelopCo incorporates NFR into the agile methodology.

Thus, the main contribution of this study is a set of proposed recommendations to deal with NFR, documentation, regulatory compliance, and safety critical work items, in a regulated environment using agile techniques. We introduce two new artefacts to the agile process: Documentation Work Item and Safety Critical Work Item, accompanied by illustrative examples that aids practitioners to handle documentation and safety critical requirements using agile.

CONFLICTS OF INTEREST

The authors declare no conflicts of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

ORCID

Scarlet Rahy https://orcid.org/0000-0002-6579-6818

REFERENCES

1. Abrahamsson, P., Conboy, K., Wang, X.: Lots done, more to do: the current state of agile systems development research. Eur. J. Inf. Syst. 18(4), 281–284 (2009)
2. CollabNet, a., Version One, a.: The 12th Annual State of Agile Report: Long Term Survey. VersionOne, Atlanta (2018)
3. Šmite, D., et al.: Software teams and their knowledge networks in large-scale software development. Inf. Softw. Technol. 86, 71–86 (2017). https://doi.org/10.1016/j.infsof.2017.01.003
4. Dingsøyr, T., Moe, N.B., Seim, E.A.: Coordinating knowledge work in multiteam programs: findings from a large-scale agile development program. Proj. Manag. J. 49(6), 64–77 (2018). https://doi.org/10.1177/8756972818798980
5. Bass, J.M.: Artefacts and agile method tailoring in large-scale offshore software development programmes. Inf. Softw. Technol. 75, 1–16 (2016)
6. Hoda, R., Noble, J., Marshall, S.: Self-organizing roles on agile software development teams. IEEE Trans. Softw. Eng. 39(3), 422–444 (2013). https://doi.org/10.1109/TSE.2012.30
7. Santos, V., Goldman, A., de Souza, C.: Fostering effective inter-team knowledge sharing in agile software development. Empir. Softw. Eng. 20(4), 1006–1051 (2015). https://doi.org/10.1007/s10664-014-9307-y
8. Dingsøyr, T., et al. (eds.): Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation: XP 2014 International Workshops, Rome, Italy, May 26–30, 2014, Revised Selected Papers. Springer (2014)
9. Pikkarainen, M., et al.: The impact of agile practices on communication in software development. Empir. Softw. Eng. 13(3), 303–337 (2008)
10. Doolin, B., McLeiod, L.: Sociomateriality and boundary objects in information systems development. Eur. J. Inf. Syst. 21(5), 570–586 (2012). https://doi.org/10.1057/ejis.2012.20
11. Fitzgerald, B., et al.: Scaling agile methods to regulated environments: an industry case study. In: Proceedings of the 2013 International Conference on Software Engineering, pp. 863–872. (2013)
12. Boehm, B.: Get ready for agile methods, with care. Computer. 35(1), 64–69 (2002)
13. Chung, L., et al.: Non-functional Requirements in Software Engineering. Springer Science & Business Media (2012)
14. Domah, D., Mitropoulos, F.J.: The NERV methodology: A lightweight process for addressing non-functional requirements in agile software development. In: SoutheastCon 2015. IEEE (2015). https://doi.org/10.1109/SECON.2015.7133028
15. Aljallabi, B.M., Mansour, A.: Enhancement approach for non-functional requirements analysis in agile environment. In: 2015 International

Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE), pp. 428–433. (2015)
16. Arbain, A.F.B., Ghani, I., Kadir, W.M.N.W.: Agile non-functional requiremnents (NFR) traceability metamodel. In: 2014 8th Malaysian Software Engineering Conference (MySEC), pp. 228–233. (2014)
17. Rottier, P.A., Rodrigues, V.: Agile development in a medical device company. In: Agile 2008 Conference, pp. 218–223. (2008)
18. Rasmussen, R., et al.: Adopting agile in an FDA regulated environment. In: Agile Conference, 2009, AGILE'09, pp. 151–155. (2009)
19. Kamal, T., Zhang, Q., Akbar, M.A.: Toward successful agile requirements change management process in global software development: a client–vendor analysis. IET Softw. 14, 265–274 (2019)
20. Vishnubhotla, S.D., Mendes, E., Lundberg, L.: Investigating the relationship between personalities and agile team climate of software professionals in a telecom company. Inf. Softw. Technol. 126, 126 (2020). https://doi.org/10.1016/j.infsof.2020.106335
21. Šmite, D., et al.: Software teams and their knowledge networks in large-scale software development. Inf. Softw. Technol. 86, 71–86 (2017)
22. Gal, U., Lyytinen, K., Yoo, Y.: The dynamics of IT boundary objects, information infrastructures, and organisational identities: the introduction of 3D modelling technologies into the architecture, engineering, and construction industry. Eur. J. Inf. Syst. 17(3), 290–304 (2008)
23. Bass, J.M., Haxby, A.: Tailoring product ownership in large-scale agile projects: managing scale, distance, and governance. IEEE Softw. 36(2), 58–63 (2019)
24. Cawley, O., Wang, X., Richardson, I.: Lean/agile software development methodologies in regulated environments–state of the art. In: Lean Enterprise Software and Systems, pp. 31–36. Springer (2010)
25. Heeager, L.T., Nielsen, P.A.: A conceptual model of agile software development in a safety-critical context: a systematic literature review. Inf Softw. Technol. 103, 22–39 (2018). https://doi.org/10.1016/j.infsof.2018.06.004
26. Fitzgerald, B., Russo, N., O'Kane, T.: An empirical study of system development method tailoring in practice. ECIS 2000 Proc. 4 (2000)
27. Bass, J.M.: Influences on agile practice tailoring in enterprise software development. In: 2012 Agile India. (2012). https://doi.org/10.1109/AgileIndia.2012.15
28. Bass, J.M.: How product owner teams scale agile methods to large distributed enterprises. Empir. Softw. Eng. 20(6), 1525–1557 (2015)
29. Behutiye, W., et al.: Non-functional requirements documentation in agile software development: challenges and solution proposal. In: International Conference on Product-Focused Software Process Improvement, pp. 515–522. (2017)
30. Asghar, S., Umar, M.: Requirement engineering challenges in development of software applications and selection of customer-off-the-shelf (COTS) components. Int. J. Softw. Eng. 1(1), 32–50 (2010)
31. Farid, W.M., Mitropoulos, F.J.: NORPLAN: non-functional requirements planning for agile processes. In: 2013 Proceedings of IEEE Southeastcon, pp. 1–8. (2013)
32. Ramos, F.B.A., et al.: A non-functional requirements recommendation system for scrum-based projects. In: The 30th International Conference on Software Engineering & Knowledge Engineering, pp. 149-148. (2018)
33. Maiti, R.R., Mitropoulos, F.J.: Capturing, eliciting, predicting and prioritizing (CEPP) non-functional requirements metadata during the early stages of agile software development. In: SoutheastCon 2015, pp. 1–8. (2015)
34. Kassab, M., Daneva, M., Ormandjieva, O.: Early Quantitative Assessment of Non-functional Requirements. University of Twente Report (2007)
35. Rao, A., Gopichand, M.: Four layered approach to non-functional requirements analysis. Int. J. Comput. Sci. Issues. 8(6), 371–379 (2011)
36. Kurtanović, Z., Maalej, W.: Automatically classifying functional and non-functional requirements using supervised machine learning. In: 2017 IEEE 25th International Requirements Engineering Conference (RE), pp. 490–495. (2017)
37. Sachdeva, V., Chung, L.: Handling non-functional requirements for big data and IOT projects in scrum. In: 2017 7th International Conference on Cloud Computing, Data Science & Engineering-Confluence, pp. 216–221. (2017)
38. Ameller, D., et al.: Dealing with non-functional requirements in model-driven development: a survey. IEEE Trans. Softw. Eng. 47(4), 818–835 (2019)
39. Charmaz, K.: Constructing Grounded Theory, 2nd ed. SAGE, Los Angeles, CA (1939/ 2014)
40. Seaman, C.B.: Qualitative methods in empirical studies of software engineering. IEEE Trans. Softw. Eng. 25(4), 557–572 (1999)
41. Stol, K., Ralph, P., Fitzgerald, B.: Grounded theory in software engineering research: a critical review and guidelines. In: 2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE). (2016). https://doi.org/10.1145/2884781.2884833
42. Patton, M.Q.: Qualitative Research and Evaluation Methods, 3rd ed. SAGE, London (1945–2002)
43. Palinkas, L.A., et al.: Purposeful sampling for qualitative data collection and analysis in mixed method implementation research. Adm. Pol. Ment. Health. 42(5), 533–544 (2015)
44. Gerring, J., Cojocaru, L.: Case-selection: a diversity of methods and criteria. Sociomethods Res. 45, 392–423 (2016)
45. Rahy, S., Bass, J.: Reserach Participant Summary. University of Salford Dataset (2020)
46. Bass, J.: Tailoring in Large Scale Agile, Interview Guide. (2018). https://salford.figshare.com/articles/Tailoring_in_Large-Scale_Agile_Interview_Guide/7122503
47. Vaivio, J.: Interviews – learning the craft of qualitative research interviewing. Eur. Account Rev. 21, 186–189 (2012). https://doi.org/10.1080/09638180.2012.675165
48. NVivo Qualitative Data Analysis Software, Version 11 ed. QSR International Pty Ltd (2015)
49. Adolph, S., Hall, W., Kruchten, P.: Using grounded theory to study the experience of software development. Empir. Softw. Eng. 16(4), 487–513 (2011). https://doi.org/10.1007/s10664-010-9152-6
50. Cruzes, D.S., et al.: Case studies synthesis: a thematic, cross-case, and narrative synthesis worked example. Empir. Softw. Eng. 20(6), 1634–1665 (2015)
51. Barrett, M., Oborn, E.: Boundary object use in cross-cultural software development teams. Hum. Relat. 63(8), 1199–1221 (2010)
52. Noll, J., et al.: A study of the scrum master's role. In: PROFES 2017: Product-Focused Software Process Improvement, pp. 307–323. Springer. (2017). https://doi.org/10.1007/978-3-319-69926-4_22
53. McHugh, M., McCaffery, F., Casey, V.: Barriers to adopting agile practices when developing medical device software. In: International Conference on Software Process Improvement and Capability Determination, pp. 141–147. (2012)
54. Berard, E.V.: Misconceptions of the agile zealots. In: Software and System Process Improvement Network. (2003)
55. Lincoln, Y.S.: Naturalistic inquiry. In: George, R. (ed.) The Blackwell Encyclopedia of Sociology, pp. 131–137. Wiley (1985)

How to cite this article: Rahy, S., Bass, J.M.: Managing non-functional requirements in agile software development. IET Soft. 16(1), 60–72 (2022). https://doi.org/10.1049/sfw2.12037